Australian National University
Department of Engineering

Autonomous Submersible Robot: State Estimation System

Terence Betlehem

A thesis submitted in partial fulfilment of a Bachelor of Engineering

Supervisors: Mr. Samer Abdallah (FEIT),
Dr. David Wettergreen (RSISE),
Dr. Alex Zelinsky (RSISE)

November 1999

ABSTRACT

The aim of this project was to develop the state estimation system for an autonomous underwater vehicle called Kambara. This project took place at the Research School of Information Sciences in cooperation with the Department of Engineering at the ANU. The purpose of the state estimation system was the filtering of data from a triaxial accelerometer, rate gyroscope, depth sensor and compass/inclinometer to track the position, orientation, linear velocity and angular velocity of the Kambara.

To accomplish this, an extended Kalman filtering algorithm was proposed. Nonlinear models were derived for both the sensors and the dynamics of the Kambara. Device drivers for the underlying hardware architecture were developed so that data could be logged from the sensors onto computer. Static testing of the sensors has begun for the sake of modelling noise processes and temperature sensitivities. The performance enhancements of prefiltering the sensor outputs with analog filters were investigated.

Plenty of scope is left for future work on the state estimation system. Further characterisation of the sensors will be required. Coding of the Kalman filter algorithm and parts of the sensor-sampling algorithm are still required. Eventually, the state estimator will be implemented onto the Kambara computer and the whole system tested and tuned. All the while, hydrodynamic and geometric parameters need to be estimated for implementation of the dynamic model.

ACKNOWLEDGMENTS

Throughout this project, I have received much help from many a different source. I would like to thank the Kambara Team at the Research School of Information Science and Engineering for the assistance, guidance and advice given over the course of the last 9 months. In particular, thanks must go to my supervisor David Wettergreen for the help and leadership he has provided over the last 9 months, not to mention the patience he has shown in many a trying time. Not only has he helped me solve many trivial computer problems, but also given me much insight into the big picture. Chris Gaskett has also been very helpful, providing plentiful quality advice. Harley Truong was always quick to provide technical assistance.

I must thank Bruce Mascord and Martin Stonebridge for the helpful advice given toward the sensor suite design and their fast service in machining the parts of the sensor mounting.

Thanks also to the guys who made the undergraduate laboratory just that little bit brighter a place. In particular to Ian Fitzgerald, Steven Harney, Aasif Javed, Chanop Silpa-Anan, Luke Fletcher and Rosalie Murray (over email) for the company you have all provided. To Ian especially for always being prepared to answer any dumb question I threw at you.

Thank you to all of my supervisors, David Wettergreen, Samer Abdallah and Alex Zelinsky, for giving me the opportunity of working on such an interesting and exciting project in a highly professional environment. The project has taught me a lot and given me useful practical experience to take into the future. Also, the feedback you guys have given me over my project proposal, progress report and draft thesis has been helpful.

Special thanks go out to my family and friends for their support and encouragement.

STUDENT CONTRIBUTIONS

As part of my project, I have completed the following works.

- A dynamic model of AUV and sensors was researched and developed. Both a literature survey and much hand derivation have gone into it. This model has been implemented in Matlab for calculation of Jacobian derivatives.
- Attendance of the lectures of a Kalman Filtering course at ADFA has enhanced my understanding and allowed me to make several important decisions in the design of the Kalman filter.
- Suitable sensor mounting was designed for mounting into the Kambara environment and onto the WAM arm.
- Work has been completed on three device drivers.
    - Code has been contributed to the PCI-9080 carrier board driver of David Wettergreen so as to allow processing of interrupts.
    - The IP-Precision ADC device driver of Bob Edwards has been completely re-written for the IP-ADIO module.
    - The IP-Serial device driver of Bob Edwards was bugfixed and extended.

  Both the IP-ADIO and IP-Serial drivers has been adapted for use on a little endian machine and for use with the carrier board driver.
- Code has been developed for state estimation algorithm. The matrix/vector library was adapted for use under the VxWorks operating system and its functionality extended. A TCM2 compass layer has been coded but still requires testing and addition of few more functions.
- Help was given to Chris Gaskett in development of his dynamic model of the Kambara.
- A simple constant acceleration Kalman filter was designed for the state estimation of the velocity of the Nomad tour-guide robot. This filter was written in C. Further work in filter tuning and implementation are required however.
- Experiments on properties of the ADC's, Summit Instruments triaxial accelerometer and MotionPak were conducted through the use of a series of static sensor tests. The temperature dependence of zero point offsets and noise distributions were investigated. The dependence of noise in ADC's was studied as a function of sampling rate.
- The use of RC filters in filtering the output from the MotionPak was explored.
- Several properties of the TCM2 compass module were investigated. In particular the affects of powering the thrusters in its presence were studied. Compass accuracy was also explored.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 - INTRODUCTION

1.1 Project Background

The Robotic Systems Laboratory (RSL), located within the Research School of Information Sciences and Engineering (RSISE) at the Australian National University (ANU), is concerned with the research and implementation of robots working in real time for real world applications. Some of the projects that the RSL have undertaken involve mobile robot navigation, active vision, and robot learning systems. Over the last two years, the RSL has been developing an underwater vehicle suitable for tasks in exploration and inspection.

The ocean covers 70% of the earth's surface. However up until now, relatively few resources have been put into its exploration. This contrasts starkly with the huge amounts of time and money put into space exploration. In the coming decades, a large growth in underwater exploration is expected.

This growth is to be assisted by the development of underwater vehicles. A vehicle able to map out underwater regions could result in the discovery of many biological and mineralogical resources. Capabilities of following and observing marine life could assist in gaining a better understanding of our oceans. Delicate underwater environments such as the Australian Great Barrier Reef could be monitored to check whether tourist-produced pollution levels are acceptable. Also, substantial underwater structures such as oilrigs are often in need of inspection and maintenance. Underwater vehicles or robots are appropriate for application in all these areas.

Figure 1.1: The Great Barrier Reef, a resource in need of protection

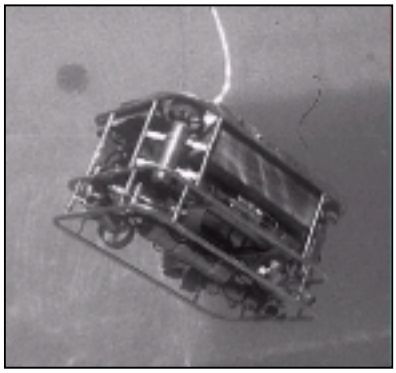


Figure 1.2: The Romeo research ROV

At the moment, a relatively large number of underwater robots are in use around the world. However, they are typically of the remotely operated vehicle (ROV) variety, such as the Romeo research ROV shown in figure 1.2. (See Bono [4] for details of this ROV.) The control of such robots requires constant human supervision by specially trained personnel. So it is desirable to design a vehicle that is autonomous. That is to say, instead of having its every movement controlled by human, only occasional supervisory commands need be provided, such as 'hold current position' and 'swim after target'.

The RSL has undertaken the construction of such an autonomous underwater vehicle (AUV). The main objectives of the project are to enable the AUV to autonomously search in regular patterns, follow along static features, observe the surrounding dynamical features and swim after moving targets. Key aims are to further the research work in the realm of underwater vehicle control and visions systems. In particular, the challenges of implementing a model-free reinforcement learning system for motion control and visual servo-control are to be investigated (Gaskett C., Wettergreen D., Zelinsky A [8]).

The AUV has been named Kambara, an aboriginal word for crocodile. The Kambara's mechanical structure is an aluminium open frame suitable for the mounting of five thrusters and two cylindrical enclosures. The frame has length, width, and height of 1.2m, 1.5m and 0.9m, respectively. The assembled vehicle displaces approximately 120 litres of water.

Figure 1.3: Kambara front view

Mounted in the upper enclosure are a computer system, frame grabber, analog-to-digital converters (ADCs), and communications equipment. A pan-tilt-zoom camera looks out the front endcap. Also in the upper enclosure is a sensor suite consisting of a triaxial accelerometer, compass, biaxial inclinometer, rate gyroscope and temperature sensor.

The lower enclosure contains six lead acid batteries, power distribution and charging circuitry, a depth sensor and a leak sensor. The batteries are 12V and capable of supplying up to 1200W of power. Mounted on the bottom plate are five watertight boxes, each containing an amplifier for boosting the control signals to the thrusters. Mounted on the front frame is a pair of stereo cameras.

The majority of the processing power required for operation of Kambara is present on-board. This is principally composed of a G3 PowerPC running on CompactPCI backplane architecture. VxWorks is used as the operating system. Its main tasks are simultaneous filtering, vision computation and communication. Eventually, autonomous control will be moved on-board too. The important hardware components for the scope of this project are industry pack (IP) modules for analog to digital conversion, digital I/O and serial communication. The state estimation system uses a combination of ADCs and a serial port for logging of sensor data. These modules interface with the PowerPC through a carrier board plugging into the PCI backplane. A Motorola 68332 module is also present. This processor manages servo control of the five thrusters through pulse width modulated control signals.

1.2 Project Overview

Work on the inertial navigation system (INS) of the Kambara was commenced by Beswick [2]. The purpose of the INS was to provide adequate sensor data to estimate the current state of the AUV in real time. State refers to the position, orientation, linear velocity and angular velocity of the AUV. This state information is utilised by the motion control system to navigate the vehicle.

Beswick's work involved the selection, acquisition and testing of several sensors for integration into the INS. The following sensor units were selected:

- Summit Instruments triaxial accelerometer unit
- Precision Navigation TCM2 compass unit integrating a biaxial inclinometer and magnetic compass
- Sensor Technics pressure sensor, enabling depth measurements of up to 3m,
- Systron Donner MotionPak combining a rate gyroscope and triaxial accelerometer

This thesis is concerned with the design and implementation of a state estimation system. Such a system converts raw sensor data into estimates of the state of the AUV. It must tackle a number of different problems. Sensor readings contain both noise and zero offset biases. It is desirable to reduce the noise levels and correct for any biases in these sensor outputs. Also, the sensors do not provide state estimates directly. For example, the triaxial accelerometer provides the acceleration at the position of the sensor, so this data must be processed to determine the velocity at the centre-of-mass of the AUV. A model (ie. a set of governing equations) must be used to convert the sensor readings into state estimates. A particular type of state estimation algorithm that uses a model to filter the sensor data is called a Kalman filter.

Beswick proposed the use of a Kalman filter incorporating a linear model of the AUV and the sensors. In doing so, he was forced to make small angle approximations, as well as ignore hydrodynamic added mass and gyroscopic acceleration terms in the modelling of the AUV. In this thesis, a more realistic, nonlinear model is developed and incorporated into a modified version of the Kalman filter called an extended Kalman filter.

Integral to the Kalman filter is a model of the sensors. Not only must the kinematics involved with the positioning of the sensors be considered, but also factors such as sensor sensitivity, noise characteristics and temperature sensitivity must be determined. In this thesis, the deterministic components of this model have been derived. The noise characteristics of two of the sensors have been characterised. From these results, the statistical component of the model has been developed.

Furthermore, for the implementation of the state estimator into a real time environment, coding of the sensor sampling algorithm and underlying drivers must be implemented. Over

the duration of the project, device drivers for the carrier board, ADCs and serial port have been either written or adapted to the purpose of sampling the sensors. As a part of the ADC device driver work, a significant amount of performance testing has been achieved.

Finally, the sensors must be physically mounted on the Kambara. Suitable mounting has been designed and implemented as part of the project work. This mounting was also designed for attachment to the wrist of the Whole Arm Manipulator for future dynamic testing of the sensors.

1.3 Thesis Overview

This thesis is concerned with laying down the foundations for the design and implementation of an accurate and robust state estimation system. Brief descriptions of each chapter and appendix follow.

Chapter 2 derives a dynamic model of the AUV. It includes application of Newton's second law, consideration of the major forces, and discretisation of the resulting differential equations.

Chapter 3 discusses the important features of the sensors and their final implementation. It describes their distinguishing features and lists data on the most important sensor characteristics. It also documents power distribution and the design of physical mounting.

Chapter 4 derives the equations describing the behaviour of the sensors.

Chapter 5 describes the extended Kalman filter algorithm, as it applies to Kambara.

Chapter 6 documents the current hardware and software heirarchy used for the state estimator. In particular, it describes the current implementation of the analog-to-digital converter and serial drivers.

Chapter 7 describes the sensor testing that has been performed as part of the project.

Chapter 8 outlines a number of conclusions coming out of the project, as well as the future work required for final implementation of the state estimation system.

Appendix 1 contains tables of all the symbols used in the sensor and system models.

Appendix 2 describes the thruster model developed by Silpa-Anan [11].

Appendix 3 tabulates the sensor sensitivity and bias data so far obtained.

CHAPTER 2 - PROCESS MODELLING

2.1 Introduction

An integral part of the Kalman filtering algorithm is the process model. A process model is a set of equations describing the current *state* of the AUV as a function of the *state* at a previous time and the control inputs. By *state*, one refers to the parameters that are important to measure for estimation of the position, attitude, linear velocity, and angular velocity of the AUV. In the case of Kambara, the control inputs are the thruster forces and torques.

An accurate process model is desirable. The Kalman filter uses such a model to make future predictions from estimates of the current state. The better the model is at predicting future states, the more the model can be relied upon to filter out measurement noise, and the more effective the Kalman filter will be at removing such noise from the sensor readings.

The ultimate aim of process modelling is to estimate the next state $\boldsymbol{x}_{k+1}$ of the AUV by a stochastic difference equation in terms of the current state vector $\boldsymbol{x}_k$, control inputs $\boldsymbol{u}_k$ and noise process $\boldsymbol{v}_k$. That is, the vector function

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{v}_k) \qquad \textbf{(2.1)}$$

is to be determined. This chapter focuses on developing the deterministic component of the process model. I.e. it develops an equation of form $\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k, 0)$. Incorporation of the noise component $\boldsymbol{v}_k$ is discussed briefly in chapter 5. It is better discussed after implementation of the state estimator, as $\boldsymbol{v}_k$ will need to be tweaked for filter tuning.

2.2 Steps in Model Derivation

As the process model basically describes the motion of the AUV, it must include in itself the dynamics of the system. The system dynamics are obtained by application of Newton's second law. However, before Newton's second law can be applied, suitable reference frames and attitude representations must be defined, and the significant forces and moments identified. Since the model is implemented in a digital system, it must be discretised. The major steps involved are

- define the reference frames (section 2.5),
- define the attitude representations (section 2.6),
- apply Newton's law (section 2.7),
- identify the forces and moments (section 2.8), and
- discretise the resulting equations (section 2.9).

The final equations are summarised in sections 2.8.5 and 2.9.

2.3 State Vector Definition

The very first step in developing any model is to decide which variables are to be modelled. The variables that become a part of the state vector $\boldsymbol{x}_k$ must be identified. In case of the Kambara, all variables describing the motion of the AUV must be included. In particular, these will be the position, attitude, linear velocity, and angular velocity of the AUV.

Another decision is in how exactly these state variables are specified. For example, one must decide which reference frame the velocity and acceleration are represented. Is it better to represent them in terms of the inertial reference frame, or a body-fixed reference frame (ie. a frame attached to the AUV)? This is an important design decision, as a bad choice of state vector can significantly increase model complexity. It turns out in the case of the Kambara, that since the sensors remain fixed in the local reference frame, the equations are simpler if linear velocity and angular velocity are specified in terms of a body-fixed reference frame.

The means of representing attitude (or orientation) also has several choices. Two common conventions used are Z-Y-X Euler angles and vector quaternions (see Craig [5]). As is explained in section 2.6, quaternions provide a more robust and computationally efficient representation of attitude so they have been chosen as the attitude representation.

In consideration of all these factors, the state vector $\boldsymbol{x}_k$ to be used is

$$\boldsymbol{x} = (\boldsymbol{v}^{\mathrm{T}}, \boldsymbol{\omega}^{\mathrm{T}}, \boldsymbol{r}^{\mathrm{T}}, \boldsymbol{q}^{\mathrm{T}}, T)^{\mathrm{T}}. \qquad \textbf{(2.2)}$$

The vector components of $\boldsymbol{x}$ are defined in table 2.1 below. For convenience, the time-step subscript k has been dropped. Note the inclusion of temperature in the state vector. The reason for this becomes apparent in chapter 4 where the temperature variable is used to correct for the temperature dependence of some of the sensors. As the vectors are each 3×1 and the quaternion is 4×1, the state vector contains 14 elements. (2.1) will hence be a system of 14 equations.

Symbol	Definition	Frame
$\boldsymbol{v}$	Velocity of centre of mass of AUV	Body-fixed
$\boldsymbol{\omega}$	Angular velocity of AUV	Body-fixed
$\boldsymbol{r}$	Position vector of centre of mass of AUV	Inertial
$\boldsymbol{q}$	Attitude quaternion of AUV	–
T	Average temperature of AUV sensors	–

Table 2.1: Definition of state vector variables, and the reference frames in which they are measured. Reference frames are fully defined in section 2.5.

2.4 Temperature Modelling

Before going on to develop the dynamic model, the temperature state variable T will first be considered. The importance of estimating T lies in its ability to correct for temperature-induced biases in sensor readings.

In general, the temperature inside the AUV will be a complicated function of the inside geometry, outside water temperature, the power dissipation properties of the electrical components and the heat transfer properties of the enclosure. However, a couple of fans are to be mounted inside the upper enclosure for the purpose of evening out the temperature distribution inside. Temperature should be approximately constant throughout this enclosure.

Fortunately, only a simple model is required. The sole purpose of this model is the smoothing out the temperature readings coming from the temperature sensor. As the temperature in the AUV will vary only slowly with time, taking hours to reach it's steady state value, it is reasonable to approximate temperature as constant. Ie. $\dot{T} = 0$.

2.5 Reference Frames

It is convenient to use two coordinate systems to describe the motion of the AUV. An inertial reference frame $\{I\}$ is used to describe the position of the AUV in global coordinates. A local or body-fixed reference frame $\{B\}$ is useful for specifying the parameters such as velocity and thruster forces, as these have much simpler representations in terms of such a reference frame. To simplify the kinematic equations, it is convenient for frame $\{B\}$ to be attached to the centre of mass of the Kambara. Both frames $\{I\}$ and $\{B\}$ are shown in figure 2.1. Note that $\boldsymbol{X_B}$ points towards the front of the AUV, $\boldsymbol{Z_B}$ points to the bottom of it, and $\boldsymbol{Y_B}$ points in a direction appropriate for forming a right handed coordinate system. The state vector component $\boldsymbol{r} = (x, y, z)^T$ then represents position vector of frame $\{B\}$ in frame $\{I\}$.

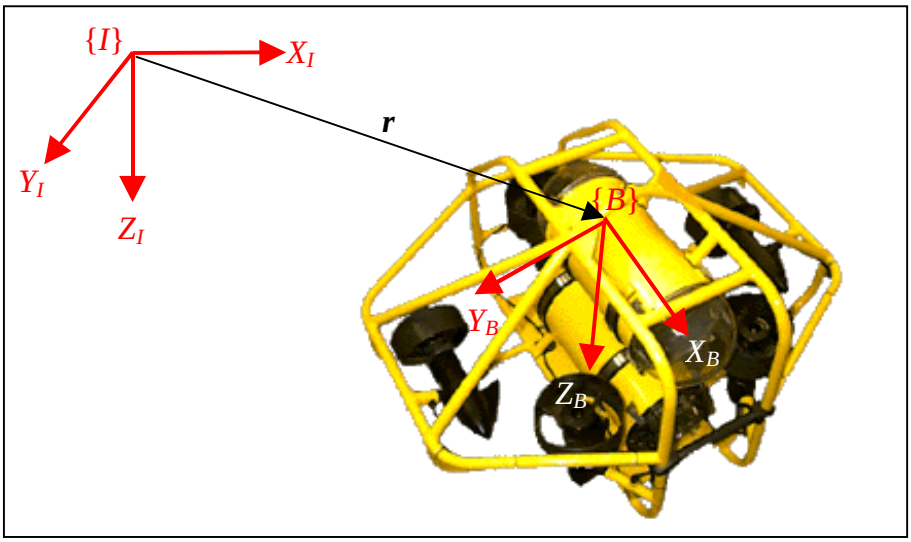


Figure 2.1: Definition of the inertial and body-fixed reference frames $\{I\}$ and $\{B\}$, and the vector $\boldsymbol{r}$ locating the origin of $\{B\}$

2.6 Attitude Representations

As indicated in section 2.3, there are different ways of representing the attitude (or orientation) of the AUV. Two common forms are Z-Y-X Euler angles and quaternions. These are described below. The reasoning for the choice of quaternions is mentioned.

2.6.1 Euler Angles

The orientation of frame $\{B\}$ in frame $\{I\}$ is commonly specified in terms of three Z-Y-X Euler angles: roll (ϕ), pitch (θ) and yaw (ψ). Figure 2.2 shows the definition of these angles.

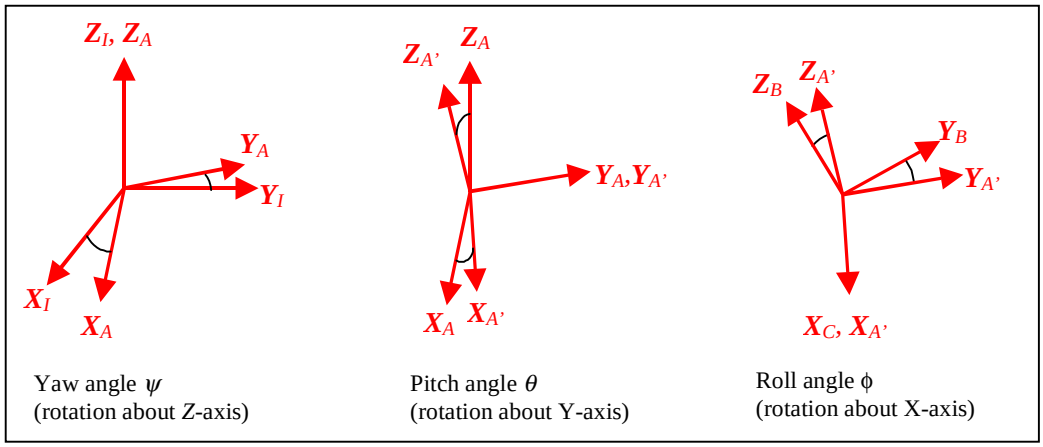


Figure 2.2: Definition of roll, pitch and yaw angles in terms of $\{I\}$, $\{B\}$ and two intermediate reference frames $\{A\}$ and $\{A'\}$. The positive sense of such angles is given by the right hand rule.

