Major Project Report

MTRX4700

Paul Phillips	308247302
Jack Umenberger	309201063
Anders Ridley-Smith	309228093

The Calibration of an IMU using a Robotic Arm and the Subsequent Teleoperation of the Arm using the IMU

Jack Umenberger, Anders Ridley-Smith and Paul Phillips

List of Notation

x_k	State vector at time *k*.	s_ϕ, c_ϕ, t_ϕ	$\sin\phi$, $\cos\phi$, $\tan\phi$
u_k	Control input vector at time *k*	$I_{3\times3}$	3x3 Identity matrix
w_k	Process model noise vector at time *k*	$0_{3\times3}$	3x3 Zero matrix
ϕ, θ, ψ	Roll, pitch, yaw	Δt	Time step
p, q, r	Rotation rates in body fixed axes	$M_{6\times6}$	Jacobian of *time update* function
ϑ	State vector of Euler angles	x_k^-, x_k^+	*A priori/posteriori* state vector
ω	State vector of body rotation rates	P_k^-, P_k^+	*A prior/posteriori* error covariance matrix
$W_{3\times3}$	Matrix to convert from body rates to Euler rates	Q	Process noise covariance matrix
v_k	Measurement noise vector at time *k*	R	Measurement noise covariance matrix
K_k	Kalman gain matrix	H	Observation model matrix
z_k	Observation vector at time *k*		

***Abstract*- A two-fold experimental investigation is conducted using an Epson Robotic arm, paired with an affordable inertial measurement unit (IMU). As a first step, the robotic arm is used as a means of calibrating the tri-axis gyroscope, accelerometers and compass as well as establishing an estimate for the covariance matrix to be used in the Kalman filter. To confirm the effectiveness of the calibration procedure, the same IMU is subsequently used to perform teleoperation between the user and the robotic arm. Two Euler angles, along with a magnetic heading reference are used to manoeuvre the end effector in Cartesian space.**

Introduction

Inertial Measurement Units (IMU) have been used in navigational systems since the early 1940's and today play an important role in commercial, military and scientific applications [1]. Strapdown IMU's, like the one used here are readily available at a very affordable price. This provides incentive for their use when compared to high-end inertial navigation units (INS) where much of the filtering is done on board [2] [3]. All IMU's are subject to some degree of error in their readings, these result in accumulative displacement errors when integrated and therefore, in standalone IMU applications, it is desirable to make use of accelerations and turn rates directly rather than their corresponding positions and heading estimates. To further enhance the accuracy of the systems state estimate the readings acquired from the IMU can be filtered. Although there is many filter types available, a Kalman filter is widely accepted to be the most suitable for this application [4] [1]. To accurately calibrate an IMU it is essential to know exactly what its state should be at a given time. In calibrating the accelerometers this is somewhat trivial as the gravity vector can be used to establish an absolute reference. The same is true in the case of the magnetometer. Determining the exact turn rate of a gyroscope poses as a slightly less intuitive problem.

Precisions Robotic arms, like the Epson C-3, are used in industry to perform tasks that are either to dangerous for humans, repetitive or require a high degree of accuracy [5] [6]. Encoders and small increment stepper motors, built into each joint provide the C-3 with a great accuracy in establishing not only its position but also its end effectors velocity and acceleration. This provides a reliable platform from which to calibrate the gyroscopic turn rates of the IMU.

Advances in computational power have resulted in robots that are able to perform relatively complex tasks however; intelligence and adaptability still favour humans [7]. An intuitive ways of teleoperating a robotic manipulator has therefore been the goal of many academic and commercial ventures [8]. The utilised methods for establishing a master-slave relationship between human and robot vary from haptic suites to visions based systems [7] [9]. The difficulty is in designing an interface that is both intuitive yet does not obstruct the operator's movements. The ability of an IMU to sense motion without the added complexity of moving parts or bulky sensors makes it a primary candidate for the use in human-robot teleoperation [10].

Experimental Setup/Design

The system hardware and software can be broadly segmented into the components shown in **Error! Reference source not found.**.

