Technical University of Denmark

Christian Haaning (s092847)

Fabian Krogsbøll Holt (s113020)

Kenneth Chabert Nielsen (s103062)

Mads Bolmgren (s103728)

Extended Kalman Filter of IMU for Position and Attitude Estimation

June 30, 2015

DTU Electrical Engineering
Department of Electrical Engineering

CHRISTIAN HAANING (S092847)
FABIAN KROGSBØLL HOLT (S113020)
KENNETH CHABERT NIELSEN (S103062)
MADS BOLMGREN (S103728)

Extended Kalman Filter of IMU for Position and Attitude Estimation

Special Course, July 2015

Supervisor:
Nils Axel Andersen, Associate Professor at Electrical Engineering Department of DTU

DTU - Technical University of Denmark, Kgs. Lyngby - 2015

Extended Kalman Filter of IMU for Position and Attitude Estimation

This report was written by:
Christian Haaning (s092847)
Fabian Krogsbøll Holt (s113020)
Kenneth Chabert Nielsen (s103062)
Mads Bolmgren (s103728)

Advisor:
Nils Axel Andersen, Associate Professor at Electrical Engineering Department of DTU

DTU Electrical Engineering
Automation and Control
Technical University of Denmark
Elektrovej
Building 326
2800 Kgs. Lyngby
Denmark
Tel: +45 4525 3576

studieadministration@elektro.dtu.dk

Project period: July 2015 - July 2015

ECTS: 5

Education: MSc

Field: Automation and Control in Electrical Engineering

Class: Public

Remarks: This report is submitted as partial fulfillment of the requirements for graduation in the above education at the Technical University of Denmark.

Table of Contents

Abstract

This report describes the development of adding an inertial measurement unit with a 3-axis - accelerometer, gyroscope and magnetometer, as a daemon for the SMRs (Small Mobile Robots) at the Technical University of Denmark. The data from the inertial measurement unit was fused using a combination of a complementary - and a Kalman filter, where the latter was used for determining the roll and pitch angles and the complementary filter for the heading.

While as the IMU fusion worked well, we were unfortunately unable to fully integrate our attitude with the SMR, fusing the existing sensors with the sensors of the IMU and thereby creating an optimized positioning system.

CHAPTER 1

Introduction

This report provides documentation on a project to perform sensor fusion and filtering on an Inertial Measurement Unit (IMU) connected to a small mobile robot (SMR). IMUs contain a 3-axis gyroscope and an accelerometer measuring angular velocity and linear acceleration respectively. 9DOF IMUs also include a magnetometer. A magnetometer measures the magnetic field intensity and can be used as a compass to determine which way is north. IMUs have traditionally been a key navigation device in aircrafts, missiles, boats, and submarines due to their ability to independently track both orientation and position. IMUs has in recent years gotten smaller and cheaper with the rise of MicroElectroMechanical Systems (MEMS), leading them to be implemented in an increasing wider array of systems ranging from motion capture to vehicle navigation and automation.

The sensors in an IMU have their own strength and weaknesses in relation to precision and reliability, as a result it's common practice to fuse the outputs of the sensors in such a way that you always get the best available data in a given situation. In this project we focus on doing sensor fusion with a Kalman and complimentary filter.

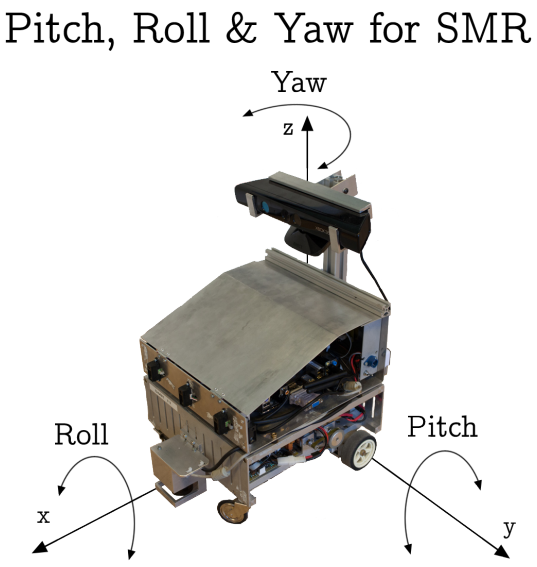

Figure 1.1: Pitch, roll and yaw as shown on the axis of a SMR from Technical University of Denmark

1.1 Objectives and Approach

The objective of this study is to determine the possibilities of using a IMU for position control on the small mobile robots at the department for automation and control at the Technical University of Denmark. The IMU can, when fused together with the odometry, improve the motion tracking. The odometry is unrivalled in precision on short distances, but on larger distances the accumulated uncertainty in the odometry can be corrected with the IMU.

CHAPTER 2

Hardware Specifications

2.1 Inertial Measurement Unit

An Inertial Measurement Unit(IMU) is the main part of an Inertial Navigation System which uses a 3-axis accelerometer and gyroscope to get the position and orientation. Usually a magnetometer which measures magnetic field strength to determine heading is added but other sensors can also be part of an IMU such as a barometer which could be used to estimate altitude.

2.1.1 Gyroscope

A conventional gyroscope uses a spinning wheel connected to a gimbal, which allows the wheel or surrounding frame to assume any orientation independently. This type of mechanical gyroscope works on the physical principle of conservation of momentum which means that the spinning wheel will remain in the same orientation regardless of the tilting and rotation of the frame. By measuring the angle between the gimbals the orientation relative to the initial can then be determined. The main downside of a mechanical gyroscope is that it contains moving parts, this requires very finely created parts as well as lubrication to avoid friction which would make the gyroscope drift over time. Another potential problem based on the situation, is that mechanical gyroscopes usually needs time to warm up before use.