It is necessary to be able to convert parameters stated in terms of frame $\{B\}$ into inertial reference frame parameters. In particular, one needs to convert velocity $\boldsymbol{v}=(u,v,w)^{\mathrm{T}}$ and angular velocity $\boldsymbol{\omega}=(p,q,r)^{\mathrm{T}}$ both expressed in terms of frame $\{B\}$, into inertial velocity $\dot{\boldsymbol{r}}=(\dot{x},\dot{y},\dot{z})^{T}$ and Euler angle parameters $\dot{\boldsymbol{\Phi}}=(\dot{\phi},\dot{\theta},\dot{\psi})^{\mathrm{T}}$. This can be performed by using the transformations $\dot{\boldsymbol{r}}=\boldsymbol{R}(\phi,\theta,\psi)\boldsymbol{v}$ and $\dot{\boldsymbol{\Phi}}=\boldsymbol{T}(\phi,\theta,\psi)\boldsymbol{\omega}$. The transformation matrices are

$$\boldsymbol{R}(\phi,\theta,\psi)=\begin{bmatrix} \mathrm{c}\theta\mathrm{c}\psi & \mathrm{s}\theta\mathrm{s}\phi\mathrm{c}\psi-\mathrm{c}\phi\mathrm{s}\psi & \mathrm{s}\theta\mathrm{c}\phi\mathrm{c}\psi+\mathrm{s}\phi\mathrm{s}\psi \\ \mathrm{c}\theta\mathrm{s}\psi & \mathrm{s}\theta\mathrm{s}\phi\mathrm{s}\psi+\mathrm{c}\phi\mathrm{c}\psi & \mathrm{s}\theta\mathrm{c}\phi\mathrm{s}\psi-\mathrm{s}\phi\mathrm{c}\psi \\ -\mathrm{s}\theta & \mathrm{c}\theta\mathrm{s}\phi & \mathrm{c}\theta\mathrm{c}\phi \end{bmatrix}, \tag{2.3}$$

$$\boldsymbol{T}(\phi,\theta,\psi)=\frac{1}{\mathrm{c}\theta}\begin{bmatrix} \mathrm{c}\theta & \mathrm{s}\theta\mathrm{s}\phi & \mathrm{s}\theta\mathrm{c}\phi \\ 0 & \mathrm{c}\theta\mathrm{c}\phi & -\mathrm{c}\theta\mathrm{s}\phi \\ 0 & \mathrm{s}\phi & \mathrm{c}\phi \end{bmatrix}. \tag{2.4}$$

Note that c and s represent cosine and sine functions. See Craig [5] for details on the derivation of these matrices.

2.6.2 Quaternions

Euler angle representations such as the one above have several disadvantages. All Euler angle representations contain singularities. That is to say, the transformation $\dot{\boldsymbol{\Phi}} = \boldsymbol{T}(\phi,\theta,\psi)\boldsymbol{\omega}$ will be undefined for some angles (ϕ,θ,ψ). In the Z-Y-X Euler angle transformation matrix (2.4), a singularity occurs at $\theta = 90°$. As θ approaches 90°, cos(θ) approaches 0° and four elements of the transformation matrix approach infinity. This results in a loss of numerical accuracy of near 90°.

Another problem of Euler angles is in computation time. Every computation of matrices (2.3) and (2.4) requires the calculation of 6 trigonometric functions. Unless some form of lookup table is implemented, trigonometric functions are computationally slow to evaluate. Consequently Euler angles are slow to implement.

An alternate representation of the attitude that avoids the above problems can be obtained by use of the quaternion. A quaternion representation uses a fourth parameter to eliminate the singularity problems of Euler angles. It also eliminates the problem of evaluating trigonometric functions. Once the first quaternion is calculated, quaternions need only be updated with a sequence of multiplications and additions – no further evaluation of trigonometric functions is necessary.

A quaternion represents an orientation as follows. Consider a quaternion $\boldsymbol{q} = (\eta, \boldsymbol{\varepsilon}^{\mathrm{T}})^{\mathrm{T}}$ where η is a scalar and $\boldsymbol{\varepsilon}$ is a 3 element vector. Define unit vector $\boldsymbol{a}$ and angle α to be defined such that rotation of frame $\{I\}$ about $\boldsymbol{a}$ by angle α yields frame $\{B\}$. A quaternion representing such a rotation is

$$\begin{bmatrix}\eta\\ \varepsilon\end{bmatrix} = \begin{bmatrix}\cos(\alpha/2)\\ \boldsymbol{a}\sin(\alpha/2)\end{bmatrix}. \qquad \textbf{(2.5)}$$

Then, for example, a quaternion representing an orientation with frames $\{I\}$ and $\{B\}$ coincident (ie. angle $\alpha = 0°$) is $(1, 0, 0, 0)^{\mathrm{T}}$.

The relevant transformation equations are now $\dot{\boldsymbol{r}} = \boldsymbol{R}(\boldsymbol{q})\boldsymbol{v}$ and $\dot{\boldsymbol{q}} = \frac{1}{2}\boldsymbol{U}(\boldsymbol{q})\boldsymbol{\omega}$ where, from Fjellstad [6], the transformation matrices are given by

$$\begin{aligned}\boldsymbol{R}(\boldsymbol{q}) &= \boldsymbol{I}_{3\times3} + 2\eta\boldsymbol{S}(\boldsymbol{\varepsilon}) + 2\,[\boldsymbol{S}(\boldsymbol{\varepsilon})]^2\\ &= \begin{bmatrix} 1-2\varepsilon_2^2-2\varepsilon_3^2 & 2(\varepsilon_1\varepsilon_2-\eta\varepsilon_3) & 2(\varepsilon_1\varepsilon_3+\eta\varepsilon_2)\\ 2(\varepsilon_1\varepsilon_2+\eta\varepsilon_3) & 1-2\varepsilon_1^2-2\varepsilon_3^2 & 2(\varepsilon_2\varepsilon_3-\eta\varepsilon_1)\\ 2(\varepsilon_1\varepsilon_3-\eta\varepsilon_2) & 2(\varepsilon_2\varepsilon_3+\eta\varepsilon_1) & 1-2\varepsilon_1^2-2\varepsilon_2^2 \end{bmatrix}, \qquad \textbf{(2.6)}\end{aligned}$$

$$\boldsymbol{U}(\boldsymbol{q}) = \begin{bmatrix} -\boldsymbol{\varepsilon}^T \\ \eta \boldsymbol{I}_{3\times 3} + \boldsymbol{S}(\boldsymbol{\varepsilon}) \end{bmatrix} = \begin{bmatrix} -\varepsilon_1 & -\varepsilon_2 & -\varepsilon_3 \\ \eta & -\varepsilon_3 & \varepsilon_2 \\ \varepsilon_3 & \eta & -\varepsilon_1 \\ -\varepsilon_2 & \varepsilon_1 & \eta \end{bmatrix}. \tag{2.7}$$

The skew symmetric[1] matrix operator $\boldsymbol{S}(\cdot)$ is defined as

$$\boldsymbol{S}(\boldsymbol{a}) \triangleq \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}. \tag{2.8}$$

The significance of such a matrix operator is that $\boldsymbol{a} \times \boldsymbol{b} = \boldsymbol{S}(\boldsymbol{a})\boldsymbol{b}$. A vector cross product is reduced down to matrix multiplication.

Now as $\boldsymbol{R}(\boldsymbol{q})$ and $\boldsymbol{U}(\boldsymbol{q})$ have full rank, the kinematics equations contain no singularities. Note the elimination of trigonometric functions from these transformation matrices.

2.7 Newton's Law

Newton's second law can now be applied to the AUV. Summing moments about the centre of mass of the AUV, the general expressions for Newton's second law for a body of constant mass *m* are

$$\sum \boldsymbol{F} = m \, {}^{I}\frac{d}{dt}\left({}^{I}\boldsymbol{V}_B\right), \tag{2.9}$$

$$\sum \boldsymbol{\tau} = {}^{I}\frac{d}{dt}\left({}^{I}\boldsymbol{I}_B \cdot {}^{I}\boldsymbol{\Omega}_B\right). \tag{2.10}$$

The notation used courtesy of Craig [5]. The superscripted letter is the frame in which the particular quantity is taking respect to. For example ${}^{I}\boldsymbol{I}_B$ is the AUV inertia tensor about the origin of frame $\{B\}$ with respect to frame $\{I\}$. Similarly, ${}^{I}\boldsymbol{\Omega}_B$ is the angular velocity of frame $\{B\}$ in frame $\{I\}$.

Expanding the derivatives of equations (2.9) and (2.10) in terms of state variables,

$$\sum \boldsymbol{F} = m \, {}^{B}\frac{d}{dt}\left({}^{I}\boldsymbol{V}_B\right) + m \, {}^{I}\boldsymbol{\Omega}_B \times {}^{I}\boldsymbol{V}_B = m(\dot{\boldsymbol{v}} + \boldsymbol{\omega} \times \boldsymbol{v}), \tag{2.11}$$

[1] A skew symmetric matrix is a matrix with property $\boldsymbol{S} = -\boldsymbol{S}^{\mathrm{T}}$

$$\sum \boldsymbol{\tau} = {}^{B}\frac{d}{dt}\left({}^{I}\boldsymbol{I}_{B}\ {}^{I}\boldsymbol{\Omega}_{B}\right) + {}^{I}\boldsymbol{\Omega}_{B} \times \left({}^{I}\boldsymbol{I}_{B}\ {}^{I}\boldsymbol{\Omega}_{B}\right) = \boldsymbol{I}_{0}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (\boldsymbol{I}_{0}\boldsymbol{\omega}), \qquad \textbf{(2.12)}$$

where it has been noted that ${}^{I}\boldsymbol{V}_{B} = \boldsymbol{v}$, ${}^{I}\boldsymbol{\Omega}_{B} = \boldsymbol{\omega}$ and ${}^{I}\boldsymbol{I}_{B} = \boldsymbol{I}_{0}$. $\boldsymbol{I}_{0}$ is the inertia tensor of the AUV about its centre of mass. Recall that $\boldsymbol{v}$ and $\boldsymbol{\omega}$ are the linear and angular velocities expressed in frame $\{B\}$ coordinates. As the Kambara possesses symmetry about the XZ plane, several terms in the inertia tensor are zero. Also, since the Kambara is fairly symmetrical about the YZ plane, the $\boldsymbol{I}_{xz}$ product of inertia term can be neglected. The resultant expression for I_0 is hence

$$\boldsymbol{I}_{0} = \begin{bmatrix} I_x & 0 & -I_{xz} \\ 0 & I_y & 0 \\ -I_{xz} & 0 & I_z \end{bmatrix} \cong \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix}. \qquad \textbf{(2.13)}$$

Using the skew symmetric operator $\boldsymbol{S}(\cdot)$, equations (2.11) and (2.12) are more conveniently expressed as

$$\boldsymbol{M}_{0}\dot{\boldsymbol{v}} + \boldsymbol{C}_{0}\boldsymbol{v} = \boldsymbol{f}\ , \qquad \textbf{(2.14)}$$

where $\boldsymbol{M}_{0} = \begin{bmatrix} m\boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{I}_{0} \end{bmatrix}$, $\boldsymbol{C}_{0} = \begin{bmatrix} \boldsymbol{0}_{3\times3} & -m\boldsymbol{S}(\boldsymbol{v}) \\ -m\boldsymbol{S}(\boldsymbol{v}) & -\boldsymbol{S}(\boldsymbol{I}_{0}\boldsymbol{\omega}) \end{bmatrix}$, $\boldsymbol{v} = \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{\omega} \end{bmatrix}$, and $\boldsymbol{f} = \begin{bmatrix} \sum \boldsymbol{F} \\ \sum \boldsymbol{\tau} \end{bmatrix}$.

The $-m\boldsymbol{S}(\boldsymbol{v})\boldsymbol{v} = \boldsymbol{0}$ term is added to $\boldsymbol{C}_{0}$ for future convenience.

2.8 Acting Forces and Moments

The $\boldsymbol{f}$ term on the right hand side of equation (2.14) represents the net force and moment acting on the AUV. It can be considered as a 'generalised force', as it places both force and moment in the same vector. The main terms contributing to such a generalised net force come from several sources:

- the thrusters
- hydrodynamic added inertia
- hydrodynamic drag
- gravity and buoyancy

Equations for these generalised force terms are developed separately below.

2.8.1 Thrusters

Motion is imparted to the Kambara through 5 thrusters. As each thruster operates, it exerts a net force F_i and torque τ_i ($1 < i < 5$). A separate model is used to evaluate the values of these forces and torques. This model is currently in the process of development. At present, only the model gives the thruster forces F_i. The thruster torques τ_i will be determined as a future

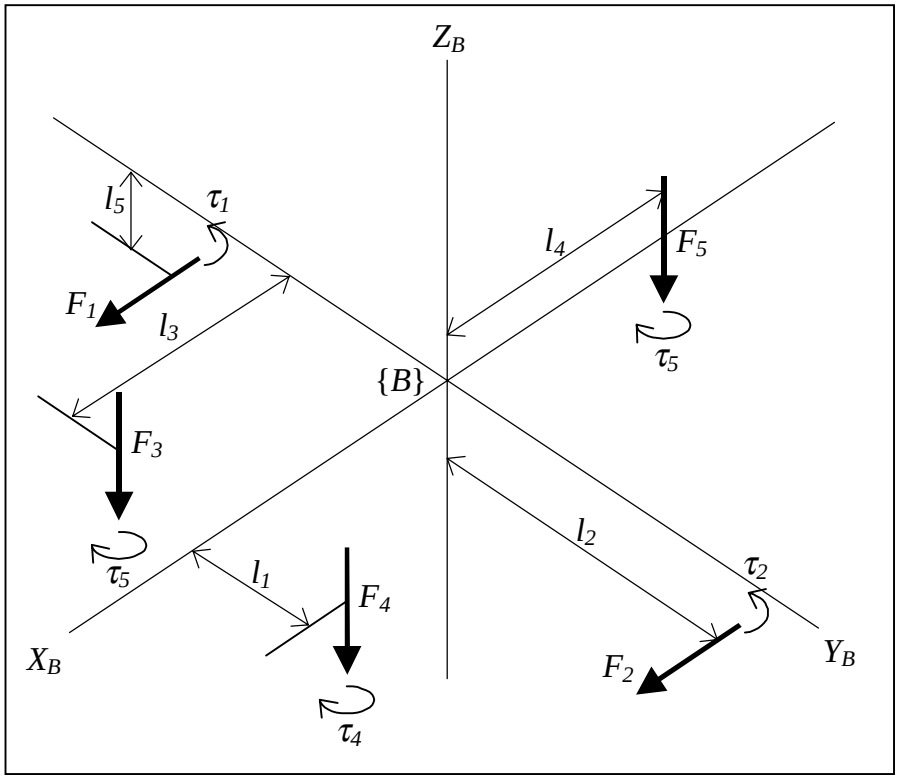


Figure 2.3: Positions of the lines of action of forces F_i and torques τ_i generated by the 5 thrusters in frame $\{B\}$. Note that the thruster positions are symmetrical about the $\boldsymbol{X_B}$-$\boldsymbol{Z_B}$ plane.

extension to the model. For the details of the thruster model at present, see Silpa-Anan [11]. A summary of this model is present in appendix 2. The lines of action of these forces and torques are specified through 5 lengths $l_1 \ldots l_5$. These lengths are defined in figure 2.3 below.

For the moment, the thruster torques τ_i have been ignored. From the positions of the thruster lines of action shown in figure 2.3, the generalised force vector exerted by the thrusters is given by $\boldsymbol{Lu}$ where

$$\boldsymbol{L} = \begin{bmatrix} \boldsymbol{L}_1 \\ \boldsymbol{L}_2 \end{bmatrix} = \left[\begin{array}{ccccc} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & -1 \\ \hline 0 & 0 & l_1 & -l_1 & 0 \\ -l_5 & -l_5 & l_3 & l_3 & -l_4 \\ l_2 & -l_2 & 0 & 0 & 0 \end{array}\right], \tag{2.15}$$

$$\boldsymbol{u} = (F_1, F_2, F_3, F_4, F_5)^{\mathrm{T}}. \tag{2.16}$$

$\boldsymbol{u}$ is the vector of control inputs for the process model.

2.8.2 Hydrodynamic Added Inertia

The movement of the AUV through a fluid causes motion to be imparted to the fluid. The pressure-induced forces and moments that result can be modelled as a virtual addition to the mass and inertia of the AUV. Such an added inertia is a function of fluid density and the body shape of the vehicle.

The contribution to the net force of this effect is $-\boldsymbol{M}_A\dot{\nu} - \boldsymbol{C}_A(\nu)\nu$ (see Fossen [7] for derivation details). The most general form of the added inertia matrix is the 6×6 matrix

$$\boldsymbol{M}_A = \begin{bmatrix} \boldsymbol{M}_{A11} & \boldsymbol{M}_{A12} \\ \boldsymbol{M}_{A21} & \boldsymbol{M}_{A22} \end{bmatrix} \tag{2.17}$$

where $\boldsymbol{M}_{Aij}$ are 3×3, $\boldsymbol{M}_A = \boldsymbol{M}_A^{\mathrm{T}}$, and the gyroscopic acceleration term $-\boldsymbol{C}_A(\nu)\nu$ has

$$\boldsymbol{C}_A(\nu) = \begin{bmatrix} \boldsymbol{0}_{3\times3} & -\boldsymbol{S}(\boldsymbol{M}_{A11}\boldsymbol{v} + \boldsymbol{M}_{A12}\omega) \\ -\boldsymbol{S}(\boldsymbol{M}_{A11}\boldsymbol{v} + \boldsymbol{M}_{A12}\omega) & -\boldsymbol{S}(\boldsymbol{M}_{A21}\boldsymbol{v} + \boldsymbol{M}_{A22}\omega) \end{bmatrix}. \tag{2.18}$$

As the shape of the Kambara is symmetrical about one plane and near symmetrical about the other two, the off-diagonal terms will be very small compared with the diagonal elements. Hence a reasonable approximation is make $\boldsymbol{M}_A$ diagonal,

$$\boldsymbol{M}_A \cong \begin{bmatrix} \boldsymbol{M}_{A11} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{M}_{A\,22} \end{bmatrix} \tag{2.19}$$

where $\boldsymbol{M}_{A11}$ and $\boldsymbol{M}_{A22}$ are now diagonal 3×3 matrices.

2.8.3 Hydrodynamic Drag

As the AUV moves through the water, it experiences a drag force from the surrounding fluid. The model of this drag force is dependent upon the nature of the fluid flow around the AUV. If the AUV moves through the fluid sufficiently slowly, the fluid flow is laminar. Under such conditions, the drag force could be modelled as proportional to velocity. However Toh [14] has shown that at reasonable operating speeds (say 0.5m/s) fluid flow is turbulent, causing the drag force to possess an approximately quadratic dependence on velocity. The generalised drag force term is hence approximated by $-\text{Diag}|\boldsymbol{v}|\boldsymbol{D}\boldsymbol{v}$ where *Diag* converts a vector into a diagonal matrix and $\boldsymbol{D}$ is a positive definite matrix of drag coefficients. The absolute value in this expression and positive definite nature of $\boldsymbol{D}$ reflects the dissipative nature of drag forces.

In general, movement in some directions will create drag forces in orthogonal directions. For example, the lift force acting on an airfoil acts perpendicular to the direction of movement. As a result, one expects $\boldsymbol{D}$ contain nonzero off-diagonal terms. In spite of this, a rough approximation is to make $\boldsymbol{D}$ diagonal,

$$\boldsymbol{D} \cong \begin{bmatrix} \boldsymbol{D}_{11} & \boldsymbol{0}_{3\times 3} \\ \boldsymbol{0}_{3\times 3} & \boldsymbol{D}_{22} \end{bmatrix} \tag{2.20}$$

where $\boldsymbol{D}_{11}$ and $\boldsymbol{D}_{22}$ are diagonal 3×3 matrices.

2.8.4 Gravity and Buoyancy

Gravity and buoyancy both apply forces to the AUV. If m is the mass of the AUV and g is acceleration due to gravity, gravity produces a force at the centre of mass of the AUV pointing downward in the inertial reference frame. This force is given by $\boldsymbol{f}_g(\boldsymbol{q}) = mg\,{}^{B}\boldsymbol{Z}_{I}$ where ${}^{B}\boldsymbol{Z}_{I}$ is the $\boldsymbol{Z}$ unit vector of frame $\{I\}$ represented in frame $\{B\}$ coordinates,

$${}^{B}\boldsymbol{Z}_{I} = {}^{I}\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \boldsymbol{R}^{T}(\boldsymbol{q})\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = {}^{B}\begin{bmatrix} 2(\varepsilon_1\varepsilon_3 - \eta\varepsilon_2) \\ 2(\varepsilon_2\varepsilon_3 + \eta\varepsilon_1) \\ 1 - 2\varepsilon_1^2 - 2\varepsilon_2^2 \end{bmatrix}. \tag{2.21}$$

Note the application of the inverse of the rotation matrix $\boldsymbol{R}^{-1}(\boldsymbol{q})=\boldsymbol{R}^{\mathrm{T}}(\boldsymbol{q})$ to convert from body-fixed to inertial coordinates.

Archimedes principle states that when a body is immersed in a fluid, it experiences a force equal to the weight of fluid displaced by that body. If the volume of fluid displaced is V and density of fluid is ρ, the buoyancy force is given by $\boldsymbol{f}_b(\boldsymbol{q}) = -\rho g V\,{}^{B}\boldsymbol{Z}_{I}$. This force acts at the centre of buoyancy (or centroid) of the AUV.