Handheld Device

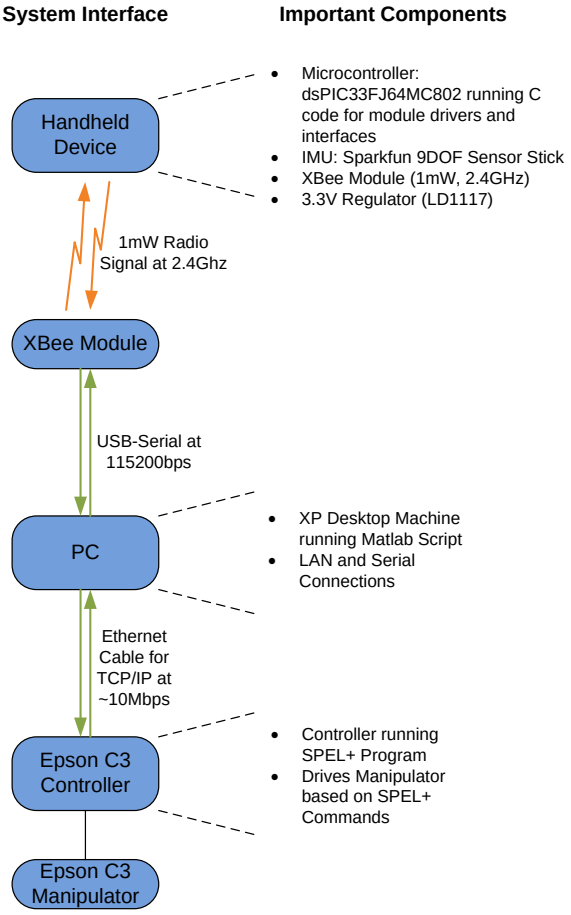

Figure 1: System Hardware Interface

The handheld device is pictured in **Figure 2** and consists of a PIC microcontroller, a 3.3V regulator, an XBee wireless transceiver and a Sparkfun 9DOF IMU Sensor Stick. The driving principle of the hand held device was the emphasis placed on form-factor and portability. This removed possibilities such as a using wired RS232 connection and necessitated wireless communication such as that offered by ZigBee or Bluetooth protocols. An additional consideration was that the device needed to allow for precise and orthogonal mounting of the IMU with respect to the mount on the robot arm for purposes of calibration. In other words, the IMU could not merely be soldered straight to the circuit bored with the hope that the resulting orientation of the IMU was precisely "flat" enough. To address this, the IMU was mounted flush against one side of the box that held the remaining circuitry. After prototyping the circuit and testing it on a breadboard, the final circuit was implemented on veroboard for cost reasons and can be seen in **Figure 3**. As a final design consideration, it was decided that the handheld device should function primarily as a "dumb" device and, besides other core functionality, simple receive data from the IMU and transmit this back to the PC terminal.

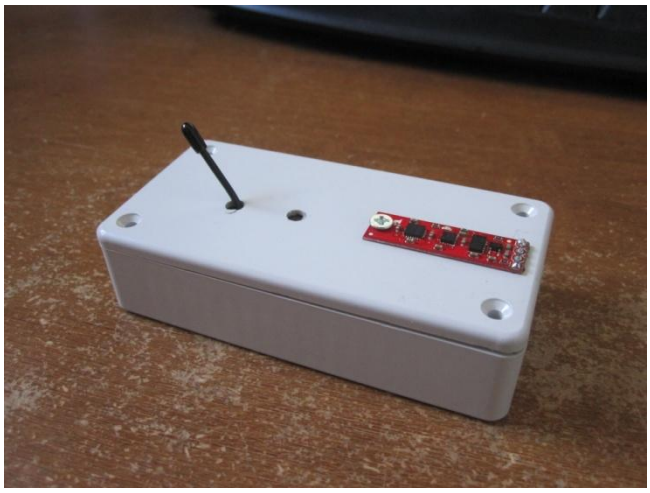

Figure 2: Handheld Device

The Sparkfun IMU is called a 9DOF (Degrees of Freedom) sensor stick due to the fact that it contains a tri-axial accelerometer, gyroscope and magnetometer. Individually, these are the ADXL345, the ITG3200 and the HMC5402 respectively. Attitude estimation theory makes clear that a single orientation sensor such as an accelerometer is

inadequate to fully resolve orientation in terms of the three Euler angles or similar. Instead, at least two vector observations are needed and so the measurement of the gravity vector from an accelerometer must be combined with the measurement of another vector from an alternative sensor. The choice of the method to obtain additional attitude vectors varies depending on the field of work. Spacecraft often use star sensors to compare with a defined star map and produce attitude vectors, medical apparatus often assess human head orientation via vision based systems. Here, we use the simple magnetometer on the sensor stick to provide an additional vector.

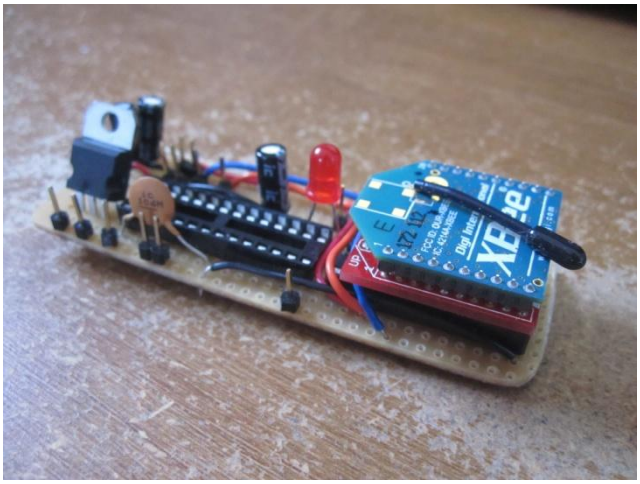

Figure 3: Veroboard Circuitry

As we are operating in an indoor environment, the accuracy of the magnetometer is diminished and it is more susceptible to noise from surrounding metallic and electrical objects. Already a low-cost commercial -off-the-shelf (COTS) component, this does not help the magnetometer data readings.