The gyroscopes used in IMUs are built with silicon micro machining techniques and called a Microelectromechanical systems (MEMS) accelerometers. The gyroscopes used in IMUs contain a radial vibrating part, which measures the resulting orthogonal force from the Coriolis effect when the gyro is rotating, to determine the rate of change in orientation. An image of this can be seen in figure 2.1. Because the output of the gyroscope is in angular velocity it means that you need to integrate the output to get your orientation relative to your starting position. MEMS gyroscopes contain very few parts and are relative cheap compared to other types of gyroscopes. They main advantages of the MEMS gyroscopes is their small size, weight, cost, power consumption combined with a good durability. The main disadvantage is that the MEMS gyroscopes are a lot less precise and their bias stability or angular random walk can be orders of magnitude worse than traditional gyroscopes.

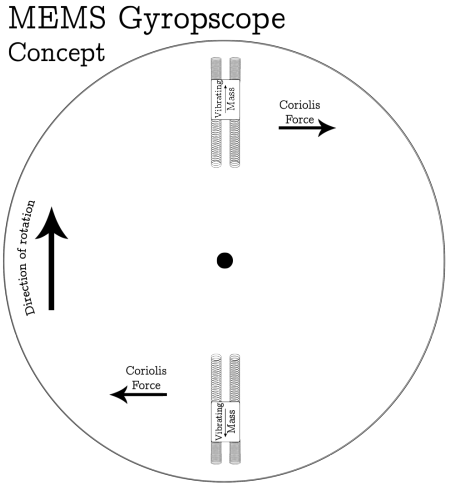

Figure 2.1: MEMS Gyroscope concept for a single axis. The rotational speed can be measured from the weight transfer of the vibrating mass during turning.

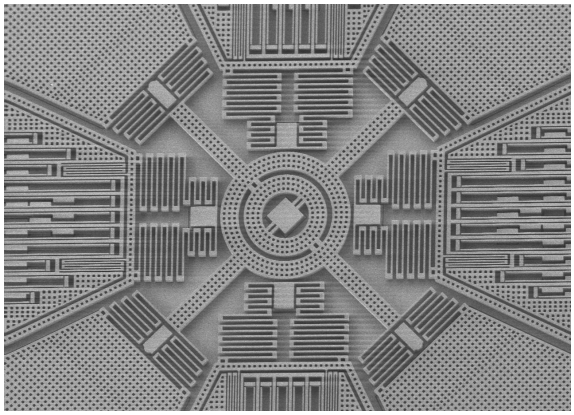

Figure 2.2: Image of MEMS Gyroscope[1]

Typical MEMS gyroscope errors

Since a MEMS gyroscope relies on integration, any small error in angular velocity will accumulate and become a large error in orientation. This problem is most prevalent in the gyroscope's bias/offset which is the average value of the output for a stationary gyroscope. If this value is not zero then the angular orientation will be integrated up and the oriantation will start drifting even if the gyroscope is not undergoing any rotations. This bias can change over time and change based on temperature which means that you will need to calculate the bias often to calibrate the gyroscope. The effect of the temperature in the bias is often not linear, which makes it more complicated to calculate the correct bias even if you know the temperature. Most IMUs contain a temperature sensor which allows you to always know the temperature of your gyroscope, so it is possible to take it into account when you get your data. Over time the bias changes due to flicker noise which causes the bias to change approximated by a random walk algorithm with a standard deviation which is usually given in the datasheet. Another error is that the simple white noise error on the angular velocity output also gives a random walk error when integrated to orientation.

2.1.2 Accelerometer

Accelerometers usually come as either a mechanical or a solid state accelerometer. A mechanical accelerometer consists of a mass suspended by springs, any acceleration will cause a displacement of the suspended mass which can then be read. By knowing the properties of the mass and the spring the force can be determined and be use of Newton's second law: F = m*a, the acceleration can be calculated. Solid state accelerometers comes I many different types. They use a small moving, straining or vibrating parts and measure acceleration by detecting change in capacitance, strain, magnetic field, frequency of the surface acoustic wave or other such changes.

The accelerometer used in IMUs and many electronic devices such as mobile phones are MEMS accelerometers. These accelerometers can be made as either a mechanical or solid state but a popular design shown in figure 2.3, uses a flexible silicon layer which can be displaced like a mechanical accelerometer.

[1]Image source: `http://www.geekmomprojects.com/geekmomprojects-wp/wp-content/uploads/2013/03/mems_gyroscope.jpg`

This displacement is measured by having the silicon layer extend “arms” out between two charged plates that works like capacitors, if the arm moves closer to one plate a difference in voltage can be measured.

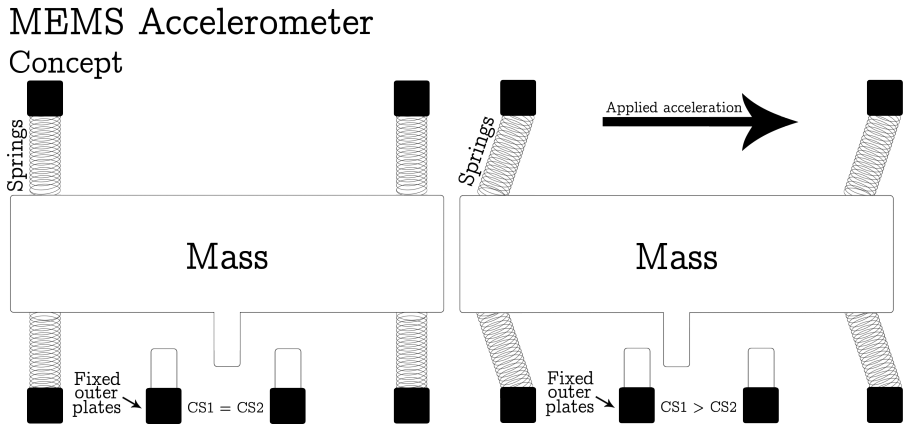

Figure 2.3: MEMS accelerometer concept for a single axis. Left drawing shows accelerometer not in motion, whereas in the right drawing, the accelerometer is being accelerated to the right.