The centre of mass and centre of buoyancy are in general not the same point. The centre of mass is dependent on the mass distribution within the AUV, while the centre of buoyancy is not. In fact, in the Kambara, the 60kg battery pack sits in the bottom compartment, making the centre of gravity about 100mm below the centre of buoyancy. The buoyancy force will hence generate a righting moment $\boldsymbol{r_b}\times\boldsymbol{f_b}(\boldsymbol{q})$ about the centre of mass where $\boldsymbol{r_b}$ is the position of the centre of buoyancy in frame $\{B\}$.

The net contribution from gravity and buoyancy is summarised in the generalised force term $\boldsymbol{g}(\boldsymbol{q})$ where

$$\boldsymbol{g}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{f}_b + \boldsymbol{f}_g \\ \boldsymbol{r}_b \times \boldsymbol{f}_b \end{bmatrix} = \begin{bmatrix} g(m-\rho V)^B\boldsymbol{Z}_I \\ -\rho g V \boldsymbol{S}(\boldsymbol{r}_b)^B\boldsymbol{Z}_I \end{bmatrix}. \tag{2.22}$$

2.8.5 Final Equations

Substituting the terms from the thrusters, hydrodynamic added mass, hydrodynamic drag, gravity and buoyancy from sections 2.8.1 - 2.8.4, equation (2.14) becomes

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} = \boldsymbol{L}\boldsymbol{u} - \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} - Diag|\boldsymbol{\nu}|\boldsymbol{D}\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{q}). \tag{2.23}$$

where in the case that $\boldsymbol{M}_A$ and $\boldsymbol{D}$ are approximated as diagonal

$$\boldsymbol{M} = \boldsymbol{M}_0 + \boldsymbol{M}_A = \begin{bmatrix} \boldsymbol{M}_{11} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{M}_{22} \end{bmatrix}, \tag{2.24}$$

$$\boldsymbol{C}(\boldsymbol{\nu}) = \boldsymbol{C}_0(\boldsymbol{\nu}) + \boldsymbol{C}_A(\boldsymbol{\nu}) = \begin{bmatrix} \boldsymbol{0}_{3\times3} & -\boldsymbol{S}(\boldsymbol{M}_{11}\boldsymbol{\nu}) \\ -\boldsymbol{S}(\boldsymbol{M}_{11}\boldsymbol{\nu}) & -\boldsymbol{S}(\boldsymbol{M}_{22}\boldsymbol{\omega}) \end{bmatrix}. \tag{2.25}$$

Finally, equation (2.23) can be solved for state vector variables $\boldsymbol{\nu}$ and $\boldsymbol{\omega}$. The result is

$$\dot{\boldsymbol{\nu}} = \boldsymbol{M}_{11}^{-1}\left\{\boldsymbol{L_1}\boldsymbol{u} + \boldsymbol{S}(\boldsymbol{M}_{11}\boldsymbol{\nu})\boldsymbol{\omega} - Diag(\boldsymbol{D}_{11}|\boldsymbol{\nu}|)\boldsymbol{\nu} + g(m-\boldsymbol{\rho}V)^B\boldsymbol{Z}_I\right\}, \tag{2.26}$$

$$\dot{\omega} = \boldsymbol{M}_{22}^{-1}\left\{\boldsymbol{L}_2\boldsymbol{u} + \boldsymbol{S}(\boldsymbol{M}_{11}\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{S}(\boldsymbol{M}_{22}\omega)\omega - Diag(\boldsymbol{D}_{22}|\omega|)\omega - \rho g V \boldsymbol{S}(\boldsymbol{r_B})^B\boldsymbol{Z}_I\right\} \tag{2.27}$$

Recall from section 2.6.2 the equations relating the derivatives of the other state variables are

$$\dot{\boldsymbol{r}} = \boldsymbol{R}(\boldsymbol{q})\boldsymbol{\nu}, \tag{2.28}$$

$$\dot{\boldsymbol{q}} = \tfrac{1}{2}\boldsymbol{U}(\boldsymbol{q})\boldsymbol{\omega}, \tag{2.29}$$

$$\dot{T} = 0, \tag{2.30}$$

where $\boldsymbol{R}(\boldsymbol{q})$ and $\boldsymbol{U}(\boldsymbol{q})$ are defined in (2.6) and (2.7). Equations (2.26) - (2.30) summarise the continuous time equations describing the motion of the Kambara.

2.9 Discretisation

Before equations (2.26) - (2.30) can be used in the (digital) Kalman filtering algorithm, they must be discretised. The first step in the discretisation process is the integration to convert the differential equations into integral equations. The integrals must then be evaluated. For linear systems, these integrals can be evaluated exactly through calculation of the state transition matrix. However, this system is highly nonlinear with no analytical solution. Instead, the backward Euler approximation is made:

$$\int_{t_k}^{t_{k+1}} \boldsymbol{f}(t)dt \cong (t_{k+1} - t_k)\,\boldsymbol{f}(t_k) \qquad \textbf{(2.31)}$$

Better approximation techniques are available (eg. a trapezoidal integration technique could be used) but these techniques significantly increase the complexity of the discretised equations and will result in greater demands on processing power.

Integration between $t=kT_S$ and $(k+1)T_S$ and application of the backward Euler approximation yields:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + T_S\dot{\boldsymbol{x}}_k \qquad \textbf{(2.32)}$$

where T_S is the sampling period used. The smaller T_S is, the better the Euler approximation becomes.

CHAPTER 3 - THE SENSOR SUITE

3.1 Introduction

Before jumping into the sensor model in chapter 4, the sensors themselves and their implementation into the Kambara will be summarised. This includes both the physical mounting onto the AUV and the distribution of power to the sensors. But firstly the sensors themselves and their basic properties will be described.

3.2 Sensor Descriptions

The sensors selected for the inertial navigation system (INS) are

- a Summit Instruments triaxial accelerometer,
- a SensorTechnics depth sensor,
- a Systron Donner MotionPak and
- a Precision Navigation TCM2.

The model numbers of these sensors are summarised in table 3.1 below. The TCM2 communicates with the outside world through RS-232 signals. It is to be interfaced with a serial port module. All other sensors produce analog output signals. These signals are read through an analog-to-digital converter (ADC) module. The sensor output interfacing is discussed in chapter 6.

Table 3.1: Sensor names and models.

Sensor Name	Manufacturer	Model No.	Output type
Accelerometer	Summit Instruments	34103A	Analog
Depth	SensorTechnics	PTE2005G1A	Analog
MotionPak	Systron Donner	MP-GDDDQBBB-100	Analog
TCM2	Precison Navigation	TCM2-50	RS-232*

* The TCM2 also outputs in analog format, but only after passing the digital signal through an internal 8 bit ADC.

Each sensor module is discussed briefly in the separate sections below. In each section, sensitivity and bias data is listed. A more accurate and complete listing of values are listed in appendix 3. The power distribution to these sensors is described in section 3.4.

3.2.1 Summit Instruments Trixial Accelerometer

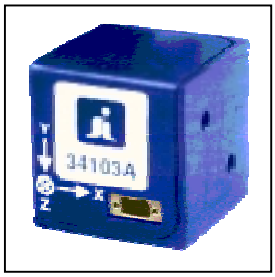


Figure 3.1: Triaxial accelerometer

A triaxial accelerometer measures acceleration in three mutually perpendicular directions. It does by using a single accelerometer unit measuring the acceleration along each axial direction. The principle of operation is measurement of the force-induced displacement acting on a mass along the direction of interest. Because gravity always exerts a force on any mass, acceleration due to gravity will be seen superimposed onto acceleration measurements. Keeping this in mind, an accelerometer with a full scale range ±1.5g was chosen for use in the Kambara. Important accelerometer characteristics are shown in table 3.2 below.

Table 3.2: Summit Instruments accelerometer characteristics.

Full scale range	±1.5g
Output voltage swing	0.25 ... 4.75V
Sensitivity	
At 25°C	1.43V/g
Drift T_{MIN} to T_{MAX}	±0.5V/g
Bias	
At 25°C	2.5V
Drift T_{MIN} to T_{MAX}	±0.2V/g
Linearity	0.2% FSR
Cutoff frequency	> 7Hz
Noise density	$< 1\,\mathrm{mg}/\sqrt{\mathrm{Hz}}$
Operating range	–40°C … 80°C

Table 3.3: Sensor Technics depth sensor characteristics.

Full scale range	0 … 3.5m*
Output voltage swing	1 … 6V*
Sensitivity (25°C)	1.42V/m
Bias	1.0V
Drift T_{MIN} to T_{MAX} (incl. both bias & span)	±2.5% FSR
Linearity	< ±0.5% FSR
Noise density	±0.04% FSR
Cutoff frequency	1kHz
Operating range	0°C … 70°C

* for water

3.2.2 SensorTechnics Depth Sensor

Figure 3.2: Depth sensor

Important sensor characteristics are shown in table 3.3. This sensor actually measures pressure. As a result, it will possess different sensitivities in media with differing densities. This is of little concern however, as water density stays relatively constant with temperature and water composition, only varying by 0.4% over a 0–30°C temperature range. Sensitivity of the depth sensor stated in table 3.3 consequently will increase by up to 0.4% under minimum temperature conditions. Also, the zero point is slightly dependent on atmospheric pressure. As atmospheric pressure typically varies over a 0.5kPa range, offset bias will vary over a 5mm or 7mV range

A major concern with the current sensor is the 3.5 metre full scale range. The pressure sensor starts taking damage above 15psi (10m depth). For final implementation, this sensor will have to be replaced by one with a larger full scale range.

3.2.3 Systron Donner MotionPak

The MotionPak is a not a single sensor but an integrated sensor package. It contains an on-board triaxial accelerometer, triaxial rate gyroscope, and temperature sensor. These shall all be used as part of the state estimation system. The temperature sensor will be used to correct for the temperature dependence of the other sensors.

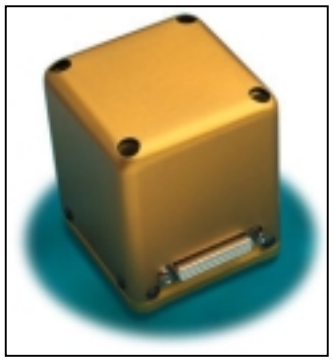


Figure 3.3: Motionpak

The relevant characteristics of the accelerometer and rate gyroscope are shown in table 3.4. The AD590 temperature sensor on-board has a sensitivity of 1μA/°C. A complete set of sensor sensitivity and offset biases for all sensors is listed in appendix 3.

Table 3.4: MotionPak accelerometer and rate gyroscope characteristics.

	Gyroscope	**Accelerometer**
Full scale range	±2g	±2g
Output voltage swing	±2.5V	±7.5V
Sensitivity		
At 22°C	25mV/°/s	3.75V/g
Temperature	< 0.03%/°C	< 0.03%/°C
Bias		
At 22°C	0°/s	0g
Drift 22°C to T_{MAX}	< 3°/s	< 100μg/°C
Linearity	< 0.05% FSR	< 10μg/g^2
Bandwidth (-90°)	> 60Hz	> 300Hz
Noise density	$< 0.01° / \text{sec} / \sqrt{\text{Hz}}$	< 7.0 mVRMS
Operating range	–40°C … 80°C	

Before feeding the MotionPak accelerometer and gyroscope outputs into an analog to digital converter, they require low pass filtering. The results of experimenting with simple low pass filters are shown in chapter 7.

3.2.4 TCM2 Compass / Biaxial Inclinometer

The TCM2 provides a measure of attitude up to an angle of inclination of 50°. It contains on-board a triaxial magnetometer, biaxial inclinometer, and temperature sensor. The magnetometers feed out magnetic field strength data used to orient the TCM2 with respect to the magnetic field of the earth. An on-board Kalman uses this magnetometer and inclinometer to estimate heading. Key TCM2 characteristics are summarised in table 3.5 below.

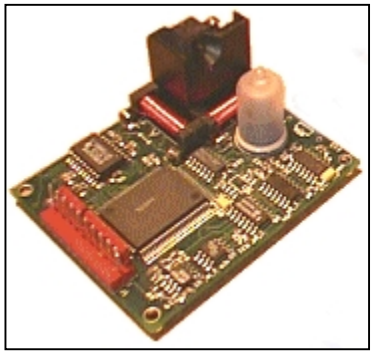


Figure 3.4: TCM2 Compass

A major influence on the accuracy on the compass heading readings is the background magnetic field. All compasses can perform well in an environment containing only the earth's magnetic field. However in practical situations the compass module will be mounted into environments containing large sources of magnetic fields such as ferrous metal components, electric currents and the permanent magnets of electric motors.

Table 3.5: TCM2-50 compass / inclinometer characteristics.

	Heading	**Tilt**	**Magnetometer**
Full scale range	0 … 360°	±50°	±80μT
Accuracy	±1.5°*	±0.4°	±0.2μT
Resolution	0.1°	0.3°	0.01μT
Repeatability	±0.3°	±0.3°	±0.2μT
Operating range	−20°C … 70°C		

*when tilted; heading accuracy is ±1.0° when level

Because of interfering magnetic fields in practical situations, the TCM2 compass module possesses a user calibration procedure for compensation. In particular, the compass module is able to compensate for *hard iron* effects. Hard iron effects are due to fields that are fixed in the frame of the compass module. As a result, the field due to the electrical circuitry in the upper enclosure of the Kambara can be corrected for.

Soft iron effects cannot be corrected unfortunately. These effects are created by the amplification of magnetic fields by highly permeable materials. In particular ferrous metals are a major concern. Fortunately, the only major sources of ferrous metals on the Kambara are the thrusters. As soft iron effects drop off with distance by an inverse square rule, soft iron effects will not be a major concern.

As mentioned above, the TCM2 contains a temperature sensor on-board. For the purposes of state estimation, this sensor is redundant on the TCM2. The TCM1 required a temperature

sensor for temperature compensation by the user, but the TCM2 compensates for temperature automatically.

Several limitations not mentioned in table 3.5 pertain to the presence of the accelerometer fluid. Firstly, sudden changes in heading will cause the inclinometer fluid to slosh. Settling time of such sloshing is 300msec. Also, acceleration the inclinometer fluid will cause it to tilt, decreasing compass accuracy. For example, an acceleration of $0.84ms^{-2}$ will produce a tilt error of 1° and a compass 1 – 8° error in heading, depending upon the strength of the earth's magnetic field at the position of measurement.

3.3 Physical Sensor Mounting

Eventually, the sensors will be fixed into the enclosures of the Kambara. Before this occurs, dynamic testing of the MotionPak and TCM2 compass modules will be accomplished. To accomplish this testing, the sensors will be attached to a 7 degree of freedom manipulator arm called the Whole Arm Manipulator (WAM arm) so that their position can be controlled and recorded. Suitable mounting was to be designed for both of these situations.

The depth sensor could not be included as part of the sensor suite. As it measures pressure, in the Kambara, it must be contact with the water outside the enclosure. As a consequence, a previous design decision dictated the depth sensor to be mounted into the front-end cap of the lower enclosure. Mounting onto the WAM arm would still possess some merit, as the performance of the depth sensor may degrade under acceleration or vibration. However, as appropriate sensor mounting is already in existence, the inconvenient of designing an additional mounting was deemed excessive.

Important considerations for the design of the sensor mounting were:

- the secure mounting of accelerometer, TCM2 and MotionPak,
- minimisation of sensor vibration so as to minimise sensor noise,
- the rigidity of the mounting under the 900gm load of the MotionPak,
- a suitable screw thread attachment for mounting onto the wrist of the WAM arm and
- choice of a non-ferrous mounting material.

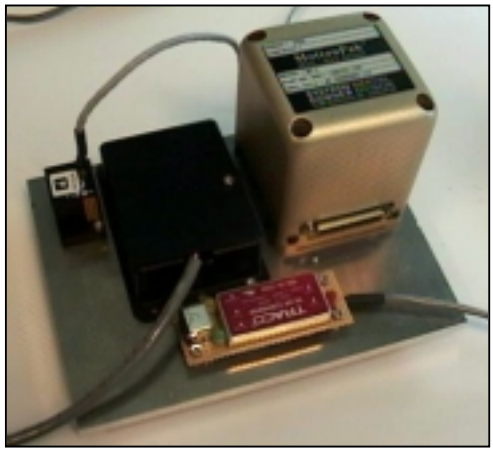


Figure 3.5: Sensors and DC-DC converter mounted fixed onto mounting plate.

Rigidity was important to prevent sagging in the mounting, which in turn could cause the sensors to change their orientations slightly without notice. As

mentioned in section 3.2.4, choice of non-ferrous mounting material was necessary to prevent soft iron effects from interfering with the TCM2 compass module.

The final design involved the mounting of the three sensors on a 200×150×8mm aluminium plate, as shown in figure 3.5. Figure 3.8 overleaf shows an AutoCAD drawing of this plate. The plate is then mounted into the aluminium 'cage' shown in figure 3.6 and made secure by use of six M3 screws. This cage then slides through a pair of aluminium rails into the Kambara.

Appendix 4 shows the final design drawing of the adaptor plate used to attach the sensor suite onto the wrist of the WAM arm. The adaptor plate was also machined out of aluminium.

A major problem with the current sensor suite design is its weight. The combined weight of the sensor suite is almost 2kg. It was later found that the WAM arm could not support the weight of the sensor suite without risk of damaging the wrist. This doesn't mean that the WAM arm mounting was useless. It will still prove useful in logging sensor position. However the WAM arm cannot be used for position control.

Figure 3.6: Positioning of the sensor suite in the hardware 'cage'.

3.4 Power Distribution System

The Kambara contains on-board a power distribution system to supply the necessary power to the sensors. Four 12V batteries are present in the lower enclosure of the Kambara for powering of all the electrical equipment and the thrusters. These batteries are used to supply 5V and 12V power lines to the sensors.

It will be shown in chapter 7 that the MotionPak triaxial accelerometer performs significantly better than the Summit Instruments accelerometer. Furthermore, it will be seen that little is gained by using both sensor units together. As a consequence, though the Summit Instruments accelerometer was included as part of the sensor suite, it will not be wired in the physical implementation of the Kambara.

The sensor power requirements are shown in table 3.6 below. Note that the MotionPak requires ±15V supply voltages. A suitable switch-mode power supply (shown in figure 3.7) has been chosen to produce these supply voltages from the 12V battery sources. The total power usage from all sensors (excluding the SI accelerometer) is 7 Watts.

Figure 3.7: DC-DC converter for MotionPak sensor.

Table 3.6: Sensor power requirements.

Sensor name	Supply voltage (V)	Supply current (mA)	Kambara power usage (W)	Output type
Accelerometer	5 ±5%	25	0.125	Analog
Depth	12 … 20	20	0.24	Analog
MotionPak	±15	430	6.5	Analog
TCM2	6 … 25*	20	0.25	RS-232**

* The TCM2 possesses a power input 5V±5% for a regulated power supply

** The TCM2 also outputs in analog format, but only after passing the digital signal through an internal 8 bit analog-to-digital converter.

A problem of using switch-mode power supplies such as the DC-DC converter mentioned above is the introduction of noise. Switch-mode power supplies are notoriously noisy. In particular, there is concern is that the noise it introduced may have detrimental effects on the operation of the MotionPak. However, experimentation with the MotionPak with and without the DC-DC converter has shown this not to be a problem. The noise standard deviations observed were approximately equal.

CHAPTER 4 - SENSOR MODELLING

4.1 Introduction

The other major component of the Kalman filtering algorithm besides the process model is the *measurement equation*. The *measurement equation* is basically a model of the sensors. It describes the expected sensor outputs in terms of the current state estimate and the control inputs. A model of the noise outputs of the sensors is an important component of this equation. Mathematically, the measurement equation can be expressed in form

$$\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{w}_k), \tag{4.1}$$

where $\boldsymbol{z}_k$ is the vector of sensor outputs or measurement vector, $\boldsymbol{x}_k$ is the current state vector, $\boldsymbol{u}_k$ the current control input (thruster forces) and $\boldsymbol{w}_k$ is the vector of noise processes.

As in chapter 2, this chapter focuses on developing the deterministic component of the sensor model. That is to say, it develops an equation of form $\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{u}_k, 0)$. Suitable models of the noise processes are explored in chapter 6 where actual data from the sensors is analysed.

The measurement vector $\boldsymbol{z}_k$ contains the outputs from all the sensors being sampled at a given time. It is hence given by:

$$\boldsymbol{z} = (\boldsymbol{a}^{\mathrm{T}}, \boldsymbol{\Omega}^{\mathrm{T}}, R, P, \psi, d, T_m)^{\mathrm{T}}. \tag{4.2}$$

The components of $\boldsymbol{z}$ are described in table 4.1. As the two vectors are 3×1 and the other parameters are scalars, equation (4.1) will be a system of 11 equations. These equations are to be derived in the sections to follow.

Symbol	Sensor quantity	Sensor name
$\boldsymbol{a}$	Acceleration	MotionPak triaxial accelerometer
$\boldsymbol{\Omega}$	Angular velocity	MotionPak rate gyroscope
P	Inclinometer roll	TCM2 biaxial inclinometer
R	Inclinometer pitch	TCM2 biaxial inclinometer
ψ	Compass yaw	TCM2 compass
d	Depth	SensorTechnics pressure sensor
T_m	Temperature	MotionPak temperature sensor

Table 4.1: Description of measurement vector variables.

These equations contain several common elements. All sensors with analog outputs will contain a sensitivity term $\boldsymbol{S_i}$ and zero offset bias $\boldsymbol{b_i}$. Any temperature dependencies in offset

bias are taken up in $\boldsymbol{k}_i T$ terms. Ie. the offset bias is approximated as a linear function of temperature $\boldsymbol{b}_i + \boldsymbol{k}_i T$.