The communication to the PC terminal was done through the XBee component of the device and achieved satisfactory results. Upon initial testing, the XBee worked well over the range that tests were carried out (0-10m). Towards completion of the project the antenna on the XBee became loose and transmission strength and package reception rates dropped significantly. After soldering the antenna back in place the signal improved, but was still significantly limited and restricted use of the device to within 1-2m of the PC receiving terminal.

The code that runs on the PIC microcontroller is entirely self-written and no external libraries were used. Driver functionality for UART, I2C and timer modules was written as were interfaces for the ADXL, ITG and HMC sensors. In the main code, program flow is structured so that received data on the UART (via XBee) and calls to transmit data trigger interrupts (off different priority). In terms of state, the PIC code monitors for received data and parses it to determine the required state to enter or function to execute. In this way, the PIC can be instructed to carry out initialisation routines and collect and transmit data from the IMU completely via UART commands. The significance of this is that no external controls on the handheld device are required and thus the user interface is profoundly simple. This decision to not have any buttons on the handheld device was also made while thinking ahead to the eventual manual control of the robot manipulator arm. In manual control mode it was, and is, desirable to achieve meaningful motion purely from inertial movements of the device and to derive no movement commands from buttons, potentiometers or other traditional interface methods.

PC

The PC terminal operated in-between the handheld device and the Epson C3 Controller. Its only additional hardware was an XBee module Figure **3** for wireless communication with the handheld device (operating through a USB-to-serial module) and an Ethernet cable providing TCP/IP communication with the Epson Controller.

The PC runs a Matlab script that (wait to see if anyone else has done already).

Robot Arm

The arm used is an industrial grade C-3 Epson Robot arm. It has six degrees of freedom made possible by six rotational joints has shown in **Figure 4**. A flat, stable platform, made up from a piece of right angle aluminium, was mounted on the end to serve as a holder for the handheld device during the calibration routine.

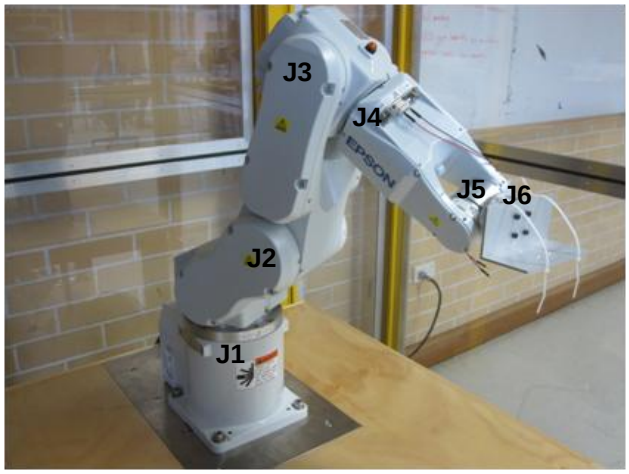

Figure 4: Epson C3 Robot Manipulator Arm with End Assembly

Filtering

For the remote operation of the robotic arm, it is necessary to develop a regime that intuitively translates the motion of the handheld control device into control commands that manipulate the position of the end effecter; this clearly requires some notion of the state or pose of the handheld control device. Initial ideas proposed the tracking of the inertial position of the hand-held device. This approach was dismissed as the only means of estimating the displacement of the device is to perform dead-reckoning on the output of the low-cost, micro-electro-mechanical (MEMS) accelerometers; with no means of measuring the absolute position of the device, these dead-reckoned estimates would rapidly diverge from the true position. Given the infeasibility of inertial position tracking, it was decided that the control regime should be based on the orientation of the handheld control device. Changes in orientation can be estimated from the output of the rate gyroscope, which can then be corrected by absolute measures of attitude obtained from the accelerometers and magnetometer to address the problem of dead-reckoning drift. The natural way to combine the orientation estimates from the inertial and magnetic sensors is the Kalman filter, which is an efficient algorithm to recursively solve the weighted least squares estimate of the state of a system. Considering this statement carefully, the Kalman filter algorithm:

- Produces an estimate of a state of the system that minimises the mean of the square of the error, as with the familiar least squares method of Gauss.
- Incorporates some notion of the quality or trustworthiness of both the measured observations and the system model, hence the term *weighted* least squares solution.
- Performs this operation *recursively* using only the most recent system state estimate and observations, rather than retaining a large *batch* of all previously acquired data; this accounts for the computational efficiency of the algorithm.

The equations comprising the Kalman filter algorithm have been both applied and described extensively in the literature, and shall not here be derived. Rather, the following shall be concerned with implementation, with particular attention to the model on which the algorithm operates and the tuning of the Kalman filter error covariance matrices. A high level overview of the Kalman filter implemented in depicted in [11]

Kalman Filter Implementation

The objective of the original Kalman filter algorithm, is to estimate the state of a process governed by a *linear stochastic difference equation*,[11]

Equation 1

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}$$

Considering this statement carefully, with reference to the above equation, we elaborate on the properties of such a system:

- The equation must be linear, which implies that *A* and *B* can be represented by matrices.
- The process is stochastic rather than deterministic, due to the inclusion of the so called *process* noise random variable w_{k-1}, which is normally distributed with zero mean.
- The equation is a discrete difference equation that describes the evolution of the system from the current state to the future state. For

this reason it is often referred to as the *time update* or *prediction* equation.