Typical MEMS accelerometer errors

MEMS accelerometers have many of the same errors as gyroscopes, most prevalent is the bias/offset of the output. The seriousness of the problem with this bias depends on what you intend to use the accelerometer for. If you use the accelerometer to determine the orientation of the device, a small bias error is unlikely to give a large error in orientation outside of the gyroscopes singularity angles. However if you use the accelerometer to determine position, even the smallest error in acceleration when double integrated will lead to a significantly larger error in position. This means that if you wish to determine position from accelerometer data you would need a very precise and most likely expensive accelerometer. Because of this accelerometer based positioning is rarely used if other alternatives are available. The bias of the accelerometer also drifts due to flicker noise and temperature, but to a lesser degree than a gyroscope of an equivalent quality.

2.1.3 Magnetometer

Magnetometers are used to measure the magnetic strength either of a magnetic material or of the magnetic field in a specific location. There are many different ways to measure a magnetic field but two popular ways are to either measure the voltage generated when a moving magnetic field passes through a coil as by the "Hall effect" or by measuring the Lorenz force which describes the force on a moving particle in an electromagnetic field. They do this by having a wire usually in a u-shape with current (particles) running through it on a platform whose displacement from the Lorenz force can be measured. An image of a Hall MEMS magnetometer can be seen in figure 2.5.

The magnetometer in IMUs is mostly intended to be used as a compass by measuring the earth’s magnetic field and determining the cardinal directions from this. While having a compass can be useful in a variety of different ways based on the situation, it is especially useful within the IMU because it does not drift unlike the gyroscope. This allows you to detect gyro drift and re-calibrate or filter your output accordingly to avoid problems.

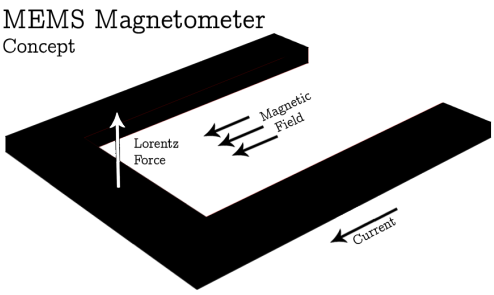

Figure 2.4: MEMS Magnetometer concept for a single axis. The magnetic field is calculated by measuring the flex of the conducting u-beam caused by the Lorentz force.

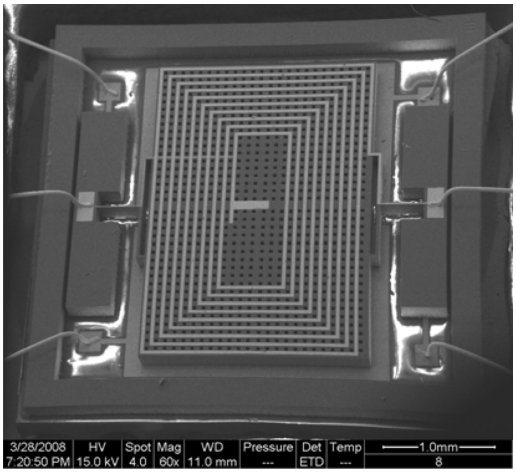

Figure 2.5: Image of a MEMS Magnetometer with a coil for measuring the Hall effect[2]

Typical MEMS magnetometer errors

The biggest cause for error for a magnetometer, when used as a compass, is interference from other magnetic fields. Since earth's magnetic field is relatively weak, it is necessary to calibrate the magnetometer by finding the range of measured field intensity in all possible/relevant orientations, in order to accurately calculate the cardinal directions. This means that the used range of values from the magnetometer will usually be much smaller than the full range the magnetometer can measure, which causes weaker noise to have more impact. The biggest problem however are fields generated from nearby sources such as electrical current, magnetised metal and magnets whose fields' strength often will be orders of magnitude larger than the earth's in close proximity.

2.1.4 MPU 9250

The IMU we use in our project is an InvenSense MPU-9250 which contains a 3-axis gyroscope, accelerometer and magnetometer with 16 bit resolution. The IMU is interfaced with I^2C Communications using a single address for the entire board with sub addresses for all data and configuration registers.

The gyroscope can be configured to measure the angular velocity in the full-scale of either ±250, ±500, ±1000 or ±2000 $^\circ/s$. It also has a digitally programmable low-pass filter which can be used to counteract noise and smoothen the output if desired. The accelerometer has a full range of either $\pm 2g$, $\pm 4g, \pm 8g$, or $\pm 16g$. It also has capabilities to go to sleep and wake the IMU if power consumption is a concern. The magnetometer has the constant measurement range of $\pm 4800\ \mu T$. In our implementation we chose ±250 $^\circ/s$ and $\pm 2g$ as this would give us the best resolution and was within the area we planned to operate.

2.2 Small Mobile Robot

The Small Mobile Robots (SMR) at DTU is an autonomous locomotive devices used for research and education. The SMR have several different sensors and actuators. The actuators is two motors to give the

[2]Image source: `http://www.mdpi.com/sensors/sensors-09-06951/article_deploy/html/images/sensors-09-06951f4-1024.png`

locomotion. The sensors is; A camera for vision based control and object detection. A Laser scanner for localization. A set of IR distance sensors for obstacle detection. And a tachometer on each wheel used for motion tracking. The SMRs commonly use odometry for motion tracking based on the tachometers. This however can lead to kidnapped robot syndrome if the odometry gives a position which is different from the actual position. This could be due to the robot being physically lifted and turned, or if there is some slippage between wheel and ground. The SMR have a hardware abstraction layer called the Robot Hardware Daemon (RHD) which stores data for the sensors and actuators.