In general $\boldsymbol{S}$ is a matrix[2] and $\boldsymbol{b}$ & $\boldsymbol{k}$ are vectors. This matrix-vector representation proves useful for sensors that output vector quantities (such as acceleration). For example, any difference in the alignment of the accelerometer axes from frame $\{B\}$ can be accounted for in the $\boldsymbol{S}$ matrix.

4.2 Trixial Accelerometer

A triaxial accelerometer measures acceleration in three mutually perpendicular directions. On Kambara, the three axes are approximately aligned with the body-fixed frame $\{B\}$.

Recall from chapter 3 that gravity is superimposed on the sensor output. This is unfortunate, as the gravity term must be subtracted from the accelerometer output before a true indication of acceleration can be given. The accelerometer readings can hence be no more accurate than the attitude estimates coming out of the Kalman filter, because the attitude needs to be accurately known for subtracting gravity completely. In fact they will be very sensitive to errors in attitude. A small error in attitude translates to a large error in acceleration.

Another influencing factor on the output of the accelerometer is temperature. Testing of the accelerometer has shown that the zero point offset is slightly temperature dependent. As chapter 7 will show, this zero point offset can successfully be modelled as a linear function of temperature and that different axes of the same sensor have significantly different temperature coefficients.

Although the axes of the accelerometer are aligned with those of $\{B\}$, the accelerometer will be offset from the centre of mass. Calling the offset vector $\boldsymbol{r}_a$, the acceleration of the sensor ${}^I\boldsymbol{a}_a$ will be slightly different from that at the centre of mass ${}^I\boldsymbol{a}_B$,

$$\begin{aligned} {}^I\boldsymbol{a}_a &= {}^I\boldsymbol{a}_B + {}^B\frac{d}{dt}\left({}^I\boldsymbol{\Omega}_B\right)\times\boldsymbol{r}_a + {}^I\boldsymbol{\Omega}_B\times\left({}^I\boldsymbol{\Omega}_B\times\boldsymbol{r}_a\right) \\ &= \dot{\boldsymbol{v}} - \boldsymbol{S}(\boldsymbol{r}_a)\dot{\boldsymbol{\omega}} + \boldsymbol{S}^2(\boldsymbol{\omega})\boldsymbol{r}_a \end{aligned} \qquad \textbf{(4.3)}$$

Expressions for $\dot{\boldsymbol{v}}$ and $\dot{\omega}$ were derived in chapter 2. In terms of these expressions, the equations relating the accelerometer output readings $\boldsymbol{a}$ are

$$\boldsymbol{a} = \boldsymbol{S}_a\left[\dot{\boldsymbol{v}} - \boldsymbol{S}(\boldsymbol{r}_a)\dot{\boldsymbol{\omega}} + \boldsymbol{S}^2(\boldsymbol{\omega})\boldsymbol{r}_a + g\,{}^B\boldsymbol{Z}_I\right] + \boldsymbol{k}_a T + \boldsymbol{b}_a \qquad \textbf{(4.4)}$$

[2] The sensitivity matrix $\boldsymbol{S}_i$ is not to be confused with the skew symmetric operator $\boldsymbol{S}(\cdot)$

where $\boldsymbol{b_a}$ are the zero offset biases and $\boldsymbol{k_a}$ are temperature drift rates. An expression for ${}^{B}\boldsymbol{Z_I}$ was given in chapter 2.

4.3 Compass / Biaxial Inclinometer

The TCM2 compass module possesses a biaxial inclinometer and a triaxial magnetometer. The purpose of this module is measurement of attitude. Two sets of equations have been derived for the modelling of this sensor module. The first set (section 4.3.1) provides a basic model that is most accurate when the TCM2 undergoes no acceleration. The second set of equations (section 4.3.2) removes this assumption and enhances filter performance. However, it also results in a more complicated model and the addition of two variables to the state vector.

4.3.1 Basic Compass Model

The biaxial inclinometer is a fluid-filled tilt sensor used to measure the orientation of the compass with respect to gravity. Provided the compass module is not accelerating, the tilt sensors measure the angles between the compass and the gravity vector. These angles are called the pitch angle *P* and roll angle *R*. They are **not** to be confused with the pitch and roll Euler angles θ and ϕ.

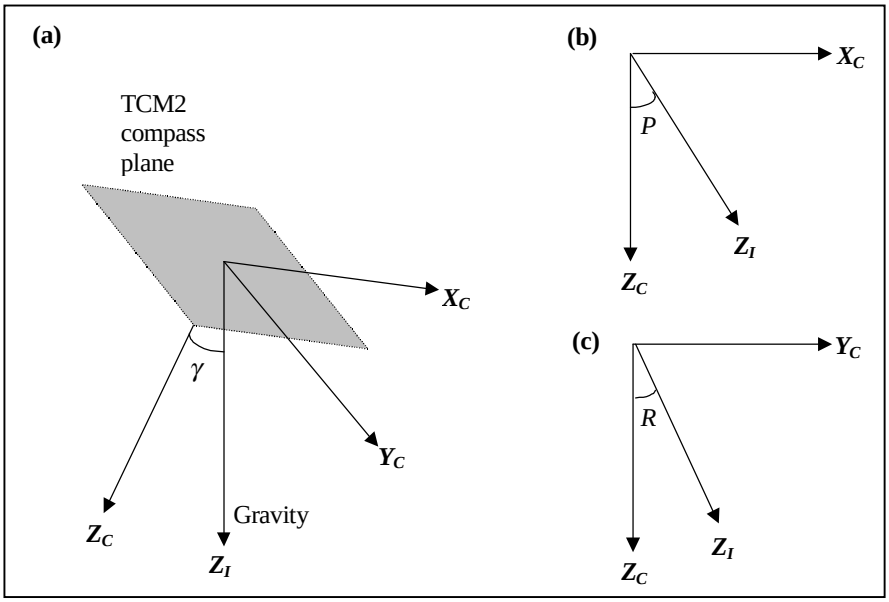


Figure 4.1: Definition of TCM2 pitch angle *P*, roll angle *R* and inclination angle γ. The TCM2 compass frame $\{C\}$ is shown in (a). $\boldsymbol{Z_C}$ is perpendicular to the plane of the TCM2 compass. (b) and (c) show angles *P* and *R*.

The pitch and roll angles are defined in figure 4.1. In figure 4.1(a), the reference frame of the compass $\{C\}$ has been defined. In terms of defined reference frames, the pitch angle *P* is the angle between the gravity vector and $\boldsymbol{Z_C}$ measured in the X_C-Z_C plane (see figure 4.1(b).

Similarly, roll angle R is the angle between the gravity vector and $\boldsymbol{Z}_C$ measured in the $\boldsymbol{Y}_C$-$\boldsymbol{Z}_C$ plane. Assuming $\{C\}$ is aligned with $\{B\}$, the mathematical expressions for P and R are

$$P = \text{Atan2}\left(\boldsymbol{Z}_I \cdot \boldsymbol{X}_C,\ \boldsymbol{Z}_I \cdot \boldsymbol{Z}_C\right) = \text{Atan2}\left(\varepsilon_1\varepsilon_3 - \eta\varepsilon_2,\ \tfrac{1}{2} - \varepsilon_1^2 - \varepsilon_2^2\right), \quad \textbf{(4.5)}$$

$$R = \text{Atan2}\left(\boldsymbol{Z}_I \cdot \boldsymbol{Y}_C,\ \boldsymbol{Z}_I \cdot \boldsymbol{Z}_C\right) = \text{Atan2}\left(\varepsilon_2\varepsilon_3 + \eta\varepsilon_1,\ \tfrac{1}{2} - \varepsilon_1^2 - \varepsilon_2^2\right), \quad \textbf{(4.6)}$$

where Atan2 is a two-argument arc tangent function[3].

If $\{B\}$ and $\{C\}$ are not aligned, the rotation matrix ${}^{B}_{C}\boldsymbol{R}$ between these reference frames must be taken into account. However in the case of Kambara, frames $\{B\}$ and $\{C\}$ do align, at least approximately. (No attempt has been made at precision alignment.)

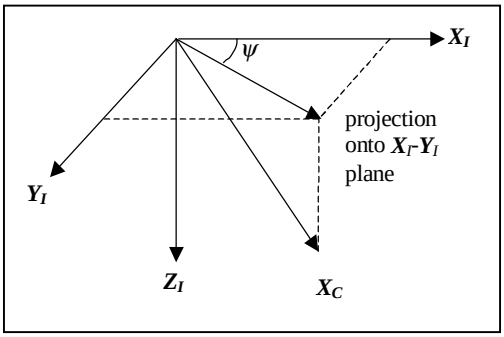


Figure 4.2: Definition of compass bearing angle ψ.

The TCM2 also outputs the compass bearing angle ψ. In terms of state variables, the compass bearing angle ψ is given by the angle between $\boldsymbol{X}_I$ and the projection of $\boldsymbol{X}_C$ onto the $\boldsymbol{X}_I$-$\boldsymbol{Y}_I$ plane. From figure 4.2,

$$\begin{aligned}\psi &= \text{Atan2}\left(\boldsymbol{X}_C \cdot \boldsymbol{Y}_I,\ \boldsymbol{X}_C \cdot \boldsymbol{X}_I\right) \quad \textbf{(4.7)}\\ &= \text{Atan2}\left(\varepsilon_1\varepsilon_2 + \eta\varepsilon_3,\ \tfrac{1}{2} - \varepsilon_1^2 - \varepsilon_2^2\right).\end{aligned}$$

The final important angle of the TCM2 compass is the tilt angle γ defined in figure 4.1. A major limitation on the TCM2 is that attitude cannot be measured above a tilt angle of 50°. It is important to obtain an expression for tilt angle. In terms of measurement variables P and R, tilt angle is given by

$$\gamma = \cos^{-1}\left(\frac{1}{\sqrt{1 + \tan^2 P + \tan^2 R}}\right). \quad \textbf{(4.8)}$$

4.3.2 Improved Compass Model

The TCM2 sensor model derived in the previous section possesses two drawbacks. The first drawback is associated with the noise properties of the compass bearing readings of the TCM2. On the TCM2, compass bearing estimates are obtained by passing the inclinometer

[3] Atan2(y,x) computes $\tan^{-1}(y/x)$ but uses the signs of both x and y to determine the quadrant in which the resulting angle lies.

and magnetometer sensor outputs through it's own Kalman filter. This filtering colours the noise in the bearing estimates and correlates them with inclinometer measurement noise. A more complicated noise model is required.

The larger problem, however, has to do with the exclusion of acceleration effects. The pitch and yaw angle equations do not account for these effects. These erroneous angle measurements propagate through to the TCM2 Kalman filter, producing even larger estimation errors in compass bearing. Such problems can be avoided by compensating inclinometer equations and filtering the triaxial magnetometer readings directly.

In the case of non-zero acceleration, the tilt sensor readings will no longer be in reference to the gravitational vector, but the net acceleration at the compass including gravity. Similar to equation 4.2, the net acceleration at the compass is given by

$$ {}^{I}\boldsymbol{a}_{C} = \dot{\boldsymbol{v}} - \boldsymbol{S}(\boldsymbol{r}_{c})\dot{\boldsymbol{\omega}} + \boldsymbol{S}^{2}(\boldsymbol{\omega})\boldsymbol{r}_{c}, \qquad \textbf{(4.9)} $$

where $\boldsymbol{r_c}$ is the position of the TCM2 inclinometer. The new equations for pitch and roll angles become

$$ P = \text{Atan2}\left({}^{I}\boldsymbol{a}_{C} \cdot \boldsymbol{X}_{C},\ {}^{I}\boldsymbol{a}_{C} \cdot \boldsymbol{Z}_{C} \right), \qquad \textbf{(4.10)} $$

$$ R = \text{Atan2}\left({}^{I}\boldsymbol{a}_{C} \cdot \boldsymbol{Y}_{C},\ {}^{I}\boldsymbol{a}_{C} \cdot \boldsymbol{Z}_{C} \right). \qquad \textbf{(4.11)} $$

The expression for pitch angle γ in equation (4.8) is still valid, as it is purely based on sensor readings. However as mentioned previously, the compass bearing readings coming from the TCM2 compass are not accurate in an accelerating system. An equation for ψ is no longer useful.

A means of incorporating magnetometer readings directly into the Kalman filter is suggested in Smith [12]. This paper suggests modifying the measurement vector $\boldsymbol{z}$ to replace the compass bearing measurement ψ with the flux density $\boldsymbol{B}$ measured by the magnetometer. An equation can then be written to express the measured magnetic field in terms of the magnetic field of the earth $\boldsymbol{B_e}$ and local fields on-board the Kambara $\boldsymbol{B_I}$,

$$ \boldsymbol{B} = \boldsymbol{R}^{T}(\boldsymbol{q})\boldsymbol{\alpha}\boldsymbol{B}_{e} + \boldsymbol{B}_{I} \qquad \textbf{(4.12)} $$

where $\boldsymbol{R}(\boldsymbol{q})$ is the rotation matrix converting from frame $\{B\}$ to frame $\{I\}$, $\boldsymbol{\alpha}$ is a 3×3 matrix used to model shielding, and $\boldsymbol{B}_e$ is given in table 4.2. Note that this definition of $\boldsymbol{B}_e$ sets the unit vector $\boldsymbol{X_B}$ of frame $\{B\}$ at magnetic north.

The shielding matrix $\boldsymbol{\alpha}$ describes how the magnetic field of the earth is attenuated and distorted in space as a function of position. The biggest amounts of distortion of the earth's magnetic field occur around ferromagnetic structures such as steel frames. There is no way to model this matrix unless the AUV environment has been pre-mapped. As this is unfeasible, the $\boldsymbol{\alpha}$ matrix must be estimated on-line through the Kalman filter.

Table 4.2: Components of the earth's magnetic field in free space.

B_{ex}	B_{ey}	B_{ez}
19.16μT	0μT	43.98μT

In general, all 9 elements of $\boldsymbol{\alpha}$ are non-zero. However, to keep the system completely observable, only 2 elements can be estimated. The best choice of shielding matrix to estimate turns out to be:

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_x & 0 & 0 \\ 0 & \alpha_x & 0 \\ 0 & 0 & \alpha_z \end{bmatrix} \qquad \textbf{(4.13)}$$

Writing the shielding matrix in this form assumes the field stays aligned in the $\boldsymbol{X_I}$-$\boldsymbol{Z_I}$ plane in the presence of any ferromagnetic materials. It also assumes the shielding is isotropic in the horizontal plane. This means that we cannot differentiate between a rotation of the earth's field, possibly due to a nearby magnet, from an actual rotation of the vehicle.

Since the shielding matrix terms are estimated by the Kalman filter, $\boldsymbol{\alpha}_x$ and $\boldsymbol{\alpha}_z$ must be added to the state vector. The appropriate process model is to estimate them as constants (ie. $\dot{\alpha}_x = \dot{\alpha}_z = 0$).

4.4 Triaxial Rate Gyroscope

The triaxial rate gyroscope on the MotionPak measures angular velocity in three mutually perpendicular directions. A straightforward model for rate gyroscope readings $\boldsymbol{\Omega}$ can be proposed:

$$\Omega = \boldsymbol{S}_\Omega \omega + \boldsymbol{k}_\Omega T + \boldsymbol{b}_\Omega \qquad \textbf{(4.14)}$$

In chapter 6, the temperature coefficient $\boldsymbol{k_\Omega}$ has been shown to be small yet measurable.

4.5 Depth Sensor

The depth sensor is used to measure the *z*-coordinate of the centre of mass of the AUV. It is to be located on the front-end cap of the lower enclosure, about 250mm away from the centre of mass. As the depth sensor is not actually located at the centre at the centre of mass, the displacement $\boldsymbol{r_d}$ from it must be accounted for:

$$d = S_d\left(z + {}^{B}\boldsymbol{Z}_{I} \cdot \boldsymbol{r}_d\right) + b_d \qquad \textbf{(4.15)}$$

If it could be assumed that the depth from the sensor to the centre of mass (ie. ${}^{B}\boldsymbol{Z}_{I} \cdot \boldsymbol{r}_d$) was constant, this equation could be simplified to

$$d = S_d z + b_d \,. \qquad \textbf{(4.16)}$$

Such an approximation seems reasonable, as the large righting moment of the AUV causes it to be oriented in or near the horizontal position. For example, a tilt of 15° can cause no more than a 50mm error. Errors of this magnitude may be significant. The resolution of the analog-to-digital converter channel used to log depth sensor readings is 1.2mV or 1mm, so the effects of making the approximation will definitely be seen. It may well be important to obtain accurate z-position readings to correct for drift in integrated accelerometer readings. But the actual effect of this on the state estimator will have to be investigated in final implementation.

4.6 Temperature Sensor

The temperature sensor on board the MotionPak will be used to correct for temperature effects. In contrast to the other analog sensors, it supplies its sensor output as a current. This current will be converted to a voltage through use of a resistor. The sensitivity S_T and bias b_T both depend on the choice of resistance *R*. The resultant equation is

$$T_m = RS_T T + Rb_T \qquad \textbf{(4.17)}$$

CHAPTER 5 - KALMAN FILTER ALGORITHM

5.1 Overview

The Kalman filter is a set of mathematical equations that provides a recursive least squares solution to the problem of estimating state variables in the face of sensor noise and an imperfect system model. The Kalman filter is a very powerful tool. Not only does it allow the estimation of present state, but also of past and future states. It also allows predictions of the position and velocity of the Kambara to be made. The Kalman filter is adaptive to time-varying situations, such as the changing accuracy of the sensors and components of the system model. This is a useful property as the accuracy of the sensors can change with time. For example, the TCM2 compass cannot measure inclinometer readings past 50°, causing it to rapidly lose accuracy past this angle of inclination.

The Kalman filter requires the model incorporated to be linear. As seen from chapters 2 and 4, the dynamic model and sensor model equations derived are highly nonlinear. To deal with these nonlinear equations, a modified Kalman filtering algorithm called the extended Kalman filter (EKF) must be applied. A critical component of this algorithm is the calculation of Jacobian matrices so that the nonlinear equations can be linearised. The calculation of such Jacobian matrices is made in section 2.3.

Another important consideration in the filter design is the modelling of the noise processes in the sensor model and dynamic model. Up until this chapter, these noise processes have not been discussed. However, they form an integral part of the model.

5.2 The Kalman filtering problem

Before describing the algorithmic details of the EKF, the problem that the discrete time Kalman filter addresses will be described. The discrete-time Kalman filter solves the problem of estimating the state $\boldsymbol{x}$ of a discrete-time process governed by the stochastic difference equation

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k \boldsymbol{x}_k + \boldsymbol{B}_k \boldsymbol{u}_k + \boldsymbol{w}_k , \qquad \textbf{(5.1)}$$

with a set of measurements $\boldsymbol{z}$ related to the state by

$$\boldsymbol{z}_k = \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{v}_k . \qquad \textbf{(5.2)}$$

As $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{H}$ depend upon k, there is no assumption of time invariance. Recall that $\boldsymbol{u}$ is the control input, $\boldsymbol{w}$ is the process noise random variable and $\boldsymbol{v}$ is the measurement noise random

variable. The standard Kalman filter models these random variables as white Gaussian noise with zero mean. Furthermore, they are assumed to be independent of each other.

The properties of the noise processes are dictated by their covariance matrices $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$. A covariance matrix is analogous to the variance of a scalar random variable. If $E(\cdot)$ is the expectation operator and $\delta(k)$ is the discrete-time delta function, these are defined by the equations,

$$E[\boldsymbol{w}_k \boldsymbol{w}_i^T] = \boldsymbol{Q}_k \delta(i-k) \quad \textbf{(5.3)}$$

$$E[\boldsymbol{v}_k \boldsymbol{v}_i^T] = \boldsymbol{R}_k \delta(i-k) \quad \textbf{(5.4)}$$

The presence of the delta function reflects the white nature of these random variables, as white noise is uncorrelated in time.

If these assumptions are unreasonable in a real system, all is not lost. Techniques are available for transforming such systems into forms that do have independence of measurement and process noise, as well as expressing the coloured noise processes in terms of white noise. (See Brown & Hwang [3] for details.) Fortunately, consideration of coloured noise has not yet been necessary.

The Kalman filtering algorithm operates in two distinct phases, a time update phase and a measurement update phase. In each phase, the error covariance matrix $\boldsymbol{P}_k$ is updated to keep track of the accuracy of the state estimate. The smaller the elements are in $\boldsymbol{P}_k$, the more accurate is the state estimate. In the time update phase, the process model is used to predict the state at the next time step $\boldsymbol{x}_{k+1}^-$ from the current state estimate $\boldsymbol{x}_k$. In the measurement update phase, the sensor readings are incorporated to obtain a better current state estimate $\boldsymbol{x}_k$ from $\boldsymbol{x}_k^-$. To do this, calculation of the Kalman gain matrix $\boldsymbol{K}_k$ must be made. $\boldsymbol{K}_k$ dictates how strongly state estimates depend on sensor measurements. For a sensor suite with little noise, $\boldsymbol{K}_k$ will be large. However, if the process model is significantly more accurate than the sensor measurements, $\boldsymbol{K}_k$ will be small. The above algorithm is iterated to obtain successive state estimates.

5.3 Extended Kalman filtering

As previously mentioned, the extended Kalman filter (EKF) allows the process and measurement equations to be nonlinear. This introduces a linearisation phase into the filter algorithm. It still contains a time update phase and a measurement update phase. A covariance matrix $\boldsymbol{P}_k$ is still used to keep track of the accuracy of current state estimates.