The first task in the implementation of the Kalman filter is the definition of the system states and the development of the linear difference equation to govern system evolution. For this application, we are interested in the orientation, which we choose to represent by Euler angles despite the singularities that may arise. The body fixed rates of rotation are also included in the state vector,

$$x = [\phi \quad \theta \quad \psi \quad p \quad q \quad r]^T$$

It is convenient to divide the state vector into two sub-state vectors as follows,

$$\vartheta = [\phi \quad \theta \quad \psi]^T$$

$$\omega = [p \quad q \quad r]^T$$

In developing the difference equation, we ignore the terms associated with the control inputs. The control inputs are entirely subject to the volition of the user and are, as a consequence, unknown; In other words, we have no reasonable model of the expected behaviour of the user. The *time update* equation describing the change in attitude is based entirely on the estimated rate of change of the Euler angles, integrated over a finite time period. The *time update* equation(s) are as follows:

The angular rotation rates measured in the body frame by the gyroscope must first be converted into rates of change of the Euler angles,

Equation 2

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} = [W_{3\times3}(\vartheta)]\begin{bmatrix}p\\ q\\ r\end{bmatrix} = \begin{bmatrix}1 & s_\phi t_\theta & c_\phi t_\theta\\ 0 & c_\phi & -s_\phi\\ 0 & \frac{s_\phi}{c_\theta} & \frac{c_\phi}{c_\theta}\end{bmatrix}\begin{bmatrix}p\\ q\\ r\end{bmatrix}$$

Then by employing a simple first order Taylor series approximation, we can write the update equation as follows,

Equation 3

$$x_k^- = \begin{bmatrix}I_{3\times3} & W_{3\times3}\Delta t\\ 0_{3\times3} & I_{3\times3}\end{bmatrix}x_{k-1}^+$$

This simply states that the new *a priori* Euler angle estimates are equal to the old *a posteriori* estimate plus the product of the rate of change of the Euler angles with the time since the last estimate. The equation also implies that the body rates remain constant from one state to the next. Although the above *time update* equation is represented as a matrix product, the state transition is in fact governed by a non-linear relationship; $W_{3\times3}(x)$ is a non-linear function of the state variables. Recall that the Kalman filter algorithm operates on a linear difference equation; to apply the standard Kalman filter equations, it is necessary to linearise this function about the *best guess* of the system state. The application of the standard Kalman filter equations to a non-linear system linearised about the current state estimate is referred to an *Extended Kalman Filter*. The update equation can be written in explicitly in terms of the sub-states, with the $f(\vartheta, \omega, \Delta t)$ notation used to emphasise the non-linear relationship.

Equation 4

$$\begin{bmatrix}\vartheta\\ \omega\end{bmatrix}_k = \begin{bmatrix}I_{3\times3} & W_{3\times3}\Delta t\\ 0_{3\times3} & I_{3\times3}\end{bmatrix}\begin{bmatrix}\vartheta\\ \omega\end{bmatrix}_{k-1}$$

$$\begin{bmatrix}\vartheta\\ \omega\end{bmatrix}_k = \begin{bmatrix}f(\vartheta, \omega, \Delta t)\\ \omega_{k-1}\end{bmatrix}$$

To linearise this system of equations we take the Jacobian of the state update function,

Equation 5

$$M_{6\times6} = \begin{bmatrix}\frac{\partial f}{\partial \vartheta} & \frac{\partial f}{\partial \omega}\\ \frac{\partial \omega}{\partial \vartheta} & \frac{\partial \omega}{\partial \omega}\end{bmatrix} = \begin{bmatrix}A_{3\times3} & B_{3\times3}\\ 0_{3\times3} & I_{3\times3}\end{bmatrix}$$

Where,

$$V_{3\times3} = \frac{\partial[W_{3\times3}\omega]}{\partial \vartheta}$$

$$A_{3\times3} = \frac{\partial f}{\partial \vartheta} = I_{3\times3} + V_{3\times3}\Delta t$$

$$B_{3\times3} = \frac{\partial f}{\partial \omega} = W_{3\times3}\Delta t$$

The linearised *time update* equation can then be used to update the new *a priori* estimate of the error

covariance, in accordance with the standard linear Kalman filter equations [11],

Equation 6

$$P_k^- = M_{6\times6} P_{k-1}^+ {M_{6\times6}}^T + Q$$

This completes the *prediction* stage of the Kalman filter; we must now consider the *measurement* or *correction* of the *a priori* estimate generated by the equations above. In addition to the *linear stochastic difference* equation, the Kalman filter also expects the system to be measurable, with observations related to the system state by the following *linear stochastic* equation [11],