2.2.1 RHD Plugin

In order to add our sensor to the stored data, we had to read the data from the device and add it to the server variables, and to do this we wrote a RHD plugin. The purpose of the plugin is to move the data from the external port, in to more global accessible variables, as we also planned on writing another plugin, for one of the other servers available in Mobotware, to process the numbers and calculate pitch, roll and yaw of the SMR.

Originally we had planned on using a USB to I^2C converter, but after a lot of different attempts, we had to figure out another solution, as we were unable to properly set up any of the three IMU's we tested, and therefore unable to receive data. We ended up using an Arduino Nano, to setup the IMU, read the raw data and pass it on to the RHD as a serial write.

CHAPTER 3

Theory

This chapter will briefly describe the theory applied for the sensor fusion of the accelerometer, gyroscope and magnetometer on the IMU.

3.1 Sensor Calibration

In order to get usable data, the IMU has to be calibrated. This involves calibrating each sensor. e.g accelerometer, gyroscope and magnetometer respectively. The calibration routine for the accelerometer and gyroscope is rather trivial, since the only thing of importance here is to determine the bias of each sensor. This is done simply by measuring a lot of raw measurements and averaging them out to obtain the bias.

The calibration of the magnetometer involves determining the range of values earth's magnetic field has in all possible orientations. This means that the values has to be obtained by rotating the magnetometer around the three axis, x, y and z and can therefore not be made simultaneously with the other two sensors who needs to be stationary. As the SMR is an electronic device which may emit an electromagnetic field, it is preferable to do the calibration while the IMU is connected to the SMR which can be more difficult process. You can simplify this slightly by only turning the SMR to orientations that it could possibly take under normal operations, by realising that if the SMR is lying on its side you probably have bigger problems than an imprecise magnetometer.

3.2 Computing Attitude

3.2.1 Orientation from Accelerometer

To get orientation from the accelerometer we use the effect of gravity in the three directions to calculate the pitch and roll as shown in equations (3.1a) and (3.1b) where A_{xyz} is the accelerometer value in the given axis.

$$\phi_{pitch} = atan2\left(\frac{-A_x}{A_z}\right) \tag{3.1a}$$

$$\theta_{roll} = atan2\left(\frac{A_y}{\sqrt{A_x^2 + A_z^2}}\right) \tag{3.1b}$$

There is a singularity problem with equation (3.1a) were both the numerator and denominator simultaneously can be zero, when the accelerometer is oriented along the y-axis. In this position pitch can not be calculated and any small amount of noise in this position will result in a unstable random angle estimation.

There is no perfect solution to this problem but a commonly used workaround for stability, is to introduce a small fraction of the y-axis' accelerometer data in the denominator of equation (3.1a). This is done as the euclidean distance of the z- and y-axis with the sign of the z-axis, as shown in equation (3.2).

$$\phi_{pitch} = atan2\left(\frac{-A_x}{sign(A_z)\cdot\sqrt{A_z^2+\mu\cdot A_y^2}}\right) \tag{3.2}$$

μ should be as small as possible while still large enough that noise on the accelerometer signal does not take precedence in the calculation, in our case we use $\mu = 0.01$.

3.2.2 Heading from Magnetometer

The heading of the magnetometer is determined by crossing the vectors of the magnetometer and the accelerometer, in order to determine the influence of the gravitational force, giving the east vector. The newfound vector is then crossed with the accelerometer once more to determine north. The heading angle is then found as atan2 of the dot-product of east and the orientation vector and the dot product of the north vector and the orientation vector. The orientation vector should be a unit vector pointing to the desired direction of the orientation.

$$East = magnometer \times accelerometer \tag{3.3}$$

$$North = accelerometer \times East \tag{3.4}$$

$$heading = atan2(East\cdot[1,0,0], North\cdot[1,0,0]) \tag{3.5}$$

3.2.3 Complementary Filter

For correcting the gyroscope drift on the yaw angle, the magnetometer and gyroscope are fused. This is done by using a complementary filter, which consists of a low and a high pass filter. The gyroscope data is filtered through the high pass and the magnetometer through the low pass. A block diagram of this filter can be seen in figure 3.1.

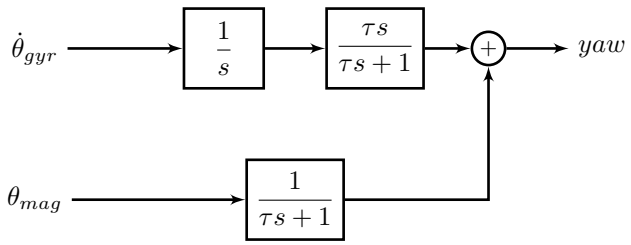

Figure 3.1: Complementary Filter (Laplace-Domain)

The complementary filter in figure 3.1 can be simplified further to a single low pass filter. This is shown in figure 3.2. This means that only a time constant is multiplied with the angular velocity, then added together with the magnetometer data and passed through the low pass filter.