In the nonlinear case, the process and measurement equations are written in the form of $\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{w}_k)$ and $\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{v}_k)$. The EKF requires the linearisation of these equations to be

made about the current state estimate $\hat{\boldsymbol{x}}_k$. The state and measurement vectors can then be approximated without knowledge of the noise by $\hat{\boldsymbol{x}}_{k+1}^- = \boldsymbol{f}(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k, 0)$ and $\boldsymbol{z}_k^- = \boldsymbol{h}(\hat{\boldsymbol{x}}_k^-, \boldsymbol{u}_k, 0)$. The resulting linearised equations about the current state are

$$\boldsymbol{x}_{k+1} \cong \hat{\boldsymbol{x}}_{k+1}^- + \boldsymbol{A}_k(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k) + \boldsymbol{W}_k \boldsymbol{w}_k, \tag{5.5}$$

$$\boldsymbol{z}_k \cong \boldsymbol{z}_k^- + \boldsymbol{H}_k(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^-) + \boldsymbol{V}_k \boldsymbol{v}_k, \tag{5.6}$$

where $\boldsymbol{A}$, $\boldsymbol{W}$, $\boldsymbol{H}$ and $\boldsymbol{V}$ are Jacobian matrices. These matrices are defined as

$$(A_k)_{ij} = \frac{\partial f_i}{\partial x_j}(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k, 0), \tag{5.7}$$

$$(W_k)_{ij} = \frac{\partial f_i}{\partial w_j}(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k, 0), \tag{5.8}$$

$$(H_k)_{ij} = \frac{\partial h_i}{\partial x_j}(\hat{\boldsymbol{x}}_k^-, \boldsymbol{u}_k, 0), \tag{5.9}$$

$$(V_k)_{ij} = \frac{\partial h_i}{\partial v_j}(\hat{\boldsymbol{x}}_k^-, \boldsymbol{u}_k, 0). \tag{5.10}$$

For convenience, the time step index *k* has been dropped from the Jacobian matrices, even though they are different at each time step.

The time update equations are

$$\hat{\boldsymbol{x}}_{k+1}^- = \boldsymbol{f}(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k, 0), \tag{5.11}$$

$$\boldsymbol{P}_{k+1}^- = \boldsymbol{A}_k \boldsymbol{P}_k \boldsymbol{A}_k^T + \boldsymbol{W}_k \boldsymbol{Q}_k \boldsymbol{W}_k^T. \tag{5.12}$$

$\boldsymbol{P}_k$ is updated to reflect the drop in accuracy of the new state estimate. The measurement update equations are

$$\boldsymbol{K}_k = \boldsymbol{P}_k^- \boldsymbol{H}_k^T (\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{V}_k \boldsymbol{R}_k \boldsymbol{V}_k^T)^{-1}, \tag{5.13}$$

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k[\boldsymbol{z}_k - \boldsymbol{h}(\hat{\boldsymbol{x}}_k^-, \boldsymbol{u}_k, 0)], \tag{5.14}$$

$$\boldsymbol{P}_k = (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k)\boldsymbol{P}_k^-. \tag{5.15}$$

In this case, $\boldsymbol{P}_k$ is updated to reflect the improvement in the state estimate.

The whole EKF algorithm is summarised in figure 5.1 below.

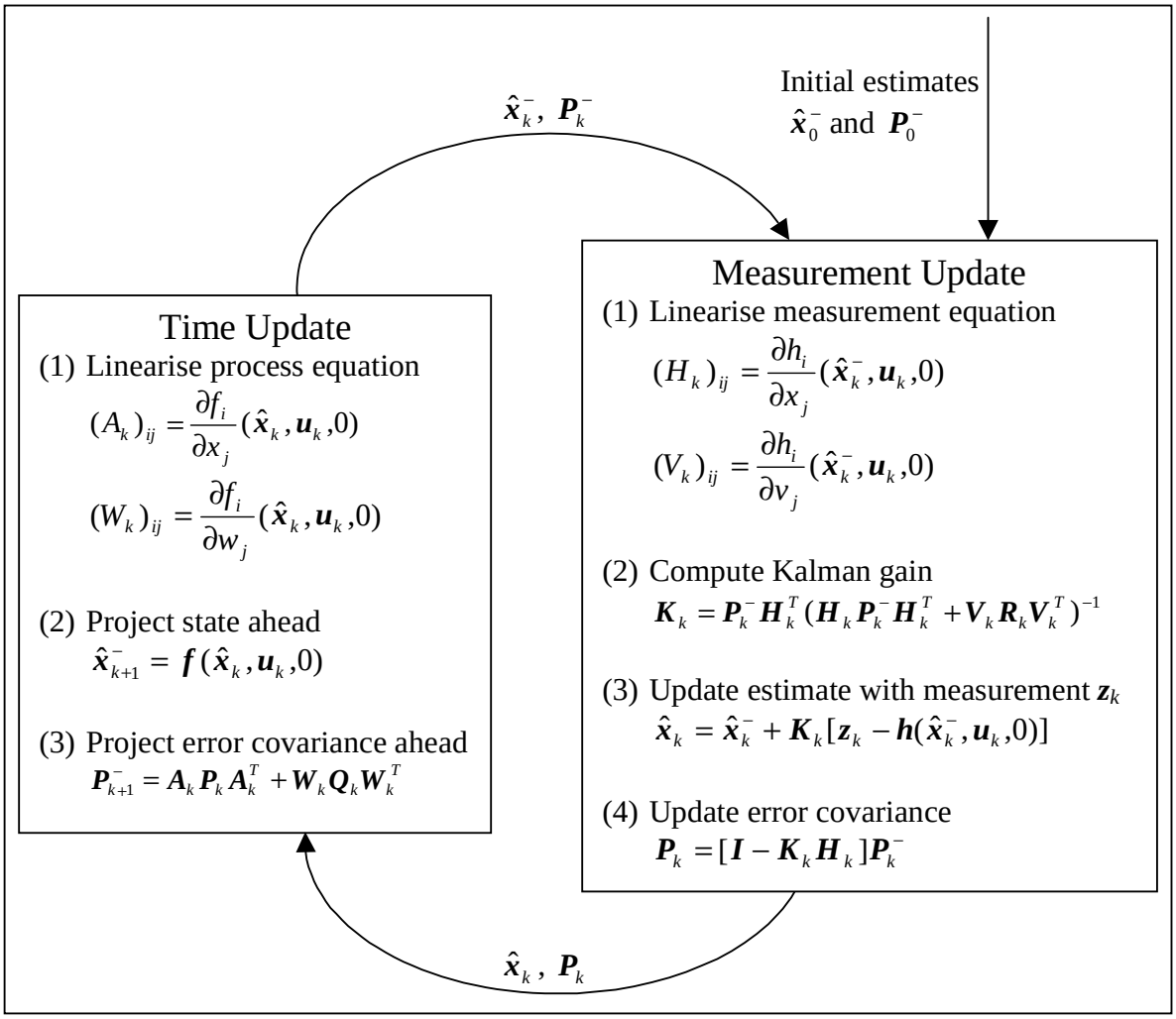


Figure 5.1: High level operation of the extended Kalman filter.

5.4 Jacobian Matrices

For the linearisation steps in the EKF algorithm to proceed, the Jacobian matrices need to be determined. Obviously, on-line symbolic differentiation in the final implementation is infeasible. As a result, the derivatives of the process and measurement model equations must be present in the final Kalman filter algorithm. Matlab code that utilises the Symbolic Toolbox to compute the Jacobian matrix derivatives is present in appendix 6. Once the relevant equations have been symbolically defined, this code uses the *jacobian* function to generate the Jacobian matrices. For final implementation, the matrices that are generated can be converted directly to C code through the *ccode* function.

CHAPTER 6 - HARDWARE / SOFTWARE HIERARCHY

A multitude of hardware and software has to be put into place before the state estimator becomes operational. This brief chapter outlines the hardware and software modules required for physical implementation of the Kalman filtering algorithm. It describes the device drivers that have been adapted or written as part of the project work.

6.1 Hardware / Software Heirarchy Overview

A block diagram of the components and software modules required for state estimator implementation is shown in figure 6.1. The sensor output signals are to feed into the computer system through ADC's and a (RS-232) serial port. The iPADIO and iPSerial modules implement these functions. Both of these modules run off a carrier board that in turn plugs into a PCI bus. A 233MHz PowerPC running under the VxWorks operating system controls these three modules. The state estimation algorithm also runs on the PowerPC. Required for its operation is an appropriate matrix/vector library that has been developed from existing code.

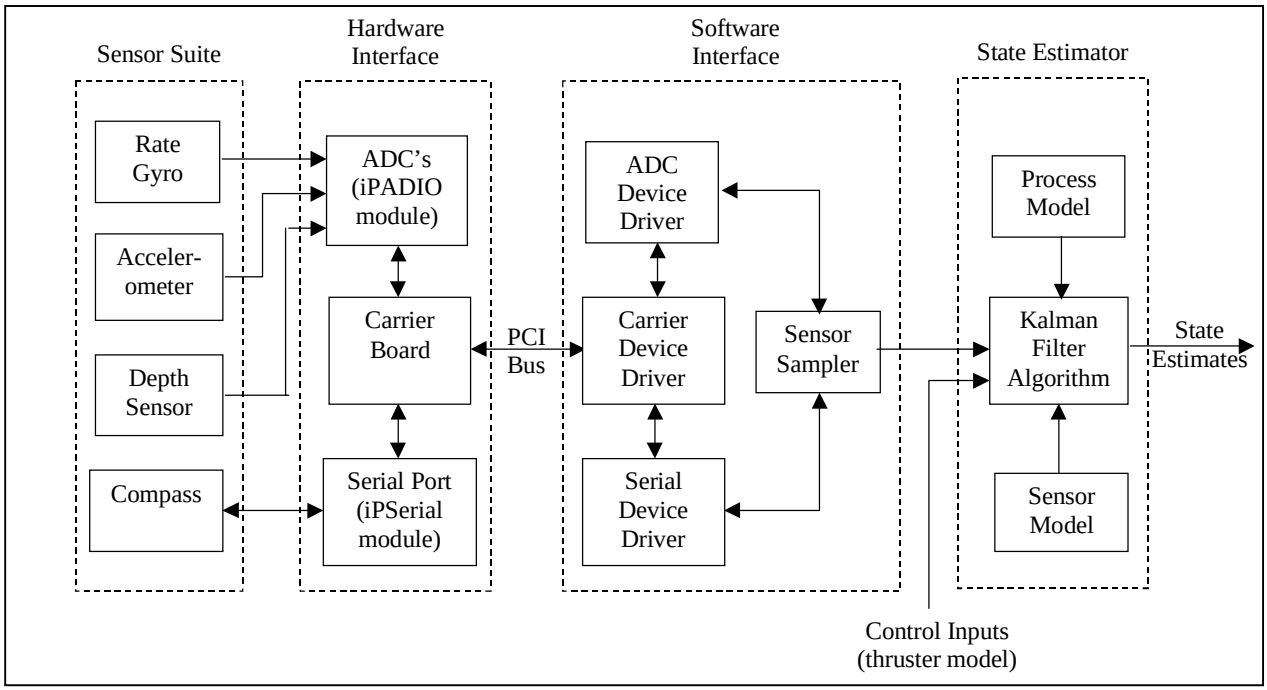


Figure 6.1: Block diagram of components and software modules in final state estimator implementation. Prefiltering of the analog sensor outputs has not been shown.

6.2 Device Drivers

The carrier board, iPSerial module, iPADIO module and carrier board all require device drivers for functionality. All device drivers were developed in C on a Pentium II machine. For implementation, endian issues will have to be addressed, as PCs are little endian while

PowerPCs are big endian. The current state of the device drivers is described below. The relevant code is listed in appendix 5.

6.2.1 Carrier Board Driver

The carrier board transfers information between the PowerPC and the industryPack modules mounted onto it (such as the iPSerial module). An important function of the carrier board is the piping of interrupts from industryPack modules such as the iPSerial module to the central processor.

iPpci9080Lib.c contains code relevant to the carrier board driver. It is initialised by calling *iPpci9080Init()*. This routine must be called before any industryPack modules are intialised. Useful debugging information can be obtained by calling *iPpci9080Show()*. Before unloading this driver from memory in VxWorks, it is important to disable carrier board interrupts with *iPpci9080DisableInts()*.

6.2.2 iPADIO Module Driver

The iPADIO module provides 16 single-ended ADC inputs and upto 64 digital input/outputs . 12 of these input lines are connected to the differential outputs of the MotionPak rate gyroscope and accelerometer. Another 2 are connecting to the pressure sensor and MotionPak temperature sensor, leaving 2 ADC channels to spare. Currently, the driver to this module allows manual logging of the ADC and 48 digital I/O lines up to 67Hz, as well as automatic logging of sensor data up to 1kHz. Unfortunately, significant noise is introduced by sampling in automatic mode, rising rapidly in magnitude as sampling rate is increased above 100Hz. However, automatic mode reduces jitter.

Critical driver code is contained within *iPADIO.c*. Calling *iPADIOInit()* initialises the device driver. *iPADIOAutoOn()* and *iPADIOAutoOff()* turns automatic data logging on and off. *iPADIOReadADC()* manually polls an ADC channel. Please see iPADIO.c for further details.

6.2.3 iPSerial Module Driver

The iPSerial module utilised the 8530 chipset to allow asynchronous transfers of up to 38400 baud. The device driver has been configured to send with 1 start bit, 8 data bits, 1 stop bit and no parity. Baud rate defaults to 9600. Relevant driver code is present in iPSerial.c and z8530Lib.c. It is initialised by calling *iPSerialInit()* and removed by calling *iPSerialUnInit()*.

Unfortunately at the moment the serial port places considerable load upon the CPU. Unlike the 16550 chipset, the 8530 provides no FIFO buffering. If it did, serial port interrupts would be generated less frequently. It does however possess DMA capabilities that could be activated to automate the serial driver operation more. However, the carrier board driver would have to be extended as well.

CHAPTER 7 - SENSOR TESTING

7.1 Introduction

For full functionality of the sensor model developed in chapter 4, the sensor parameters must be estimated. Specifically, these sensor parameters are

- sensor sensitivity $\boldsymbol{S_i}$
- zero input offset voltage $\boldsymbol{b_i}$, and its dependence on temperature $\boldsymbol{k_i}$
- statistical models of the noise distributions measuring the noise covariances R_{ij}

In the case of the rate gyroscope and triaxial accelerometers, sensitivities have been accurately determined experimentally by the manufacturers. Also, Beswick [2] contributed accurate measurement of the sensitivity of the depth sensor. However, the temperature dependent bias has not been determined. The temperature coefficients have been estimated in section 7.2.

The Kalman filter equations assume the statistical models of the sensor noise distributions to be white Gaussian. It is desirable either to justify these assumptions, or otherwise create sensor noise models expressed in terms of white Gaussian noise. The noise distributions for the rate gyroscope and triaxial accelerometers have been explored in section 7.3.

Finally, the performance of the TCM2 compass module is investigated in section 7.4. In particular, the influence of thruster magnetic fields on compass operation is explored.

7.2 Temperature Coefficient Measurement

The temperature coefficients $\boldsymbol{k_a}$ and $\boldsymbol{k_\Omega}$ of the accelerometers and rate gyroscope were measured by logging sensor data over extended periods of time. The temperature of the MotionPak and Precision Instruments accelerometer was measured using the AD590 temperature sensor on-board the MotionPak. In doing so, the Precision Instruments accelerometer was assumed to be in sufficient thermal contact with the MotionPak for both units to be at approximately the same temperature. This assumption was found to be reasonable, as the aluminium mounting plate is a good conductor of heat.

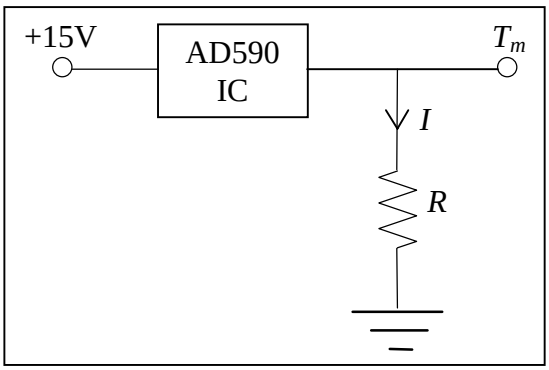


Figure 7.1: AD590 current to voltage conversion circuit.

As the output signal of the temperature sensor is a current, and the ADCs measure voltages, the current of the temperature sensor had to be converted to a voltage. This was achieved by inserting a resistor between the sensor output and ground (figure 7.1). The specifications for this resistor are:

1. The resistor should not drain excessive current from the AD590 IC (definitely no more than it can source).
2. The resistor current should be large enough for the ADC current drain to be negligible.

Given a typical AD590 output current I of 300μA and ADC current drain of approximately 1μA, and ADC full scale range of 10V, R=20kΩ has been chosen as appropriate. This resistor has been found to set the voltage at ambient temperature to ~6V.

In testing temperature dependence, no attempt was made to map temperature sensor readings to actual temperatures. The temperature readings have all been left in units of volts. Besides accounting for changes in resistance R (see figure 7.1 above), the actual units of temperature are unimportant. For all experiments, a value of R of 21.2kΩ has been used.

The first test constituted of leaving the sensor suite on in the RSISE undergraduate laboratory overnight and logging sensor data to disk. The temperature within this room is prone to considerable variation (at least 10°C) over a 24-hour period. The x-axis of the Summit Instruments triaxial accelerometer was found to exhibit the most sensitivity to temperature. The result of logging the data from this sensor over an 18-hour period is shown in figure 7.2. This graph shows a significant (0.3V) variation of zero bias voltage with temperature. It also shows the value in using the MotionPak temperature sensor in estimating the temperature of the Summit Instruments triaxial accelerometer.

Figures 7.3 – 7.5 show the relationship between temperature and accelerometer bias. From these graphs, an estimate of the temperature coefficient $\boldsymbol{k_a}$ can be made. The slopes of the lines of best fit suggest $\boldsymbol{k_a} = (0.391, 0.184, 0.031)^T$. As anticipated, the elements of $\boldsymbol{k_a}$ vary significantly between the three axes.

In comparison to the Summit Instruments accelerometer, the accelerometer and rate gyroscope on the MotionPak were found to be relatively temperature insensitive. In fact, the temperature sensitivity was so small that it could not be measured with the above means of testing. Zero point voltage needed to be measured over a larger temperature range. To achieve this, another test was run, this time by placing the 50W light bulb of a desk lamp above the MotionPak.

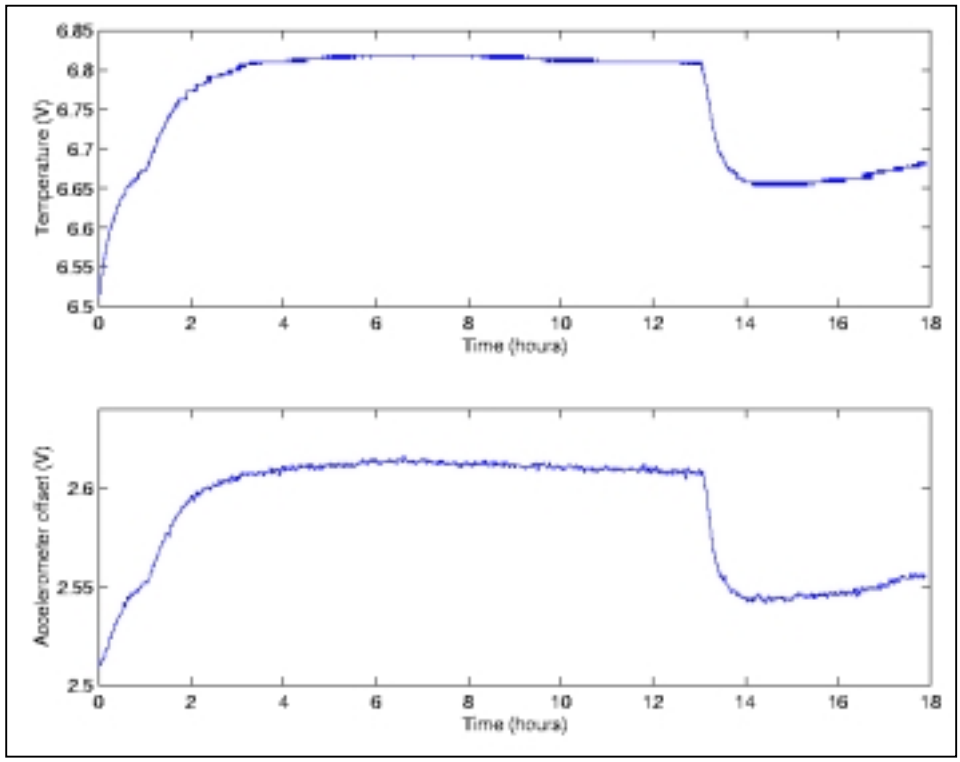


Figure 7.2: Graphs of MotionPak temperature data and Summit Instruments accelerometer *x*-axis data versus time. The accelerometer data has been filtered with a 5th order Butterworth filter to reduce noise.