Equation 7

$$z_k = Hx_k + v_k$$

In this case v_k represents the measurement noise, which is again modelled by a random variable with normal distribution and zero mean. For the system under consideration, H is simply the identity matrix, as all states are directly observable. The angular rates ω are measured directly from the gyroscope. The roll ϕ and pitch θ are estimated from the accelerometer data, by assuming that the measured acceleration can be entirely attributed to gravity, which implies that translational accelerations must be negligible. The yaw ψ is estimated by projecting the measured magnetic flux vector into the local horizontal plane, as defined by the current *a priori* estimate of the roll and pitch, and then computing the rotation of this vector about the local vertical axis. It should be noted that the measured magnetic flux vector shall point to magnetic North; we would need to consider the magnetic declination if the heading relative to true North were required.

After collecting the state observations from the inertial and magnetic sensor data, the Kalman gain may be computed

Equation 8

$$K_k = P_k^- H^T (H P_k^- H^T + R)^T$$

The Kalman gain may then be used to compute the *a posteriori* state estimate,

Equation 9

$$x_k^+ = x_k^- + K_k(z_k - Hx_k^-)$$

It should be noted that the *a posteriori* update was modified slightly to handle the yaw angle. Yaw was measured over $\pm\pi$ radians and so complications can arise at the $\pm\pi$ cross-over point. A simple series of check statements were used to ensure the yaw angle was updated correctly. Finally, the *a posteriori* error covariance matrix may be updated

Equation 10

$$P_k^+ = P_k^- - K_k H P_k^-$$

Results

Calibration

The objective of the calibration procedure is two-fold. The first objective is to correct the raw output from the inertial and magnetic sensors by determining the offset and gain factors. The second objective is to gain some notion of the quality of the data output, so as to furnish the Kalman filter algorithm with measurement noise covariance matrices. Before calibration of the sensors could be conducted, it was necessary to first calibrate the robotic arm to the local horizontal reference frame. This was accomplished by commanding the robotic arm to assume the various end effecter poses required for calibration, and then manually adjusting these positions in software with reference to a high precision digital spirit level until the desired sensor axis for each pose was correctly aligned with the local gravity vector.

The calibration routine consists of three ‘sweeps’ as depicted in Figure **5**; each sweep is proceeded by a period of time during which the end effecter is held in a static position, so the calibration sequence actually generates six distinct data sets, each corresponding to a different pattern of motion. The static end effecter positions are used to determine the accelerometer offsets, gyroscope offsets and accelerometer scale factors, while the sweeping motions are used to compute the gyroscope scale factors. Each of the three static poses aligns a different accelerometer axis with the local vertical. This vertical axis then measures the gravitational field which is used to calibrate the gain of the sensor;

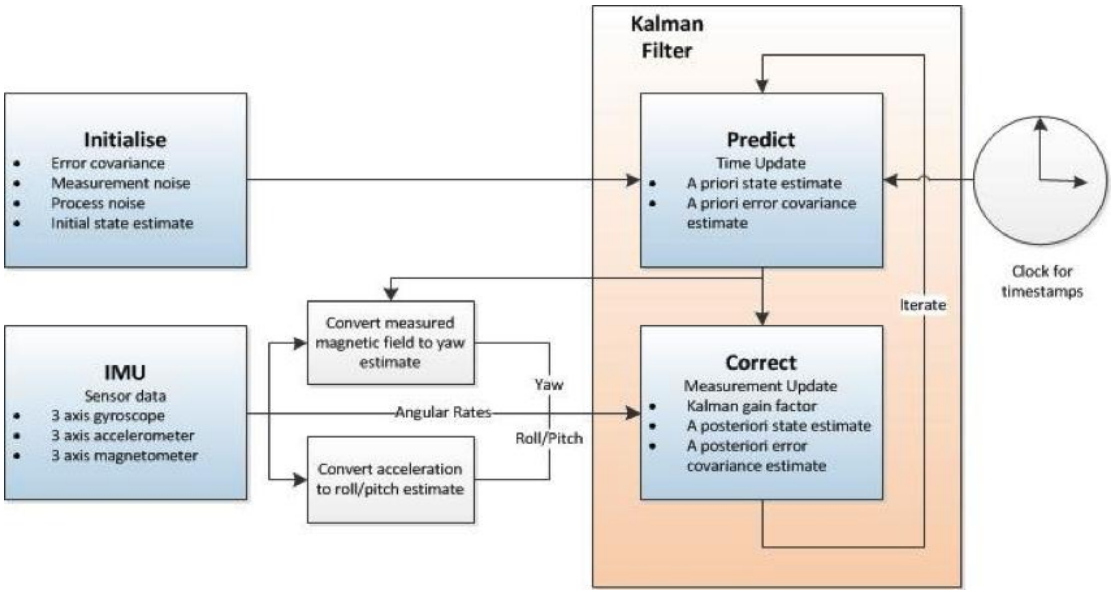

Figure 6 Kalman filter overview

the other two horizontal axes should ideally record no signal, and so the acceleration measured is used to determine the offset values of the accelerometer. The sweeping motion generates a rotation at a known angular rate; each sweep targets a different axis of the gyroscope to allow the gain factors to be calibrated. It is noted that the calibration routine assumes that the sensor output is linear, and can be accurately described by a single gain value and offset. There is potential for obtaining non-linear calibration functions by fitting second order polynomials to data collected from multiple sweeps at different angular rates, and translations at different constant acceleration rates.