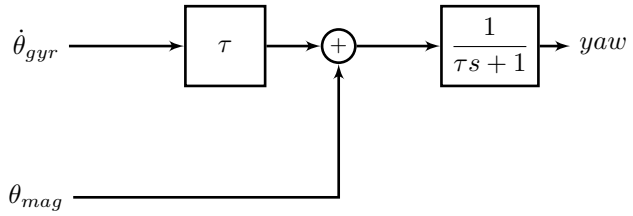

Figure 3.2: Simplified Complementary Filter (Laplace-Domain)

3.2.4 Kalman Filter

For determining the roll and pitch angles, a Kalman filter is used. This type of filter attempts to predict states in the future, in this case the roll and pitch angles. The filter consists of two steps, a prediction estimate and an update step. These consist of the following equations. Equation (3.6) and (3.7) which describes the prediction estimation step.

$$\hat{x} = Ax_{n-1} + Bu_n \tag{3.6}$$

$$\hat{P} = AP_{n-1}A^T + Q \tag{3.7}$$

And the update equations are described in (3.8) to (3.12).

$$y = z_n - H\hat{x} \tag{3.8}$$

$$S = H\hat{P}H^T + R \tag{3.9}$$

$$K = \hat{P}H^T S^{-1} \tag{3.10}$$

$$x_n = \hat{x} + Ky \tag{3.11}$$

$$P_n = (I - KH)\hat{P} \tag{3.12}$$

$\hat{x}$ and x_n describes the states, which is the roll and pitch angle here. $\hat{P}$ and P_n shows how well or confident the Kalman filter is with a determined solution. Q describes the covariance of the noise of the process, here the accelerometer and gyroscope is that process. During the update step, K is updated, which is the Kalman gain. H is used to describe the sensors. R is very similar to Q, however, it describes the confidence in the sensors. z is the sensor measurements, which is basically what is being Kalman filtered. I is just an identity matrix. It should be noted, that the Kalman filter shown here only applies to linear processes. However, an Extended Kalman filter can be used for non-linear systems. If the filter is applied for only roll or pitch, the extended Kalman filter is not necessary, as the pitch and roll calculated individually can be considered linear.

When determining the roll or the pitch, the state x_n can be determined by three variables as shown in equation (3.13).

$$x_n = \left(\theta, \dot{\theta}, \dot{\theta}_{bias}\right)^T \tag{3.13}$$

These three variables describe the angle, the angular velocity and a bias. The bias is used for the angular velocity and the angle and angular velocity can be determined as in equations (3.14) to (3.16)

$$\theta_n = \theta_{n-1} + \Delta t \left(\dot{\theta}_{n-1} - \dot{\theta}_{bias-1}\right) \tag{3.14}$$

$$\dot{\theta}_n = \dot{\theta}_{n-1} \tag{3.15}$$

$$\dot{\theta}_{bias} = \dot{\theta}_{bias-1} \tag{3.16}$$

The entire process can be summarised as follows

1. The current state is predicted from the previous state, i.e $\hat{x}$
2. $\hat{P}$ is updated with respect to uncertainties from the Q matrix, which, as previously mentioned, is the covariance of the noise of the process
3. Sensor data is sampled and the angle and angular velocity is determined. This is what z_n holds, i.e $z_n = \left(\theta_n, \dot{\theta}_n\right)$
4. When data has been sampled S is determined. This is the covariance of the measurements just sampled.
5. The Kalman gain K is then determined. This value will tell something about the weight of the model and the actual data. If e.g the confidence of the value K is high, the measurements will be more trusted than the model and vice versa.
6. Now the state x_n can be determined by scaling it with the Kalman gain K and the covariance of the state P_n is updated according to the new measurements.

In order to optimise the Kalman filter, the only thing left is to tweak the Q and R matrices.

CHAPTER 4

Implementation

4.1 RHD Plugin

The function of the RHD plugin is to read the IMU data from the Arduino board and parse the data to the globally accessible RHD variables. The Arduino board sends the data from the IMU as a string. The data needs to be converted by the RHD plugin, from characters into integers, and if a dash (-) is present the number will be multiplied by -1. The data is space separated between x, y and z values, and semicolons (;) separate the sensors. The data-set is then ended with a newline (\n). These characters is used by the RHD to cut the string into singular values which are saved into the appropriate variables.

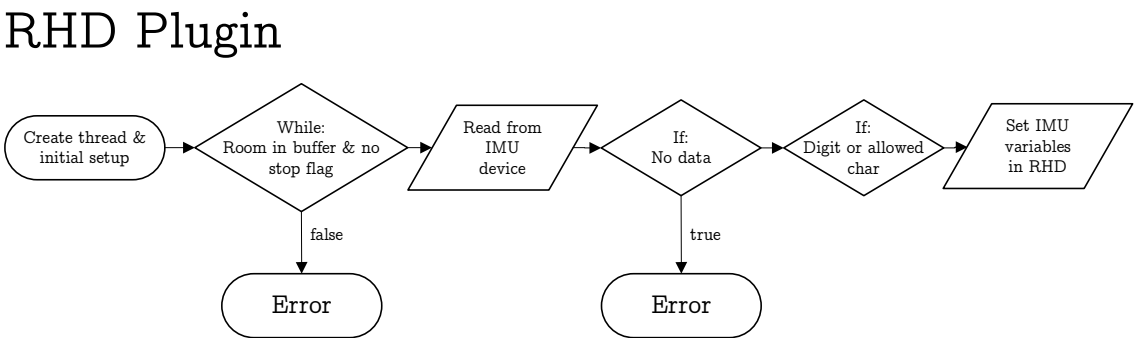

Figure 4.1: RHD plugin flowchart

4.2 Server plugin

The server plugin reads the data saved in the global RHD variables and processes the data. The values are converted to SI units as well as calibrated using either the values read from the server initialisation or from the values saved in to the server from running a calibration command. We have implemented functions to calibrate the sensors using two simple commands as explained in table 4.1.

Magnetometer heading is determined using the data as described in section 3.2.2. Pitch and roll are determined by integrating the gyroscope data and using atan2 to determine the total rotation around the specific axis of the accelerometer.

When the pitch, roll and yaw has been found for each of the three sensors in the IMU, the data has to be fused. For the pitch and roll, we have implemented a Kalman filter as describe in section 3.2.4 in which the angle is predicted from previous data and known errors.

The data from the magnetometer includes considerable amounts of white noise, while the gyroscope data drifts over time, due to the integration. So as explained in section 3.2.3, we implemented a complementary

filter for calculating the yaw.