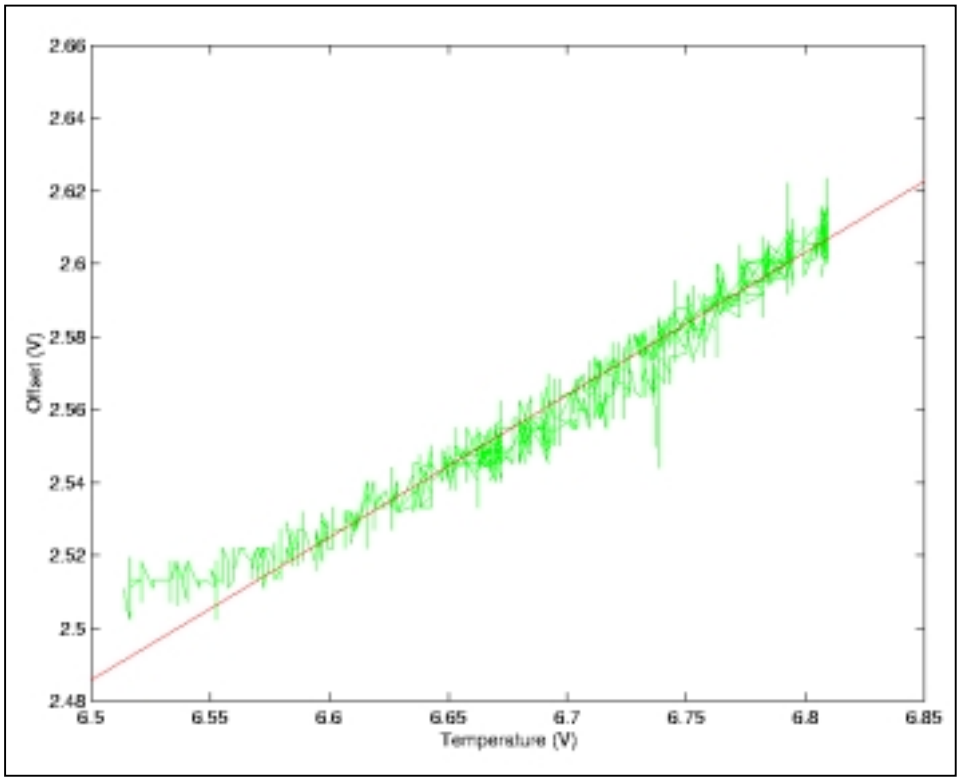


Figure 7.3: Dependence of Summit Instruments triaxial accelerometer *x*-axis on temperature. The slope of the line of best fit is 0.3912V/V

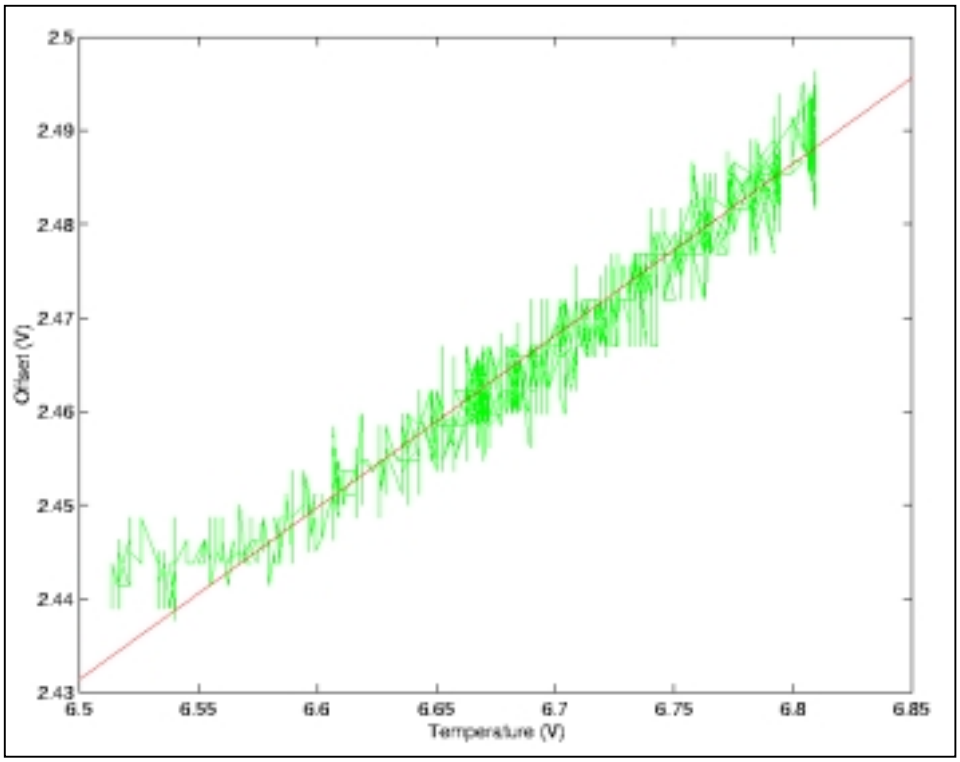


Figure 7.4: Dependence of Summit Instruments triaxial accelerometer *y*-axis on temperature. The slope of the line of best fit is 0.1836V/V.

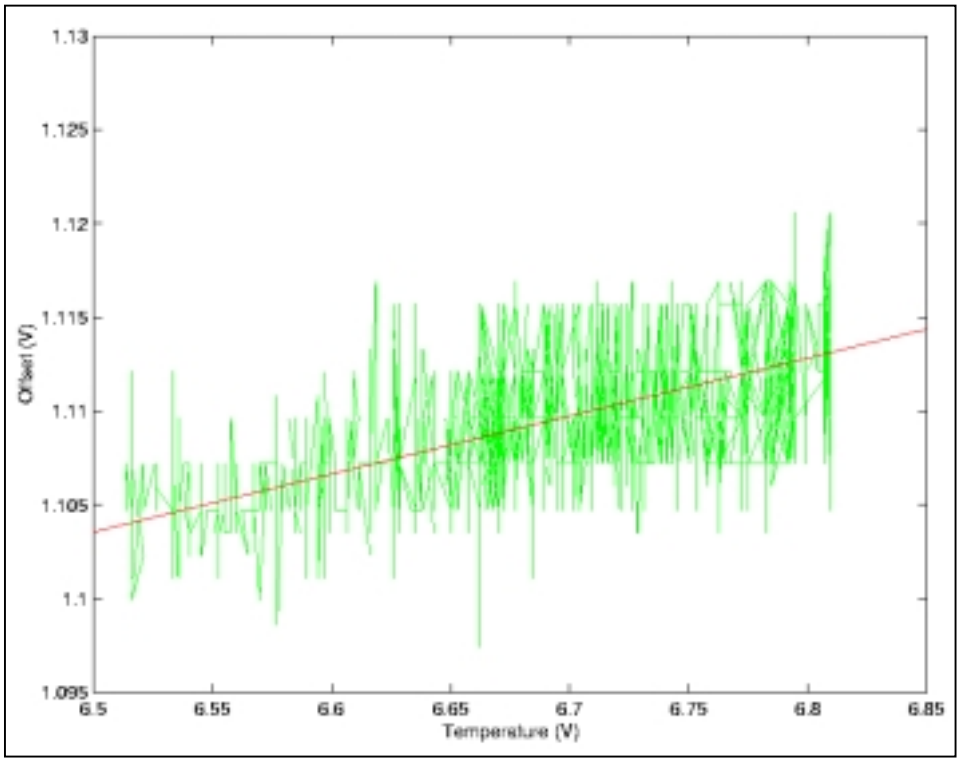


Figure 7.5: Dependence of Summit Instruments triaxial accelerometer *z*-axis on temperature. The slope of the line of best fit is 0.0309V/V.

Similarly, the temperature coefficient of the MotionPak rate gyroscope was determined to be $\boldsymbol{k}_{\boldsymbol{\Omega}}=(-0.0041, -0.0074, -0.0187)^T$. The fact that it is so much smaller than that for the Summit Instruments accelerometer (and negative) suggests some temperature compensation circuitry is used on-board the MotionPak.

Unfortunately the temperature coefficients of the MotionPak accelerometer have not been determined. In the second test, RC filter circuits were introduced between the sensor outputs and the ADC input lines. The purpose of these circuits will become apparent in section 7.3 below. Moreover, in this configuration, the accelerometer outputs reverse biased the unipolar capacitors used in the RC circuitry, shorting the ADCs to ground and voiding the accelerometer results.

7.3 Noise Distribution Testing

A complete sensor model requires the modelling of the statistical noise distributions of the sensors. This section develops such models for the accelerometers and the rate gyroscope from experimental data.

The models of noise distributions were obtained as follows:

1. Each sensor was sampled for 7 – 8 minutes at a fast rate. The sampling rate chosen was no less than the expected maximum sampling rate of the final state estimator (ie. 40Hz).
2. The power spectral density of the resulting sensor data was estimated using the Matlab *psd* command. The graphs of such power spectral density were used to determine whether sensor noise could be well approximated by white noise (which has constant power spectral density). Note that the DC component of the noise has been subtracted from all power spectral density graphs.
3. The noise covariances were estimated by calculating the standard deviations of the sensor data. These standard deviations are present in table 7.1.

The power spectral density of the Summit Instruments triaxial accelerometer is shown in figure 7.6 below. The origin of the frequency spikes at 16Hz is unknown. However it looks like the spike at 33Hz is the second harmonic of the 16Hz spike. Since these spikes possess only a small fraction of the noise power, their presence should not be too significant.

The more important characteristic of the power spectral density is attenuation of higher frequency noise. This noise distribution obviously cannot be modelled as simply as a white Gaussian noise. A better approximation is to model the noise with a *Gauss Markov* process. A Gauss Markov process is a white Gaussian noise process passed through a low pass filter. The simplest type of Gauss Markov process is produced with a first order low pass filter. If

implemented, an extra variable would need to be added to the process model[4]. However, as the MotionPak accelerometer will be used instead, it is unnecessary to develop such a model any further.

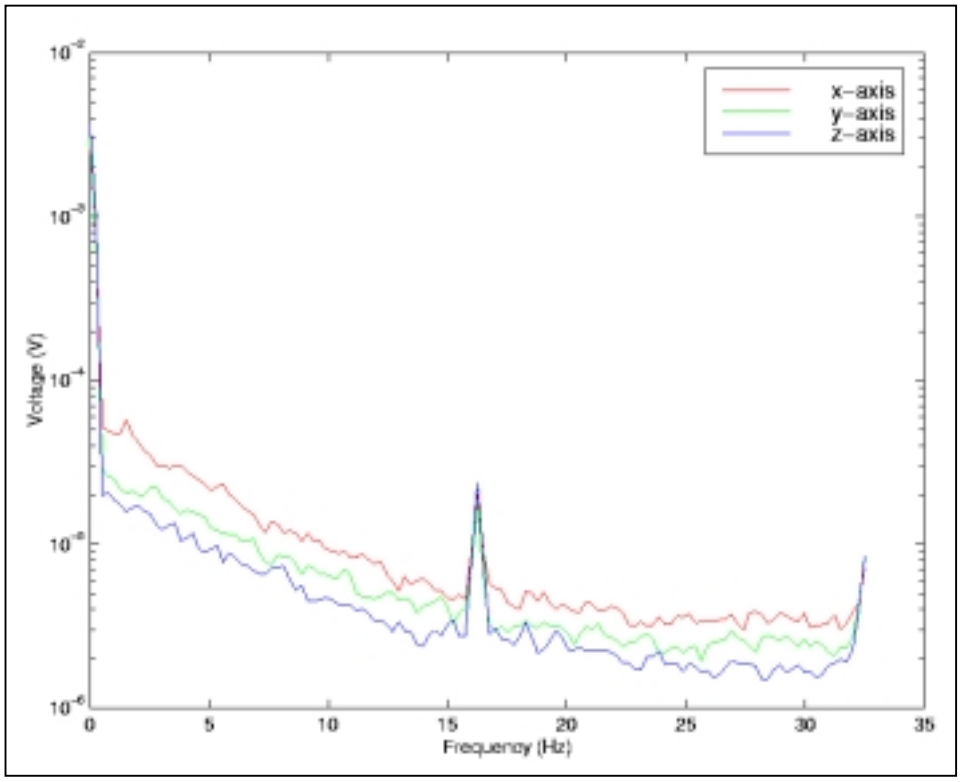


Figure 7.6: Power spectral density from the unfiltered Summit Instruments accelerometer data logged at 67Hz.

The power spectral density of the rate gyroscope noise is presented in figure 7.7. Three large frequency spikes (one for each axis) are shown in this graph. These spikes are significant sources of noise, as their magnitudes are around 100 times the size of the background noise. Figure 7.8 demonstrates their significance, as the presence of a low frequency component can clearly be seen. If it were not for the presence of these spikes, the rate gyroscope noise could be well modelled as white Gaussian. This is because the power spectral density curve is otherwise (approximately) constant.

The spikes in figure 7.8 are actually the result of high frequency noise aliasing onto the 0 – 52Hz frequency spectrum. The fact that the spikes do result from aliasing becomes obvious by sampling at a different frequency. The frequency spikes will shift, each by the same amount (see figure 7.11 for an example of this).

[4] The process model for the added state $m(t)$ would be $\dot{m}(t) = km(t) + n(t)$ where $n(t)$ is a white Gaussian noise process and k is the corner frequency of the low pass filter through which the Gaussian noise is passed.

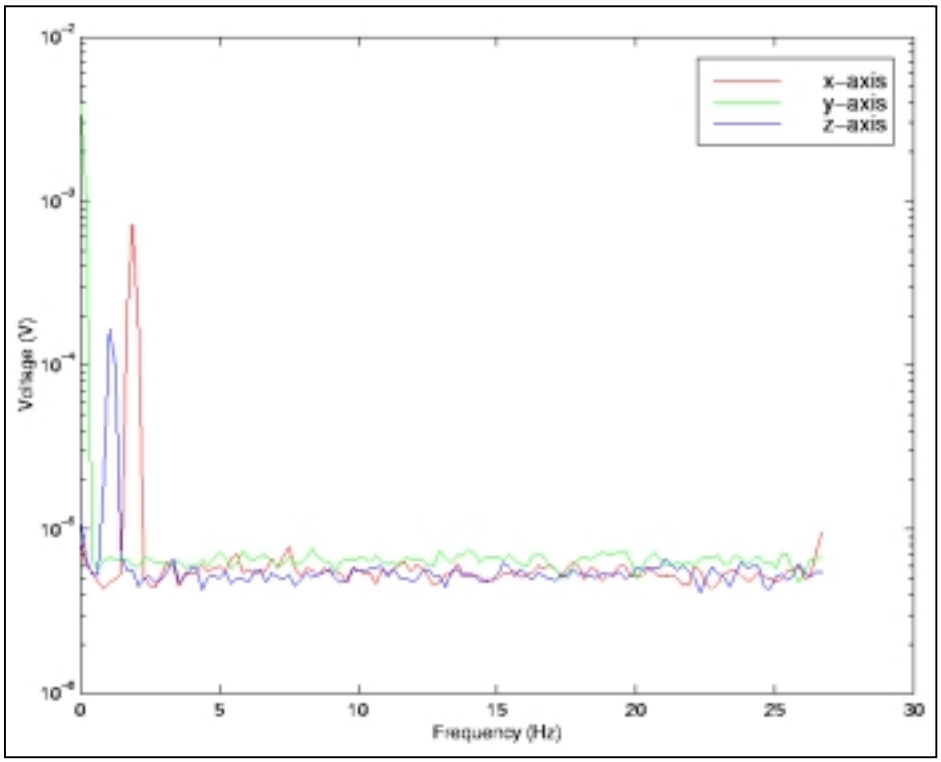


Figure 7.7: Power spectral density from the unfiltered rate gyroscope data logged at 52Hz.

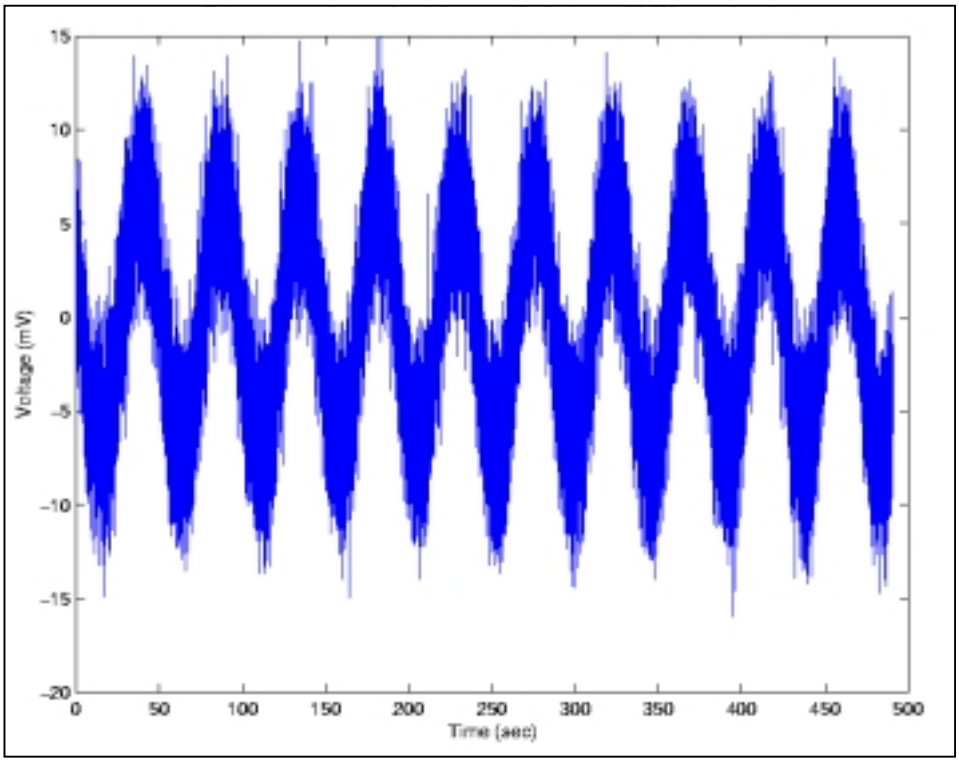


Figure 7.8: Oscillation in the unfiltered *y*-axis rate gyroscope data. The low frequency oscillations are caused by aliasing of a high frequency noise component.

Because the spikes in the rate gyroscope power spectral density are high frequency, they can be successfully removed by low pass filtering. The higher frequency white Gaussian noise would also be removed by a low pass filter, reducing the noise covariances and increasing state estimator performance. The design of a suitable low pass filter is hence desirable.

The simplest type of low pass filter to use is a first order one, generated by the RC circuit presented in figure 7.9. The critical design constraints of this circuit are:

(1) R is small enough for the voltage drop between V_{in} and V_{out} to be negligible.
(2) C is not too large, as capacitor size begins to increase rapidly after ~1μF.
(3) Cutoff frequency f is given by $f = 1/2\pi RC$. This should be at least double the sampling frequency. If cutoff frequency is low, the RC circuit would have a significant effect on the sensor outputs

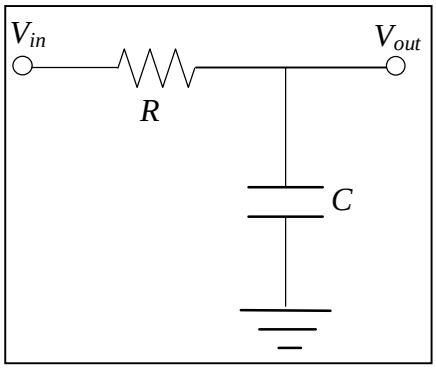


Figure 7.9: RC circuit configuration.

Using a corner frequency of 120Hz, R=380Ω and C=3.3μF were chosen. The resulting frequency response of the circuit is shown in figure 7.10. The improvement in the power spectral density of the rate gyroscope was significant. Figure 7.11 below shows the magnitude of the frequency spikes to be attenuated by a factor of 10. This degree of attenuation indicates the high frequency noise component occurring around 1kHz.

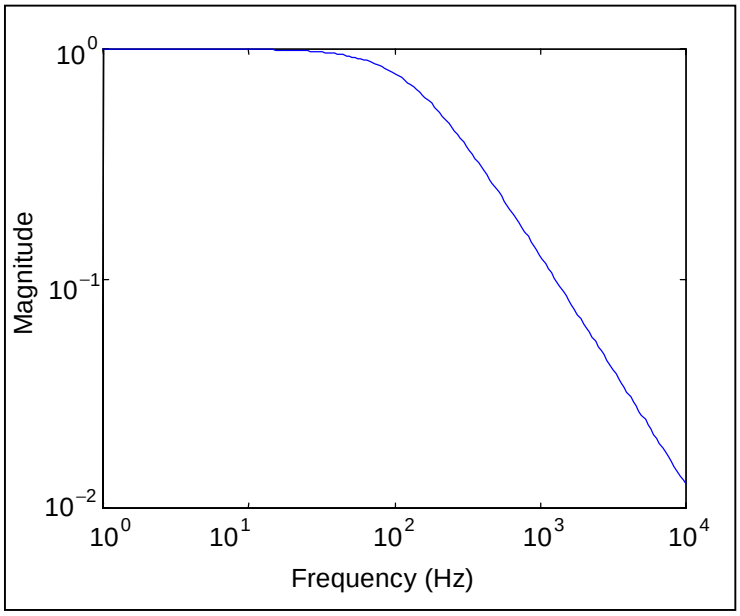


Figure 7.10: Magnitude response of the RC circuit used for testing of the MotionPak sensors.

Figures 7.12 and 7.13 below show the difference the RC circuits makes to the MotionPak accelerometer noise. Insertion of the RC circuit allows this sensor output to be well approximated by white noise too.

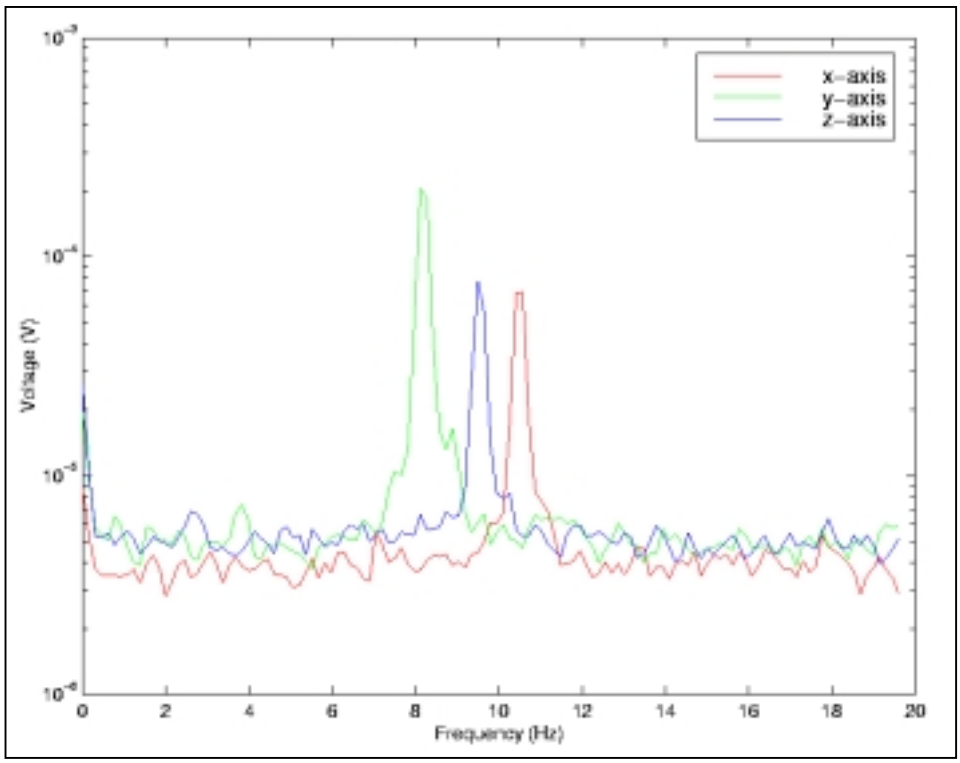


Figure 7.11: Power spectral density from the RC-filtered MotionPak rate gyroscope data logged at 40Hz.