The second objective of the calibration routine is to determine the measurement error covariance matrix R for the Kalman filter. The measurement noise covariance has been approximated by the sample variance of the *processed* system state observations. Considering this statement carefully for each state observation:

- Body rotation rates are directly observable by the gyroscope. The gyroscope is held still and the sample variance of the calibrated data output – converted by the gain factor to the units of rad/s – is computed.
- The roll and pitch angle are computed from accelerometer data. The accelerometer is held still, axes aligned with the local horizontal plane and the calibrated acceleration data is used to compute the pitch and roll angles. The sample variance is used to estimate the error covariance of each measurement. A few remarks are necessary:

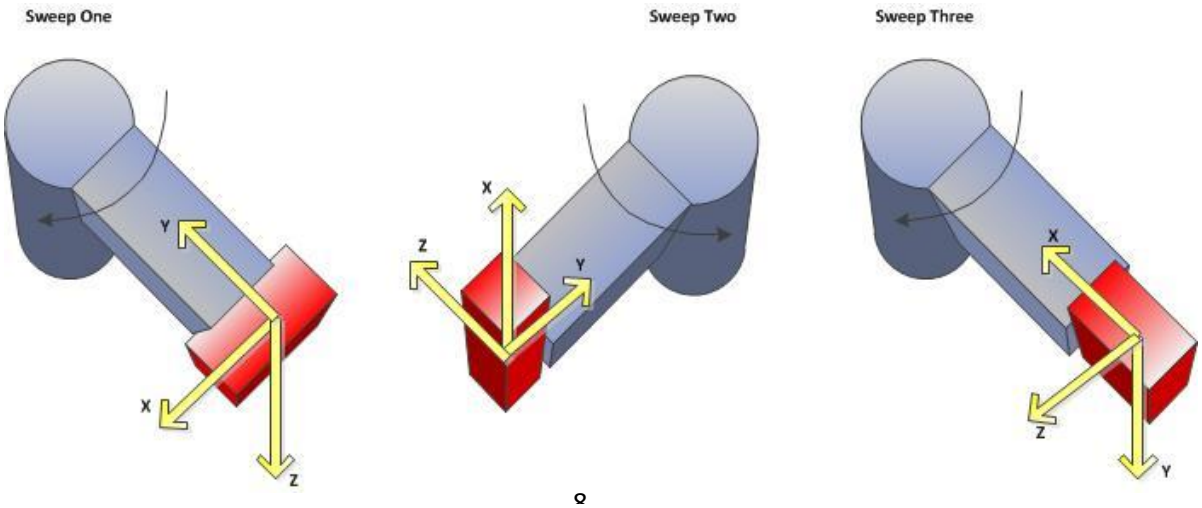

Figure 5 Calibration sequence

1) the actual error in the attitude estimates from the accelerometer will be dramatically affected by translational acceleration, not merely the sensor noise, which will not be normally distributed as expected by the Kalman filter, 2) there will be correlation between the roll and pitch errors, because they depend on the same acceleration measurements. These correlations have been ignored, and the covariance matrix is diagonal.

Table 1 Calibration routine

Calibration Results				
	Accelerometer		Gyroscope	
Axis	Offset	Gain (g/LSb)	Offset	Gain (rad/s/LSb)
X	-2.03296	0.003868	-15.53846	0.00125
Y	2.85714	0.003795	-48.21978	0.00126
Z	43.35164	0.003979	-36.41758	0.00126

The robotic arm was used to calibrate the inertial sensors only; no calibration of the magnetometer was attempted. We are only interested in relative changes in yaw angle, not the actual heading angle relative to true North required for navigation. Some simple tests were conducted in which the horizontal magnetometer was rotated through a known angle about the vertical yaw axis; the angular displacement measured by the magnetometer was found to accurate to less than one degree. The error covariance associated with the yaw angle is difficult to estimate as it is dependent on the magnetometer noise and the best guess of the orientation of the sensor; as such, a somewhat arbitrary value was selected and the correlation was ignored.

The process noise error is also difficult to estimate. By inspection of the *time update* model, there are clearly two major sources of error: 1) the assumption that angular rates remain constant, or alternatively, that the angular acceleration is zero and 2) the use of a first order Taylor polynomial approximation for the rate of change of the Euler angles. From inspection of the second order Taylor series below, it is clear that the error magnitude associated with each of these functions is dependent on the Euler angle acceleration, and is on the order of $\frac{1}{2}\ddot{\theta}(t)\Delta t^2$. Whilst we could estimate the maximum expected angular acceleration, it is common practice to provide a generous estimate of the process noise, and so a somewhat arbitrary value was selected.