Because of the way the Mobotware plugins work, we implemented global variables, saved in the server, when we needed to save values from each iteration of the measurements.

The plugin is made so that it works with the camera server, replying with the necessary XML-tags for the MRC-server of Mobotware, to be able to read the data in the "*$vis*"-variables. These variables can then be used in a SMR-CL script, for driving purposes.

Call	**Variable**	**Description**
help		Shows the help function of the plugin
data		Main call for all data associated with the IMU. This call will collect data from the RHD-variables, convert them to SI-units, transform them according to the calibration, calculate the pitch, roll and yaw for the individual sensors, as well as calculate the Kalman fused pitch and roll and yaw from the complementary filter. The position is also estimated using the accelerometer data (double integration). All this data is returned in a MRC-ready XML-tag, using *$vis0* to *$vis4* (*pitch*, *roll*, *yaw*, *x* and *y*).
	cf & *ts*	Two possible settings are possible to set which are used for the data collections. cf is the cutoff frequency of the low-pass filter used in the complementary filter, and can be set by simply writing *cf=X*, where *X* then will be the new cutoff frequency. *ts* is the truncation setting. We attempted to minimize the drift of the position measurement from double integrating the accelerometer data through truncation by nullifying acceleration and velocity values bellow a certain threshold.
reset		Resets all the values associated with the data call. Values reset are; old measurements, covariance matrix of the Kalman filter, the estimated position, the last measurement time and the initial heading of the magnetometer (used for calculating movement rather than angle of magnetic north). When data is called after the data has been reset, the values necessary for a measurement,will be recalculated.
heading	*deg*	Shows current heading angle in relation to magnetic north. (Requires the magnetometer to be calibrated!) If *deg* is added to the call, the return will be in degrees rather than radians.
calib		Initializes the accelerometer and gyroscope calibration function. The device takes a lot of measurements to measure the static error. This error is then saved in to the server variables, so it can be used for further testing. If the user wishes for the values to be saved through a server restart, the values should be saved in to the server initialization file.
mag_calib		Initializes the magnetometer calibration function. The device should be moved around to reach the highest and lowest possible values. As the device is very sensible to magnetic fields, keep watches, screwdrivers and other metallic objects away from the IMU. Should the IMU be placed on a robot where there may be constant noise in the form of a magnetic field, this should be present in the calibration process.