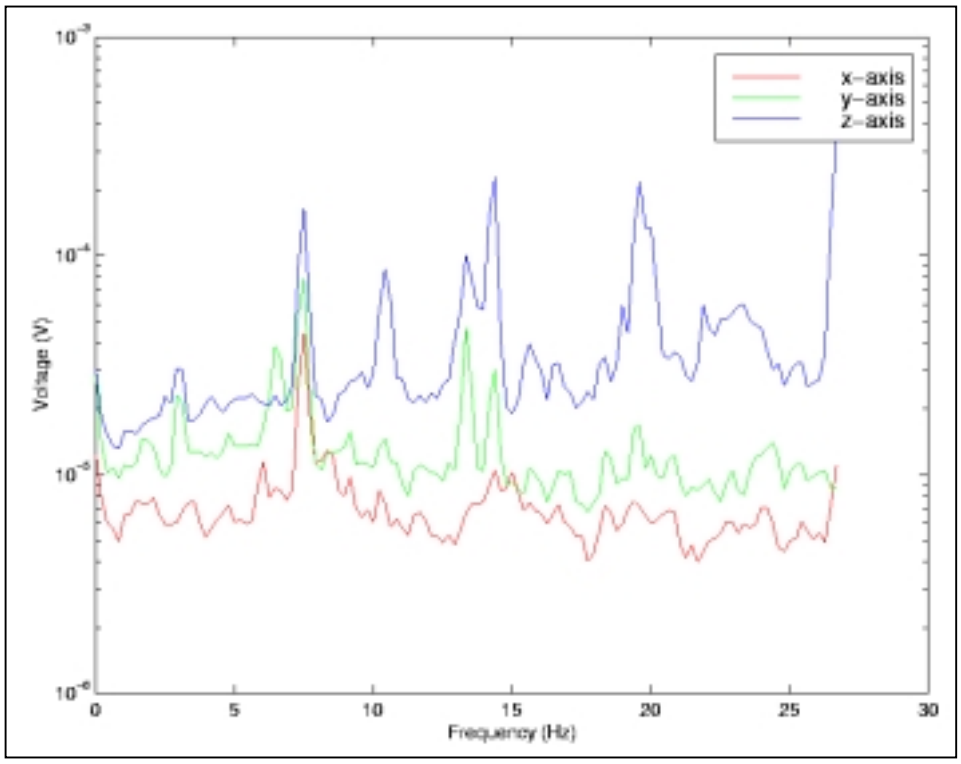


Figure 7.12: Power spectral density from the unfiltered MotionPak accelerometer data logged at 52Hz.

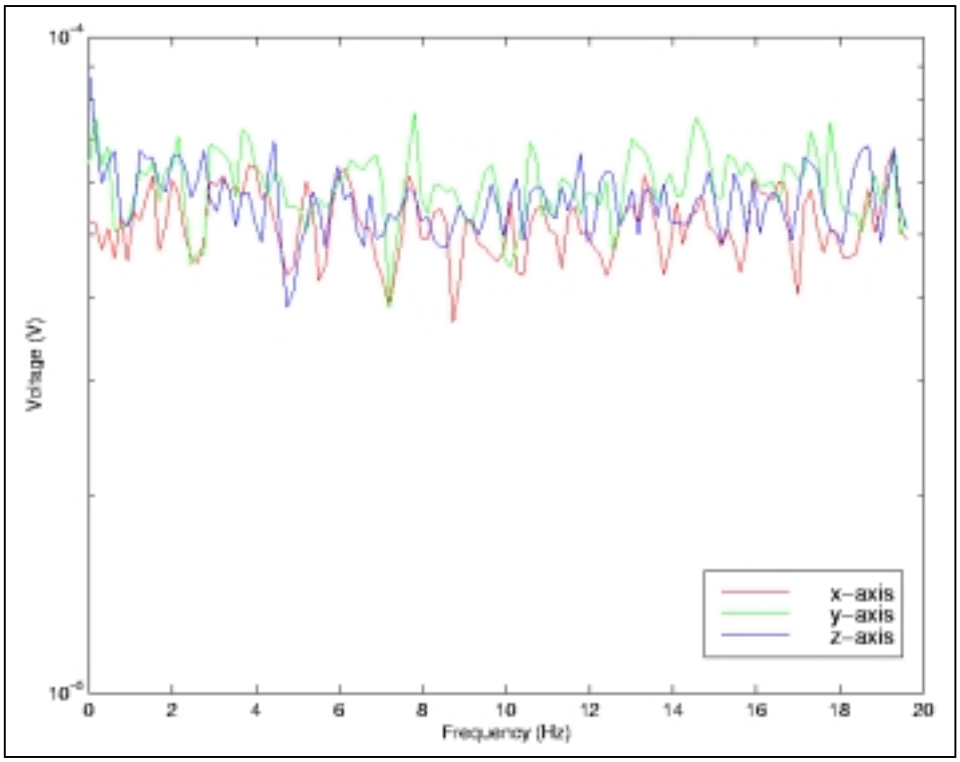


Figure 7.13: Power spectral density from the RC-filtered MotionPak accelerometer data logged at 40Hz.

A summary of the noise standard deviations is presented in table 7.1 below. All data upon which these are based was taken with the ADC driver set in manual mode. An important observation to make is the decrease in noise that results in filtering the sensor outputs.

Table 7.1: Noise standard deviations, with and without filtering.

Sensor name	Filtered?	Standard deviation (V)		
		X	Y	Z
Summit accelerometer	N	7.2	5.4	4.3
MotionPak rate gyroscope	N	6.2	5.5	5.3
MotionPak accelerometer	N	6.3	6.6	5.3
MotionPak rate gyroscope	Y	2.3	3.1	2.6
MotionPak accelerometer	Y	2.7	3.7	4.5

7.4 TCM2 Compass Investigation

For estimation of the heading, the TCM2 compass uses a triaxial magnetometer to determine the direction of the (rather weak) magnetic field of the earth's core. Electrical devices on board the Kambara are likely to produce magnetic fields comparable to the earth's field. Much of this interfering field will be constant and can be corrected by calibrating the TCM2 correctly. For example, the DC-DC converter was found to introduce a small 0.5μT constant field component.

A major concern was that the thrusters would produce a significant dynamic field component that would interfere with compass readings. This was found not to be the case. The influence of the thrusters on the compass readings was quantified by placing the compass in several different positions about one of the thrusters of Kambara, and for each position, to observe the change in compass bearing as a function of thruster angular speed. Compass to thruster distance ranged between 200 – 400mm.

The results showed that the dynamic field component of the thruster magnetic field was relatively minor. By varying thruster speed between 100 and 1800rpm, magnetic field strength was observed to change by only 1μT in magnitude, with a corresponding change of 1° in measured bearing angle.

A much larger contribution to the magnetic field came from the static field of the thruster. At a distance of 400mm down the axis of the thruster, the static field component was so large as to saturate the TCM2 magnetometer. In this position, it exceeded the 80μT full scale range of the magnetometer. Fortunately, a 200mm offset from the axis of the thruster produced more readily measurable results. Magnetometer readings were 60μT or below, fitting within full scale range.

CHAPTER 8 - FURTHER DISCUSSION

8.1 Water Current Effects

The equations describing the behaviour of the Kambara derived in chapter 2 are highly nonlinear. However, even though a number of nonlinear effects are accounted for, several important ones have been ignored completely on the basis of difficulty of modelling. The dynamic model makes assumptions on the nature of the body of water through which the AUV moves. The body of water is assumed to be infinite in all directions, and to contain no currents (except for those generated by the AUV itself.)

In practical applications, the AUV will encounter wave disturbances and water currents. The process model developed does not correct for them. Indeed it would be difficult to factor them into the model, as the water flow properties obviously depend upon the body of water in which the AUV is operating. These water flow properties would have a detrimental effect to the operation of the state estimator. A constant water flow would cause the acceleration, hence position and velocity estimates to become biased. Waves would introduce oscillation into the state estimates.

The body of water will not be infinite in all directions either, as the AUV will be operating around other underwater structures. Moving close to underwater structures would modify added mass or drag characteristics. However, at reasonable distances from underwater structures (say 1m), the effect from this factor should be negligible.

Fortunately, for lake and deep-sea operation, the effects of waves can be considered negligible. Waves are really only a surface phenomenon. At depths below about 20 metres, the presence of wave disturbances is minimal.

8.2 Usage of Both Accelerometers

It was suggested that incorporation of readings from both accelerometers would increase the accuracy of state estimates. The general philosophy of Kalman filtering is that the more sensor data is available, the better will be the state estimates. However, in consideration of the implementation difficulties that result and an only minor enhancement, exploration of this idea below shows it to be infeasible.

One can expect two types of improvements in state estimates by using both accelerometers. Firstly, the noise in the filtered output can be reduced. For example, for 2 identically distributed and independent Gaussian random processes, the effect of averaging reduces noise covariance down by 30%.

Secondly, by separating the two sensors so that they are as far as possible away from each other, estimates of the angular velocity could be obtained by taking the difference between these two sensor readings. In the simple case of the AUV moving with a constant angular speed ω with the accelerometers separated by length l, the difference in acceleration experienced between the two accelerometers is given by $a = l\omega^2$. For example, if the accelerometers were separated by 1 metre (which is the maximum they can be separated in Kambara), a 30mV reading on a rate gyro axis will be measured as a 6mV difference in accelerometer readings. The signal to noise ratio is 5 times less. And this figure assumes both accelerometers to be of same quality as the one on the MotionPak. They are not.

After considering the enhancements to state estimation above, there are a couple of inconviencies that make using both sensor difficult. It is inconvient to locate different sensors on opposite sides of the enclosure, both for wiring and mounting. The analogue to digital converter currently on-board does not possess enough input lines for measurement of the data of both sensors. In consideration of these factors, it is not planned to sample data from both sensor simultaneously.

8.3 Prefiltering of Sensor Inputs

Unfortunately, it was found that the simple RC filter network designed in chapter 6 had a significant impact on the accuracy of the ADCs. Each ADC was found to significant quantities of current from each sensor output. The net result of this current drain is a drop across the resistor of the RC circuit used for filtering. The drop was 4mV with a 4V input voltage, which is noticeable.

There are several possible solutions for this problem. The first is to reduce the size of the RC circuit resistor. However, this increases cutoff frequency, attenuating the high frequency noise spikes in rate gyroscope readings less. It is infeasible to increase the capacitance, as at 3.3µF, capacitor sizes begin to grow rapidly. Besides, capacitors above 1µF are only really available in unipolar form when bipolar capacitors are required.

The second alternative is to perform some type of active filtering. For example, a buffer could be placed between the RC circuit and ADC input. A bonus to this method is that as the voltage drop of the RC circuit resistor suddenly becomes negligible, smaller capacitor and larger resistor values could be used. Unfortunately, to implement such filtering, 12 operational amplifier circuits (two for each sensor output) will have to be used. Wiring would be messy.

CHAPTER 9 - CONCLUSION & FUTURE WORK

A solid foundation for the design and implementation of an accurate and robust state estimation system has been layed. Although not as many project goals were achieved as initially planned, a considerable amount has been accomplished.

- A dynamic model of the Kambara and sensors has been researched, developed and implemented in Matlab.
- Suitable sensor mounting was designed for mounting into the Kambara environment and onto the WAM arm for future dynamic testing.
- A significant portion of coding work has also been done. Work has been completed on three device drivers. A matrix/vector library was adapted for use under the VxWorks operating system and its functionality extended. A TCM2 compass layer has been coded.
- Static testing of the sensors has begun for the sake of modelling noise processes and temperature sensitivities. Prefiltering of the sensor outputs was shown to yield sizable reduction in sensor noise. Filtered MotionPak rate gyroscope and accelerometer noise was shown to be well modelled as white Gaussian.
- The MotionPak accelerometer has been identified as superior to the Summit Instruments accelerometer.

A large amount of work is still needed for final implementation, however. The sensors must be further tested, both statically (to determine more noise properties and temperature coefficients) and dynamically (to measure zero point offsets and sensor sensitivity). The rest of the sensor sampler algorithm and whole of the Kalman filter algorithm still must be coded. Hydrodynamic parameters of the dynamic models need estimating, either through physical testing or computer-based modelling. A Proengineer model of the Kambara's geometry has been provided.[5] This model needs to be extended to estimate the centres of gravity and buoyancy of the Kambara. Final implementation will require tuning of the Kalman filter covariance matrices for optimum performance.

[5] Courtesy of Harley Truong

REFERENCES

[1] Bar-Shalom Y., Li, X., *Estimation and Tracking – Principles, Techniques and Software*, Artech House, Inc., 1993.

[2] Beswick D., *Inertial Navigation System for an Autonomous Underwater Vehicle*, Department of Engineering, Australian National University, 1998.

[3] Brown R. G., Hwang P. Y. C., *Introduction to Random Signals and Applied Kalman Filtering*, John Wiley & Sons, 2nd Edition, 1985, pp.254-257.

[4] Bono R., "Romeo goes to Antartica", *IEEE Conference Proceedings*, 1998.

[5] Craig J. J., *Introduction to Robotics*, 2nd Edition, Addison-Wesley Publishing Company, Sydney, 1986, pp.48-55.

[6] Fjellstad O., Fossen T. I., "Position and Attitude Tracking of AUVs: A Quaternion Feedback Approach", *IEEE Journal of Oceanic Engineering*, Vol. 19, 1994.

[7] Fossen, T. I., "Underwater Vehicle Dynamics", Norwegian Institute of Technology, University of Trondheim, 1994.

[8] Gaskett C., Wettergreen D., Zelinsky A., "Reinforcement Learning Applied to the Control of an Autonomous Underwater Vehicle", Research School of Information Sciences and Engineering, Australian National University, 1998.

[9] Gelb A. et al, *Applied Optimal Estimation*, The Analytical Sciences Corporation, M.I.T. Press, 1974.

[10] Shing L. H., *Platform Control of an Autonomous Underwater Vehicle*, Department of Mechanical and Mechatronics Engineering, University of Sydney, 1997.

[11] Silpa-Anan C., "Thruster Modelling and Control", Research School of Information Science and Engineering, 1999.

[12] Smith, R., Frost, A., Probert, P., "A Sensor System for the Navigation of an Underwater Vehicle", *The International Journal of Robotics Research*, Vol. 18, 1999.

[13] Stovall S. H., *Basic Inertial Navigation*, Systems Integration Branch, Naval Air Warfare Centre Weapons Division, 1997.

[14] Toh T. C., *Platform Control of an Autonomous Underwater Vehicle*, Department of Mechanical and Mechatronics Engineering, University of Sydney, 1997.

[15] Wake M., *Inertial Navigation of an Autonomous Underwater Vehicle*, Department of Mechanical and Mechatronics Engineering, University of Sydney, 1997.

[16] Welch G., Bishop G., "An Introduction to the Kalman Filter", UNC-CH Computer Science Technical Report 95-041, 1995.
http://www.cs.unc.edu/~welch/kalman/kalman.html

APPENDICES

9.1 APPENDIX 1 – Tables of Symbols Used

Tables A1.1 to A1.4 below summarise the many symbols used to represent quantities in the process and measurement models.

Table A1.1: Definition of symbols used in the state vector $\boldsymbol{x}$. Inclusion of the shielding matrix parameters is only relevant in the extended TCM2 compass sensor model.

Physical quantity	Symbol	Unit	Frame
Linear velocity at centre of mass	$\boldsymbol{v} = (u\ v\ w)^{\mathrm{T}}$	m/s	$\{B\}$
Angular velocity	$\boldsymbol{\omega} = (\omega_x\ \omega_y\ \omega_z)^{\mathrm{T}}$	rad/s	$\{B\}$
Position of centre of mass	$\boldsymbol{r} = (x\ y\ z)^{\mathrm{T}}$	m	$\{I\}$
Attitude quaternion	$\boldsymbol{q} = (\eta\ \varepsilon_1\ \varepsilon_2\ \varepsilon_3)^{\mathrm{T}}$	–	–
Sensor suite temperature	T	°C	–
Shielding matrix parameters	$(\alpha_1\ \alpha_2)^{\mathrm{T}}$	–	$\{I\}$

Table A1.2: Definition of symbols used in the sensor reading vector $\boldsymbol{z}$. Inclusion of magnetometer readings (and exclusion of bearing readings) is only relevant in the extended TCM2 compass sensor model.

Sensor name	Sensor quantity	Symbol(s)	Unit
MotionPak	Acceleration	$\boldsymbol{a} = (a_x\ a_y\ a_z)^{\mathrm{T}}$	V
MotionPak	Angular velocity	$\boldsymbol{\Omega} = (\Omega_x\ \Omega_y\ \Omega_z)^{\mathrm{T}}$	V
MotionPak	Temperature	T_m	V
TCM2	Inclinometer roll	R	°
TCM2	Inclinometer pitch	P	°
TCM2	Compass bearing	ψ	°
SensorTechnics	Pressure	d	V
TCM2	Magnetometer	$\boldsymbol{B} = (B_x\ B_y\ B_z)^{\mathrm{T}}$	μT

Table A1.3: Definition of geometric & hydrodynamic parameter symbols. These parameters must be determined (experimentally or otherwise) for implementation of the AUV model.

Parameter	Symbol	Units	Frame
Mass	m	kg	–
Volume	V	m^3	–
Inertia tensor about centre of mass	$\boldsymbol{I_0} = \begin{bmatrix} I_x & 0 & -I_{xz} \\ 0 & I_y & 0 \\ -I_{xz} & 0 & I_z \end{bmatrix} \cong \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix}$	$kg \cdot m^4$	$\{B\}$
Accelerometer position vector	$\boldsymbol{r_a} = (x_a\ y_a\ z_a)$	m	$\{B\}$
Buoyancy vector	$\boldsymbol{r_b} = (x_b\ y_b\ z_b)$	m	$\{B\}$
Compass position vector	$\boldsymbol{r_c} = (x_c\ y_c\ z_c)$	m	$\{B\}$
Depth sensor position vector	$\boldsymbol{r_d} = (x_d\ y_d\ z_d)$	m	$\{B\}$
Thruster position / orientation matrix	$\boldsymbol{L} = \begin{bmatrix} \boldsymbol{L_1} \\ \boldsymbol{L_2} \end{bmatrix}$	$\boldsymbol{L_2}$ in m	$\{B\}$
Added mass matrix	$\boldsymbol{M_A} \cong \begin{bmatrix} \boldsymbol{M}_{A11} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{M}_{A22} \end{bmatrix}$	$\boldsymbol{M}_{A11}$ in kg $\boldsymbol{M}_{A22}$ in $kg \cdot m^4$	$\{B\}$
Total mass matrix	$\boldsymbol{M} \cong \begin{bmatrix} \boldsymbol{M}_{11} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{M}_{22} \end{bmatrix}$	$\boldsymbol{M}_{11}$ in kg $\boldsymbol{M}_{22}$ in $kg \cdot m^4$	$\{B\}$
Drag matrix	$\boldsymbol{D} \cong \begin{bmatrix} \boldsymbol{D}_{11} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{D}_{22} \end{bmatrix}$	$\boldsymbol{D}_{11}$ in $N \cdot m^{-2} s^2$ $\boldsymbol{D}_{22}$ in $N \cdot m \cdot rad^{-2} s^2$	$\{B\}$

Table A1.4: Miscellaneous symbols.

Parameter	Symbol	Units	Frame
Fluid (water) density	ρ	$kg \cdot m^{-3}$	–
Filter sampling period	T_S	s	–
Rotation matrix from $\{I\}$ to $\{B\}$ coordinates	$\boldsymbol{R}(\boldsymbol{q})$	–	–
Transformation matrix from $\boldsymbol{\omega}$ to $\dot{\boldsymbol{q}}$	$\boldsymbol{U}(\boldsymbol{q})$	–	–
Thruster forces	$\boldsymbol{u} = (F_1\ F_2\ F_3\ F_4\ F_5)^{\mathrm{T}}$	N	$\{B\}$

9.2 APPENDIX 2 – Thruster Model

Silpa-Anon [10] performed experimentation on the five thrusters of the Kambara so as to develop a thruster model. This model describes the force generated by each thruster as a function of shaft velocity. The shaft velocity has, in turn, been described as a function of the voltage across and current drained by the electric motor of the thruster.

The governing equation for the thrust force F_i under steady state was shown to be

$$F_i = k|\Omega_i|\Omega_i, \qquad \textbf{(A2.1)}$$

where Ω_i is the shaft velocity of thruster i. The constant of proportionality k was found to differ significantly between the forward and reverse directions. In the forward direction k was found to be $(4.4\pm0.2)\text{N}\cdot\text{sec}^2\text{rad}^{-2}$ over all thrusters, while in the reverse direction k was $(3.15\pm0.1)\text{N}\cdot\text{sec}^2\text{rad}^{-2}$. This difference in forward and reverse k values was to be expected, as thrusters are usually optimised for forward motion.