Equation 11

$$\theta(t+\Delta t) = \theta(t) + \dot{\theta}(t)\Delta t + \frac{1}{2}\ddot{\theta}(t)\Delta t^2 + \cdots$$

The process noise and measurement noise covariance matrices are presented below

$$R = \begin{bmatrix} 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix}$$

$$Q = \begin{bmatrix} 6.2\times10^{-5} & 0 & 0 & 0 & 0 & 0 \\ 0 & 4.461\times10^{-3} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2\times10^{-6} & 0 & 0 \\ 0 & 0 & 0 & 0 & 2\times10^{-6} & 0 \\ 0 & 0 & 0 & 0 & 0 & 3\times10^{-6} \end{bmatrix}$$

Manual Control Principles

After calibration of the IMU via the described calibration routine, the handheld device may be removed and used to control the robot manipulator arm in a way that has been designed to be as intuitive as possible. To reach this point, two main methods were devised and implemented with varying degrees of success. As a basis for both attempted methods, the three Euler angles that describe the

Table 2 Calibration Results

Calibration Routine					
	Stationary				Sweep
Sweep No.	Acc. Offset	Acc. Gain	Gyro. Offset	Covariance	Gyro. Gain
1	X, Y	Z	X, Y, Z	Gyro, roll, pitch	Z
2	Z	X	-	-	X
3	-	Y	-	-	Y

handheld device's orientation in space (Roll, Pitch and Yaw) are translated into movements of the robot arm. The first method was to use the Euler angles to control rotations of individual limbs. The Yaw was to be translated into rotation of the base (J1) joint, the Roll into the rotation of the J4 Joint and the pitch into the rotation of the J2 and J3 Joints.

By far and away the most problematic issue encountered during the project was the lack of information and level of control available using the SPEL+ language. Epson's SPEL+ provides many high level functions that carry out a range of tasks easily and efficiently with only a few lines of code. No inverse kinematic solutions need to be obtained and commands are generally in the form of "Go here" or "Rotate this joint by 20 degrees". As a result, many of the tasks that one would like to carry out with the C3 Arm are fairly trivial and straight forward. Having

such a high level of command, however, was a disadvantage in some of the scenarios for this project. For example, in the first method of manual control, the command found that was most suitable was *JTran* which moves a specified joint by a specified angle. In our control philosophy, we wished manual control to be smooth, responsive and intuitive. For smooth motion, the arm needs to be instructed to rotate a certain joint in a certain direction indefinitely while the user is holding the IMU in a certain configuration. Instead, the *JTran* command can only be used to move discrete amounts. As a result, implementing this command resulted in a control mode that was jittery and slow to respond to changes in direction. This was primarily because the robot arm had to accelerate and decelerate for every incremental change in angular position.

As a second attempt and the method that was eventually settled upon, the Euler angles were translated into linear movements of the robot end assembly in X-Y-Z coordinates according to the world frame. The reason for this method being more favourable was the introduction of the Continuous Path (CP) parameter that became available when using the *Move* command rather than *JTran*. The CP parameter was not adequately documented but appears to function in a SPEL+ program by "looking ahead" to see if a *Move* command follows the current motion command being executed. By doing so, the arm does not accelerate and decelerate for each motion command but moves in a smooth path. Using this method, the Roll and Pitch were converted into movements in the X and Y directions respectively.

In order to meaningfully convert the roll and pitch into a movement, it was desired that the magnitude of the angle determine the speed at which the arm move and the sign determine the direction, positive or negative. With a magnitude of 90 degrees being taken as the maximum movement speed, a roll angle of, say, -30 degrees would be translated into movement of the end assembly in the negative direction with 1/3 the maximum speed.

In software, the Matlab script turns the Euler angles into instruction packets that are sent to the SPEL+ program where they are then parsed. An example of the instruction packet pattern can be seen in Figure 7.

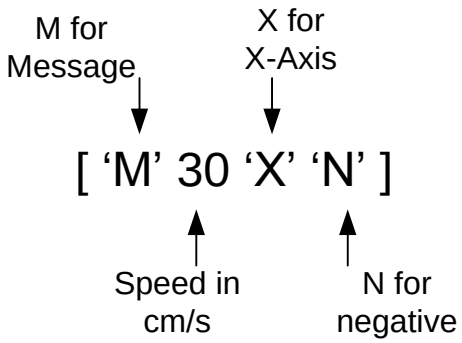

Figure 7: Instruction Packet Protocol

Due to further limitations of the command type used in the SPEL+ language, motion in only one axis at a time could be achieved (as long as CP motion was in use). To determine which axis would be chosen for control, a precedence set up was used to give roll and pitch priority over yaw control via the use of deadbands. Both roll and pitch control operate with a deadband of 5 degrees meaning that any angular deviation of magnitude less than this will be interpreted as 0 by the controller. The deadband for the yaw control was higher at 20 degrees. The effect of this was that a user could hold the IMU relatively flat and no motion would occur, as is desirable. Without a deadband, even when a user thought they

had the device level the robot arm would move with very small speed.