Table 4.1: Possible calls of the IK plugin

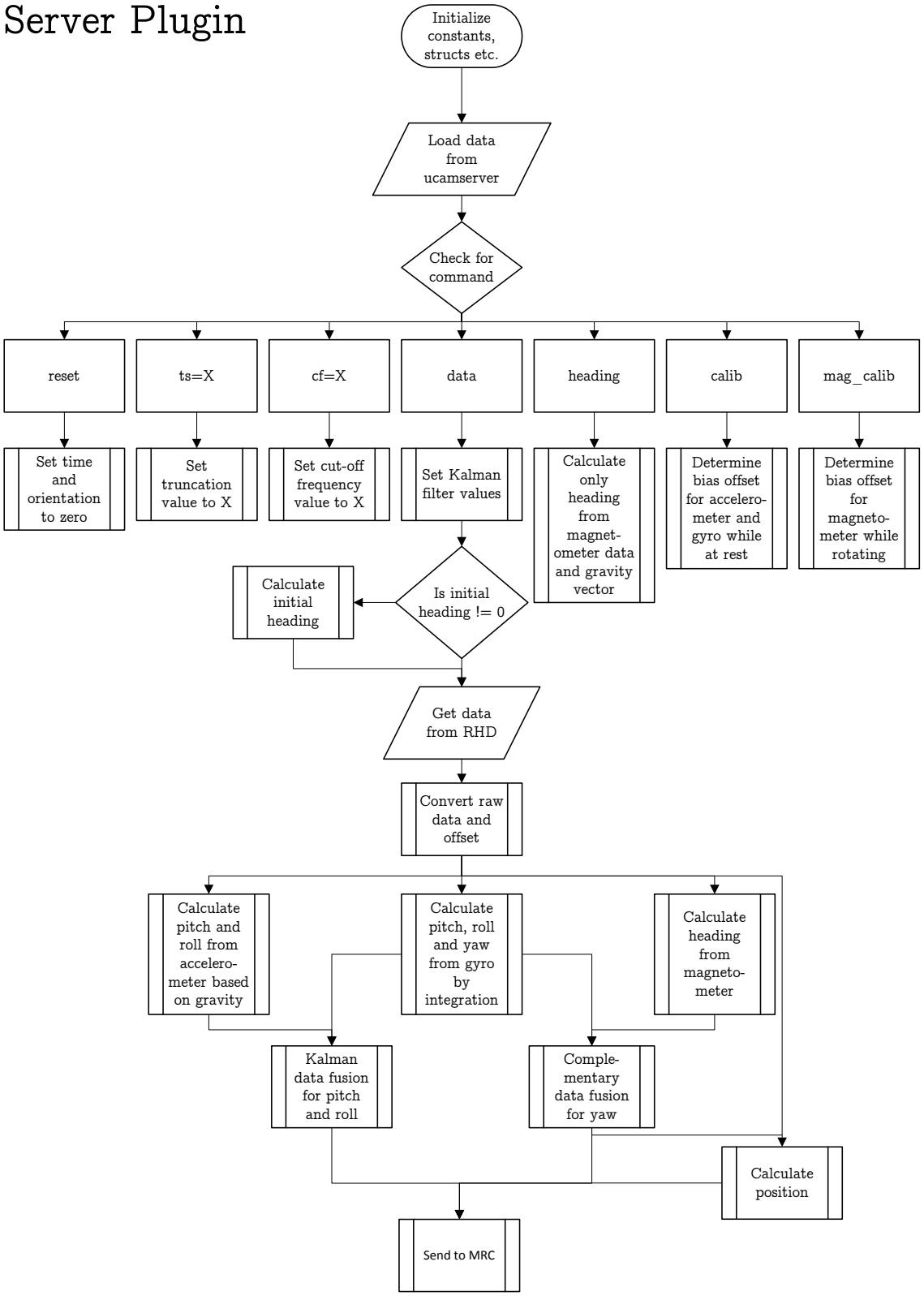

Figure 4.2: Flowchart of plugin

CHAPTER 5

Results

5.1 Test Setup

The IMU device is attached to a SMR and calibrated. The IMU device is placed with the IMU y-axis pointing forward, and the z-axis pointing up. Two tests are made; One where the SMR drive a square with length of 1.2 meter. The second test makes the SMR drive 5 meters on a straight line turn around and return to the starting point. The tests are done by using a SMR-CL script, which is then run on the MRC server of the robot. The script itself, starts off by resetting the IMU data and then proceeds to set the server plugin to broadcast data every tenth of a second. The robot then drives 0.1 meter at a time, turn in to the desired angle, relying solely on IMU data for the angle, before continuing the journey, stopping after another 10 cm.

The first test is made to show the precision of the IMU on short distances and the second test shows precision on long distance. Both tests is compared to an equal drive but with the use of odometry instead of the IMU. The odometry test works in the same manner, except it only relies on the odometry data.

5.1.1 Calibration

Calibration of the accelerometer and gyroscope was an important step which we needed to do regularly to get accurate measurement. To calibrate the sensors we simply ran the calibration function in the server plugin when the IMU was stationary. The comparison between a calibrated and uncelebrated output can be seen in figure 5.1.

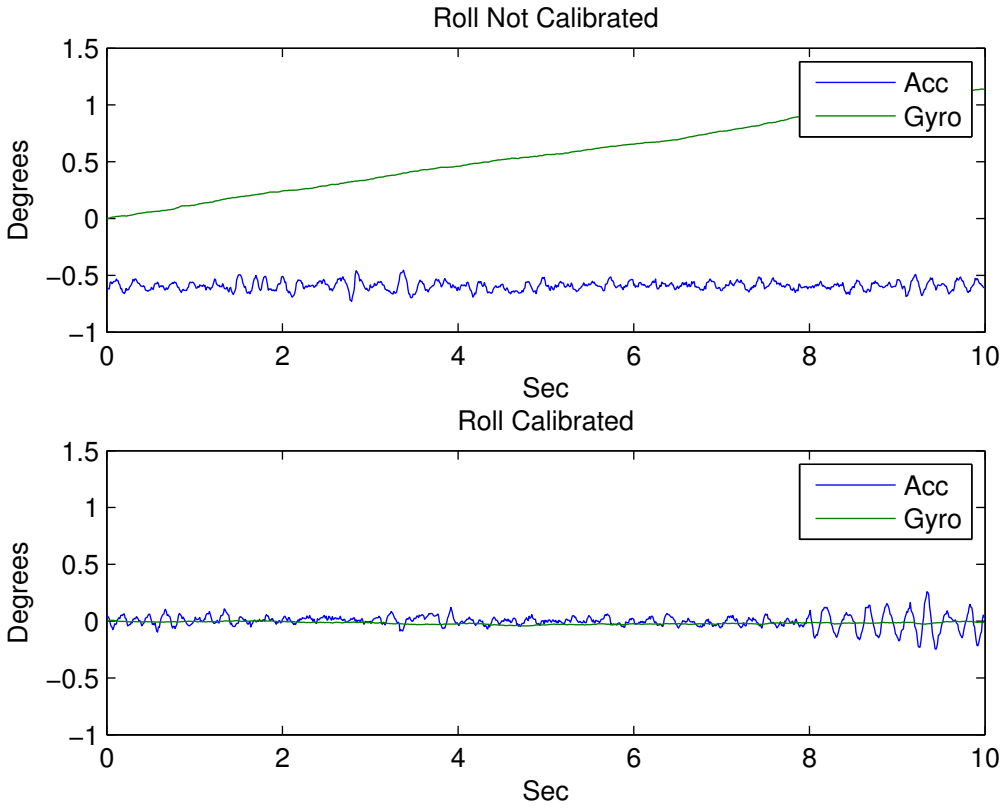

Figure 5.1: Before and after calibration of roll measurements

5.1.2 IMU Based Drive

Straight drive test

The purpose of the long drive test (see figure 5.2 were to establish the precision of the complementary filter. On the roll and pitch we observed the pitch being more noisy each time the robot corrects its angle. We believe this to be caused the SMR braking and starting before and after correcting the angle causes the lid of the SMR the shift slightly up and down. The roll has a slight offset caused by the attachment to the SMR, where the IMU is connected through a breadboard where the angle of the IMU can change slightly. For the yaw during a long drive on a straight line, the complementary filter should be $0°$and on the the return trip $180°$. During the forward motion the complementary filter approaches the magnetometer due to the low-pass nature of the filter for the magnetometer data and when the SMR turns the complementary filter approaches the gyroscope because of the high-pass.

The SMR returned to within 5 cm the starting position, this error is most likely caused by the fact that the gyroscope magnetometer does not obtain the exact same value after the turn which means that one of them is wrong. In this case we believe that the error has to do with the calibration of the magnetometer.

In order to have something to compare the IMU data with, we include the odometry yaw in the plot. We decided not to calibrate the odometry and use the standard values as that would allow us to see a difference between the two even over short distances. It can be seen that the SMR - if driven on the odometry only, slowly drifts to the left. The rate of the drift seems to be about $9°$per meter and looking at the odometry position it would result in a end point more that 2 meters away from the start point.