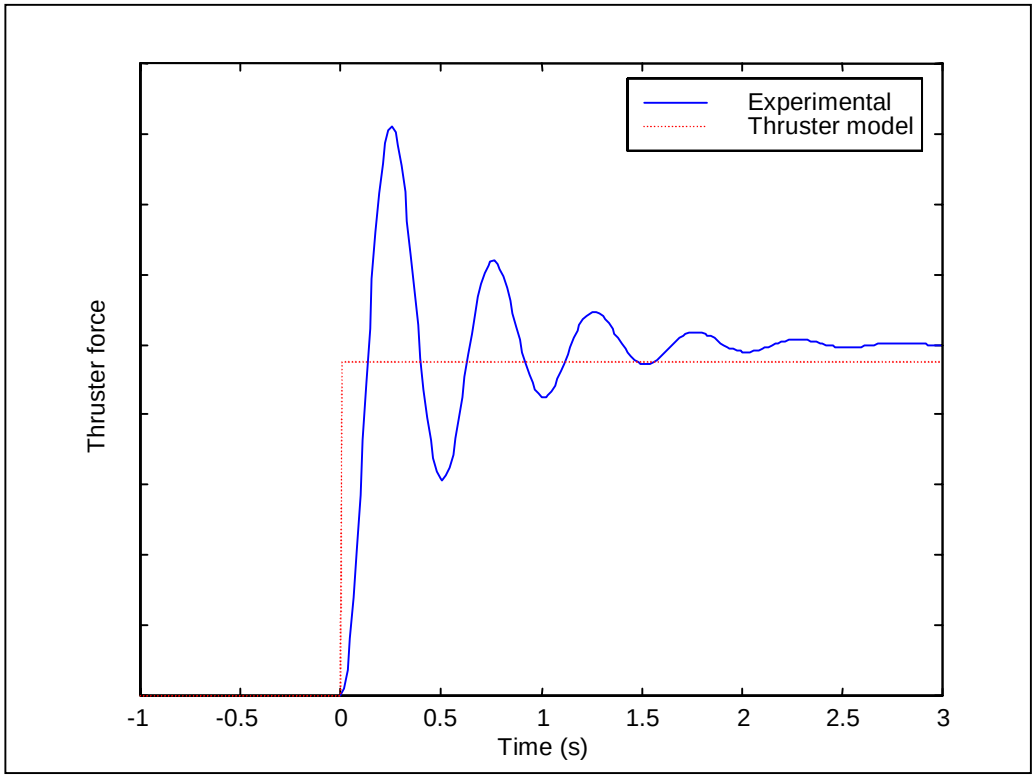


Figure A2.1: Typical thruster step response. Both experimental data and the thruster model have been graphed against time.

The typical features associated with fitting equation (A2.1) to experiment data are shown in figure A2.1 above. Although significant oscillation was found in the measured response of the motor, this was probably due to the vibration in the thrust measurement method. The actual response is expected to possess a much smaller overshoot and settling time. Once these

transients had died out of the experimental data, the thruster model was found to predict the thruster force to within 5%.

The equation for the shaft velocity Ω_i was shown to be

$$\Omega_i = \frac{V_i - i_i R_w}{K_{emf}}, \qquad \textbf{(A2.2)}$$

where V_i and i_i are the voltage across and current drained by thruster i, R_w is the winding resistance (0.208Ω), and K_{emf} is the motor back emf constant. K_{emf} varies between 0.124 and 0.138 sec·rad^{-1}V^{-1} over the five thrusters.

Equation (A2.2) too makes steady state assumptions, as it ignores the motor winding inductance. The consequence of this will be poor estimates of motor thrust for when the power supplied to the thrusters changes rapidly.

9.3 APPENDIX 3 – Sensor Sensitivity and Bias Data

Table A4.1: Sensitivity and zero point bias information gained from sensor data sheets. MotionPak bias readings were taken at 22°C and Summit Instruments accelerometer bias readings at 24°C.

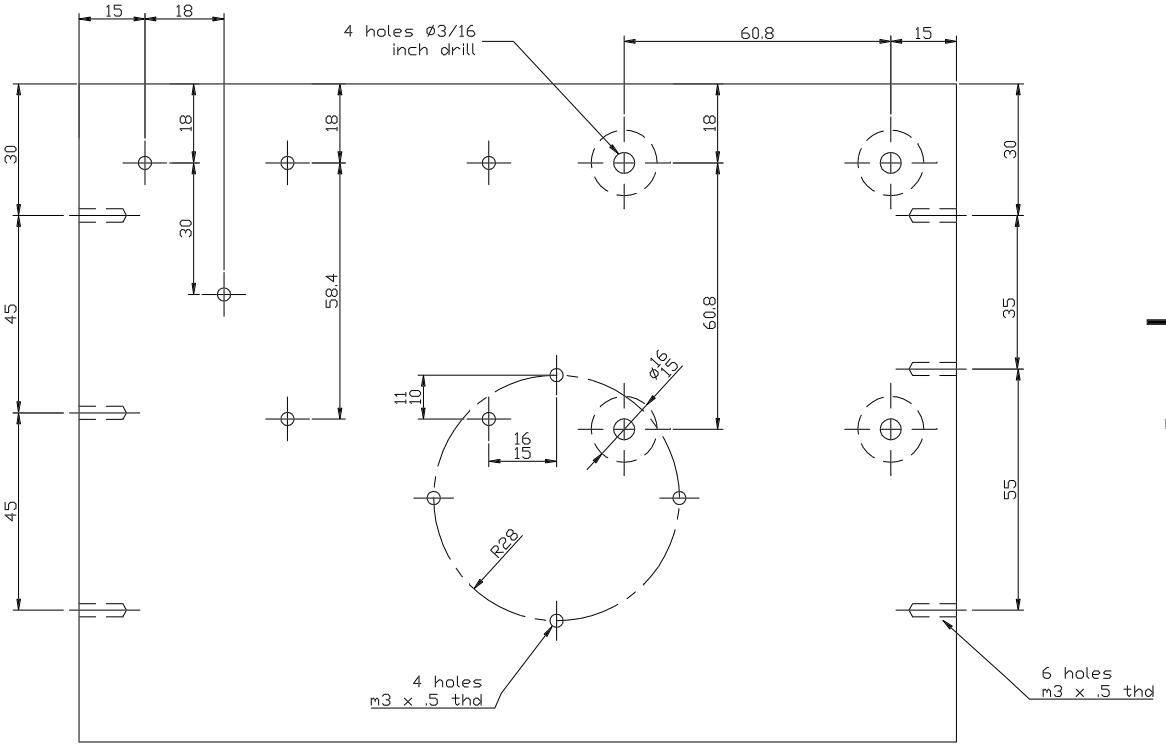
Sensor name	Sensitivity	Bias k_i	
MotionPak rate gyroscope			
x-axis	24.804 mV/°/s	0.10 °/s	2.48 mV
y-axis	24.922 mV/°/s	0.01 °/s	0.25 mV
z-axis	24.943 mV/°/s	0.07 °/s	1.75 mV
MotionPak accelerometer			
x-axis	3.753 V/g	–2.52 mg	9.46 mV
y-axis	3.747 V/g	3.76 mg	14.09 mV
z-axis	3.760 V/g	–1.93 mg	-7.26 mV
MotionPak temperature	1 μA/°C		
Summit Accelerometer			
x-axis	1.421 V/g	1.778 g	2526.4 mV
y-axis	1.424 V/g	1.723 g	2454.0 mV
z-axis	1.433 V/g	1.779 g	2549.2 mV
Pressure	1.42* V/m	0.725 m	1.03 V

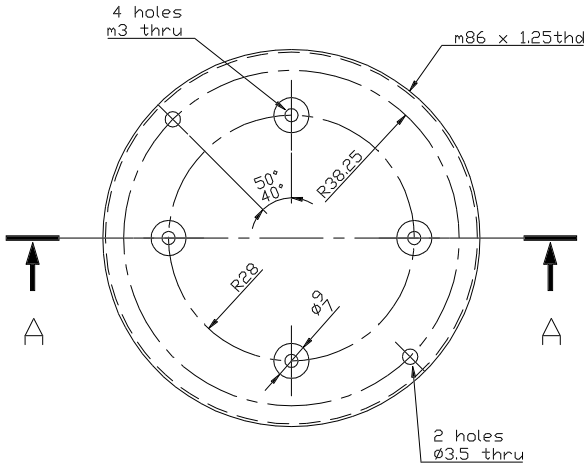
*taken from Beswick [2]

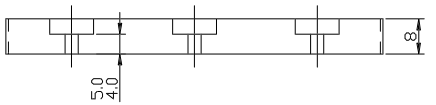
9.4 APPENDIX 4 – Sensor Mounting Drawings

The design of the sensor mounting plate and WAM arm adaptor plate is shown in the AutoCAD technical drawing over the page.

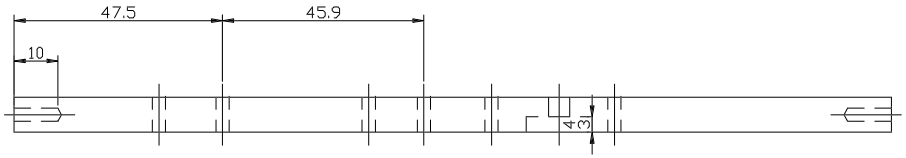
ZONE	REV	Sensor mounting and adaptor plate	15/7/99







Section A-A



Notes:

Unless otherwise stated, all holes are m3 x .5 thread

Drawn by Terence Betlehem	Sensor mounting and adaptor plate	
	SIZE A4 / FSCM NO. / DWG NO. 001-1999 / REV	
	SCALE 1:1	SHEET 1 of 1

9.5 APPENDIX 5 – Jacobian Matrix Matlab Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% jacobians.m
%
% This file calculates the jacobians required for the dynamic model.
%

%%%%%%%%%% Process Equation %%%%%%%%%%

syms Tt real;                   % sampling period

% State variable parameters
syms u v w p q r real;          % linear & angular velocities
syms x y z real;                % world coordinates of centre of mass
syms eta e1 e2 e3 real;         % quaternion components
syms T;                         % temperature
V = [u v w]';                   % linear velocity vector
W = [p q r]';                   % angular velocity vector
X = [x y z]';                   % centre of mass vector
Q = [eta e1 e2 e3]';            % quaternion vector
e = [e1 e2 e3]';                % imaginary quaternion part

% Control inputs
syms F1 F2 F3 F4 F5 real;       % thruster forces
F = [F1 F2 F3 F4 F5]';

% Body parameters
syms m Ix Iy Iz Ixz real;       % inertia tensor components
syms g B xB yB zB real;         % buoyancy vector components
syms m g B real;                % mass, grav. accel. & buoyancy force
syms l1 l2 l3 l4 l5 real;       % thruster positions
I0 = [Ix 0 -0; 0 Iy 0; -0 0 Iz];    % inertia tensor
M0 = [m*eye(3) zeros(3,3); zeros(3,3) I0];  % mass/inertia matrix
rB = [xB yB zB]';               % buoyancy vector
L = [1 1 0 0 0;                 % thrust to force/moment matrix
   0 0 0 0 0;
   0 0 -1 -1 -1;
   0 0 l1 -l1 0;
   -l5 -l5 l3 l3 -l4;
   l2 -l2 0 0 0];

% Hydrodynamic parameters
syms m_A1 m_A2 m_A3 i_A1 i_A2 i_A3 real;   % added mass components
syms d1 d2 d3 d4 d5 d6 real;               % drag matrix components
```

```
MA = diag([m_A1 m_A2 m_A3 i_A1 i_A2 i_A3]); % added mass matrix
M = M0 + MA;                                % total mass matrix
D = diag([d1 d2 d3 d4 d5 d6]);              % drag matrix


% Transformation matrices
Rot = eye(3) + 2 * eta * skew(e) + 2 * skew(e)^2; % rotation matrix
U = [ -e'; eta*eye(3) + skew(e)];           % converts W to 2*Qdot


% Rigid body dynamics
C = [ zeros(3,3) -skew(M(1:3,1:3)*V)
     -skew(M(1:3,1:3)*V) -skew(M(4:6,4:6)*W) ];     % gyroscopic acceleration
fG = Rot'*[0; 0; m*g];                      % gravitational force
fB = Rot'*[0; 0; B];                        % buoyancy force
G = [ fG-fB; -skew(rB)*fB ];                % net grav/buoy force/moment


% Final equations
Vdot = inv(M(1:3,1:3))*(L(1:3,:)*F + skew(M(1:3,1:3)*V)*W - diag(D(1:3,1:3)*abs(V))*V + G(1:3));
Wdot = inv(M(4:6,4:6))*(L(4:6,:)*F + skew(M(1:3,1:3)*V)*V + skew(M(4:6,4:6)*W)*W -
diag(D(4:6,4:6)*abs(W))*W + G(4:6));
Xdot = Rot * V;
Qdot = U/2 * W;

% Calculate jacobians of discretised system
A = jacobian([V; W; X; Q; T] + Tt*[Vdot; Wdot; Xdot; Qdot; 0], [V; W; X; Q; T])



%%%%%%%%%% Basic Measurement Equation %%%%%%%%%%

% Measurement variables
syms ax ay az real;                 % accelerometer readings
syms Ox Oy Oz real;                 % rate gyro readings
syms P R phi real;                  % compass readings
syms p real;                        % pressure reading
syms Tm real;                       % temperature reading
a = [ax ay az];
O = [Ox Oy Oz];

% Sensor parameters
syms Sax Say Saz real;              % accelerometer sensitivities
syms Sox Soy Soz real;              % rate gyro sensitivities
syms Sd St real;                    % depth and temp sensor sensitivities
syms kax kay kaz real;              % accelerometer temp drift rates
syms kox koy koz real;              % rate gyro temp drift rates
syms kd kt real;                    % depth and temp sensor biases
```

```
syms bax bay baz real;              % accelerometer temp biases
syms box boy boz real;              % rate gyro temp biases
syms bd bt real;                    % depth and temp sensor biases
Sa = diag([Sax Say Saz]);
So = diag([Sox Soy Soz]);
ka = [kax kay kaz]';
ko = [kox koy koz]';
ba = [bax bay baz]';
bo = [box boy boz]';

% Sensor positions
syms xA yA zA real;                 % accelerometer position in {B}
syms xC yC zC real;                 % compass position in {B}
syms xD yD zD real;                 % depth sensor position in {B}
rA = [xA yA zA]';
rD = [xD yD zD]';
rC = [xC yC zC]';

% Compass equations
P = atan((e1*e3-eta*e2)/(1/2-eta^2-e3^2));
R = atan((e2*e3+eta*e1)/(1/2-eta^2-e3^2));
phi = atan((e1*e2+eta*e3)/(1/2-eta^2-e3^2));

% Accelerometer equations
a = Sa*(Vdot - skew(rA)*Wdot + skew(W)^2*rA - Rot'*[0 0 g]') + ka*T + ba;

% Rate gyro equations
omega = So*W + ko*T + bo;

% Depth sensor equation
d = -Sd * (z + rD'*Rot*[0 0 1]') + bd;

% Temperature sensor equation
Tm = T;

H_basic = jacobian([a; omega; P; R; phi; d; Tm], [V; W; X; Q; T]);



%%%%%%%%%% Improved Measurement Equation %%%%%%%%%%

syms Bx By Bz;                  % magnetometer readings
syms Bex Bey Bez;               % earth's magnetic field
syms Blx Bly Blz;               % local fields
syms alphax alphaz;             % shielding matrix parameters
B = [Bx By Bz]';
Be = [Bex Bey Bez]';
Bl = [Blx Bly Blz]';
```

```
alpha = [alphax 0 0; 0 alphax 0; 0 0 alphaz];

P = atan(a(1)/a(3));
R = atan(a(2)/a(3));

B = Rot' * alpha * Be + Bl;

H_improved = jacobian([a; omega; P; R; B; d; Tm], [V; W; X; Q; T; alphax; alphaz]);
```

9.6 APPENDIX 6 – Original Project Proposal

9.6.1 Abstract

Over the next 32 weeks, a state estimation system is to be implemented on a submersible robot. A system of sensors, models and algorithms will be used to estimate current and future state (ie. position, orientation and velocity) of the robot. Such state estimates will be used by the robot for navigation purposes. To accomplish this, work will need to be done in testing and installing sensors, refining a previously developed dynamic model[1], choosing and implementing a filtering algorithm and in the software development of device drivers under a real-time operating system.

A draft project schedule is provided in table 1. It is hoped that, by the end of the project, the state estimation system will have been experimentally proven to accurately predict the state of the submersible robot.

9.6.2 Background

For the past year, the Research School of Information Sciences and Engineering (RSISE), Australian National University (ANU), has been working on the design and implementation of an underwater vehicle for use in marine environments.

The major feature of this vehicle is that it is autonomous. This is to say that instead of having its every movement controlled by a human, only occasional supervisory commands need be provided, such as ‘hold current position’ and ‘swim after target’.

The aim of constructing such an autonomous underwater vehicle (AUV) is to allow further research work to be undertaken on the problems of the control of an underwater vehicle and in vision systems. Eventually, it is hoped that such robotic technology could be used to undertake monitoring, inspection and maintenance tasks in Australia’s vast coastal environment.

The AUV has been named Kambara, which is an aboriginal word for crocodile. The original submarine robot frame was purchased from the University of Sydney, who had already developed an AUV called Oberon. The Kambara differs from Oberon in that the central computer and power source are both on-board, and a vision system implemented. At the moment, communication with the outside world occurs along an optical fibre cable.

9.6.3 Introduction

Last year, Dale Beswick completed some work on the inertial navigation system (INS) of the Kambara[6]. The purpose of the INS was to estimate the present and future state, (ie. position, orientation and velocity) of the Kambara in real time. Dale's work involved the selection, acquisition and testing of several sensors used for the navigation system. The sensors chosen included a triaxial accelerometer, pressure/depth sensor, combined magnetic compass & biaxial inclinometer and a triaxial rate gyroscope. He derived a linear dynamic model of the Kambara and used it in a Kalman filter. Kalman filtering provides an algorithm which converts the raw sensor data and what we know about the dynamics of the AUV into state estimates (of both present and future). Dale also did some basic simulation testing on the Kalman filter to show it was working.

My task, to a large extent, is to continue development of the state estimator. The dynamic model and filtering algorithms will have to be extended, the sensors further tested and calibrated, a device driver for an analog to digital converter (ADC) implemented, the sensors integrated into the Kambara, and the whole system tested.

9.6.4 Project Proposal

Implementation of the state estimation system can be broken down into the following major areas:

9.6.4.1 Sensor Testing, Calibration & Characterisation

Dale performed thorough testing on the depth sensor, but minimal testing on the compass/inclinometer. Furthermore, neither the triaxial accelerometer nor the triaxial rate gyroscope have been examined. Accuracy, sensitivity, drift error and other tests must be performed on these three sensors. Careful note of the triaxial accelerometer's resolution should be made, as it is suspected that this may not be fine enough. Temperature effects should be investigated. Where appropriate, sensors should be calibrated.

Furthermore, the operation of the sensors in the AUV environment needs to be investigated. In particular, the magnetic fields produced by thruster operation are anticipated to significantly degrade the accuracy of the magnetic compass.

Eventually, the sensors must be tested in a complete package. Such testing will allow the sensor-sensor interactions to be gauged, and allow evaluation of the accuracy of the INS.

---

[6] D.Beswick, *Inertial Navigation System for an Autonomous Underwater Vehicle*, Research School of Information Sciences and Engineering, Australian National University, 1998.

9.6.4.2 Sensor - Software Interface Implementation

The sensor signals are fed into the computer system through ADC's. A device driver must be written under VxWorks to interface the ADC card with the PowerPC. The driver will be based on code written by Bob Edwards for a similar ADC card. Such a driver will facilitate accurate measurement of sensor signal values, and hence aid sensor testing.

9.6.4.3 Dynamic Model Development

At the moment the dynamic model ignores fluid forces and models the thrusters as point forces. The dynamic model of the AUV used in the filtering algorithm should be extended to cover non-linear effects such as hydrodynamic forces. The thrusters will have to be modelled so that the output thrust can be determined as a function of input motor current and thruster velocity. This work will be undertaken in cooperation with Chanop Silpa-Anan.

9.6.4.4 Filter Design and Implementation

To cope with the non-linear terms introduced into the dynamic model, an Extended Kalman filter will have to be designed and take the place of the current Kalman filter. Consideration of other possible filtering algorithms would be useful. The filtering algorithm must be designed to utilise all of the information outputted by the above-mentioned sensors. The final filter will be implemented in C++ on a PowerPC running VxWorks.

9.6.4.5 Sensor Integration

Suitable mountings for the sensors must be investigated, designed and implemented. The sensors should be easily attached and detached, and placed in appropriate locations that minimise interference from other devices in the AUV. An option is to create a board onto which all sensors are mounted. The electrical wiring of the sensors must at least be checked.

9.6.4.6 Underwater System Testing

Half way through the year, full immersion testing of the AUV will commence. This will allow experimental testing of the filtering algorithm to make sure the prediction/estimation algorithm works correctly and accurately in an underwater environment. In particular, the accuracy of the thruster and AUV physical models will be determined. The results of these experiments may in turn lead to further refinement in the models.

9.6.5 Project Schedule

As other people in the Kambara team are assigned to work on different subsystems of the AUV, scheduling our work will be very important. With the help of the Kambara team project schedule, the draft schedule in table 1 below has been prepared.

Table 1: Project schedule. Boldfaced dates represent assessment submission dates.

	March	April	May	June	July	Aug	Sep	Oct
Finalise project specifications	15..19							
Submit project proposal	**19**							
Develop dynamic model of vehicle	15				30			
Develop dynamic model of thruster	15		30					
Test & calibrate sensors	22			30				
Implement sensors on AUV		1		31				
Code state estimation algorithm			1		31			
Write progress report					5…19			
Submit progress report					**19**			
Prepare seminar					20..26			
Hold seminar					**26..28**			
Test inertial navigation					15		30	
Write thesis				30	25		1	27
Submit draft thesis								**5**
Submit thesis & notebook								**27**

9.6.6 Conclusion

This project is highly interdisciplinary. It will allow me to utilise a wide range of skills. I will gain experience in real-time device interfaces, filtering algorithms, hardware assemblies, and in testing and refining a state estimation system. Not to mention device driver coding in C++ and dynamic model development incorporating some fluid mechanics. However, the emphasis shall be on real-time sensor filtering and state estimation of underwater vehicles.

9.6.7 References

[1] D.Beswick, *Inertial Navigation System for an Autonomous Underwater Vehicle*, RSISE, ANU, 1998.