As a final note on the manual control, the yaw angle was used not in relation to absolute North but in relation to the yaw angle at the time manual mode was activated. This meant the user did not have to face North to have no motion in the Z axis.

Conclusion

A three-axis IMU calibration routine was conducted using an Epson Robotic arm. The results clearly showed that the generic scaling factors provided with the unit are not accurate and are presumed to vary between devices. The possibility of a slight deviation in our hardware setup does mean that our measurements are not to be trusted one hundred precent. We can however presume that the calibration procedure discussed here is far more thorough than that which would be performed prior to sale, on an IMU of this cost. Ensuring a level platform when measuring all DOF's of the IMU is vital in obtaining an accurate result. For this reason the pose of the end effector needs to be measured to a precisions far greater than that, which can be determined by the IMU. As the goal of this procedure was purely experimental the method used in mounting and levelling the platform was deemed sufficient. For real world implementation of this system a far more accurate setup would need to be employed. It can also be concluded, in the case of IMU calibration, that the extended functionality of a Robotic arm is redundant and could easily be replaced with a less complex platform capable or rotating at a known rate.

The methods used in controlling the robotic arm was partly chosen based control intuition and partly based on limitations of both the arm and the IMU. In the end a reasonable result was obtained. The user was able to position the end effector with a surprising degree of precision. There was a noticeable delay between an input and the corresponding change in speed; this can contributed to the real time Kalman filter running in Matlab. The mapping of pitch and roll to the x y motion of the end effector is fairly intuitive as it is much the same as using a joystick. Having the yaw control movement in the z axis served as a means of demonstrating the capabilities of the magnetometer.

Given the recourses and time available to us for bringing this project to completion the results achieved were up to our expectations. There are however, a number of aspects that could be improved given more time.

Future Improvements

The communication between all the associated modules added a new layer of complexity. This was especially true in the interfacing between SPEL+ and Matlab. Given more time it would be useful to establish a standard protocol for communicating between these two components. This would allow for fast prototyping of ideas as well as simplified code structure. Further improvements would include running the filter algorithms on-board the pic as apposed to Matlab, the addition of a grabber and increased robustness of the control algorithm.

For a useful implementation of this teleoperation solution a robotic manipulator, designed for this type of operation would need to be employed. If this were the case smoother and more precise movements would be achievable. By aiding the IMU with an external reference it may even be possible to map the position of the operators hand to corresponding position in the manipulators work space. This would further increase the systems capabilities.

References

[1] M. B. Jay Farrel, *Global Positioning Systems and Inertial Navigation*: McGraw-HIll, 1998.

[2] D. H. Titterton and J. L. Weston, "Strapdown Inertial Navigation Technology (2nd Edition)," ed: Institution of Engineering and Technology.

[3] M. L. G.H. Elkaim, L. Pedersen, "Camparison of low-cost GPS/INS Sensors for autonomous vehicle applications," 2008.

[4] Z. Xiaochuan, Q. Yi, Z. Min, N. Jinzhe, and K. Yuxiang, "An improved adaptive Kalman filtering algorithm for advanced robot navigation system based on GPS/INS," in *Mechatronics and Automation (ICMA), 2011*

International Conference on, 2011, pp. 1039-1044.

[5] Z. Bo, Y. Maeda, T. Chiba, Y. Kobayashi, and M. G. Fujie, "Development of a robotic manipulator system for Congenital Diaphragmatic Hernia," in *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, 2011, pp. 723-728.

[6] N. Sakagami, M. Shibata, H. Hashizume, Y. Hagiwara, K. Ishimaru, T. Ueda, T. Saitou, K. Fujita, S. Kawamura, T. Inoue, H. Onishi, and S. Murakami, "Development of a human-sized ROV with dual-arm," in *OCEANS 2010 IEEE - Sydney*, 2010, pp. 1-6.

[7] H. Kazerooni, "Issues on the Control of Robotic Systems Worn by Humans," in *American Control Conference, 1991*, 1991, pp. 386-388.

[8] F. Yu, Y. Yong, C. Feng, and G. Yunjian, "Dynamic analysis and control strategy of the Wearable Power Assist Leg," in *Automation and Logistics, 2008. ICAL 2008. IEEE International Conference on*, 2008, pp. 1060-1065.

[9] S. Verma, J. Kofman, and W. Xianghai, "Application of markerless image-based arm tracking to robot-manipulator teleoperation," in *Computer and Robot Vision, 2004. Proceedings. First Canadian Conference on*, 2004, pp. 201-208.

[10] A. Rodriguez-Angeles, A. Morales-Diaz, Bernabe, x, J. C., and G. Arechavaleta, "An online inertial sensor-guided motion control for tracking human arm movements by robots," in *Biomedical Robotics and Biomechatronics (BioRob), 2010 3rd IEEE RAS and EMBS International Conference on*, 2010, pp. 319-324.

[11] G. B. G. Welch, "An Introduction to the Kalman Filter," *University of North Carolina,* 2006.