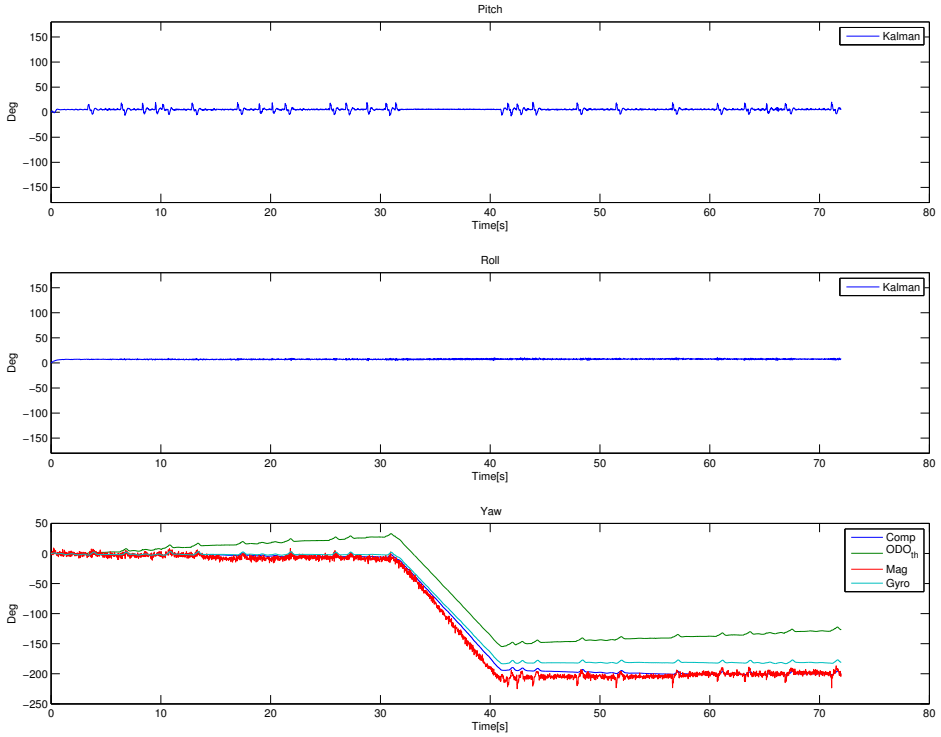

Figure 5.2: The Pitch, Roll and Yaw data of a 10m drive with a 180 degree turn halfway

Square drive test

The square test was made mainly to test the angle precision in all yaw angles. The square drive returned the SMR to within 3 cm of the starting position. We use the same calibration as with the other test and we see again that the magnetometer takes a different value from the gyroscope as we move away from 0 yet seems to return to it when we move back, which might suggest a calibration problem. We still see filter trust gyroscope when turning and the magnetometer when driving straight as it should. We also still observing the odometry slowly drifting to the left which would return the robot far from starting point compared to the IMU run.

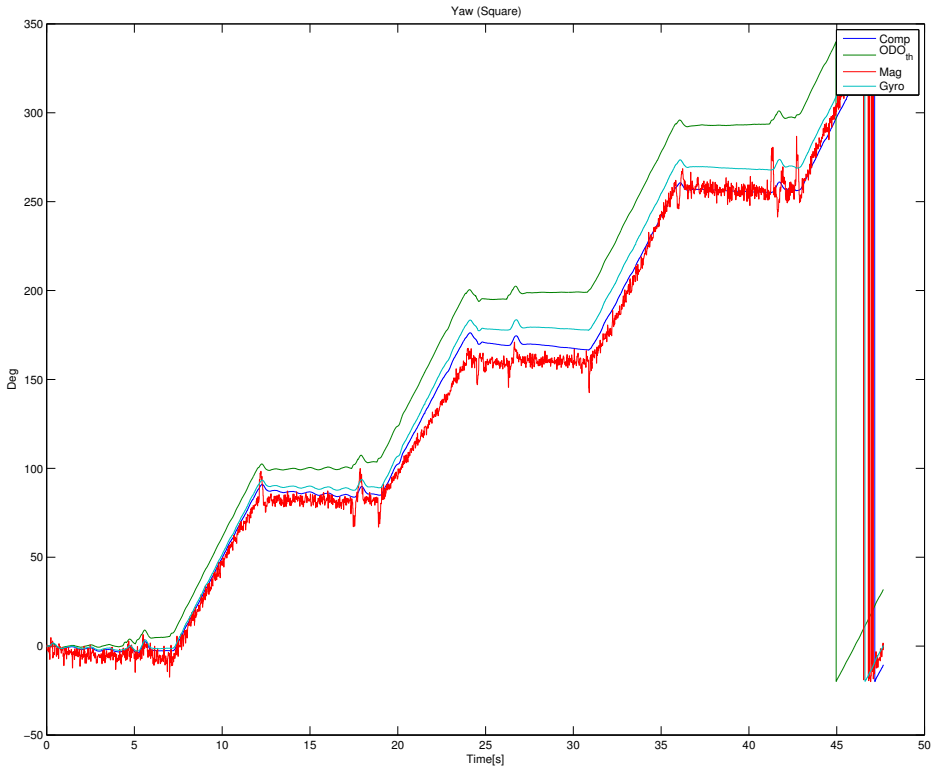

Figure 5.3: Yaw data of SMR during square test

5.1.3 Pose Estimation

We did some tests trying to estimate our position entirely from IMU data, using the accelerometer to determine forward velocity and the complementary filter to determine turns. However we found that the double integral quickly grew far too large for even the smallest errors. We tried to truncate the data by forcing small accelerations and velocities to zero and while we got data which resembled the path we drove in form, the magnitude of the values varied greatly. In the end we did not manage to get acceptable data which could be used in any real scenario.

CHAPTER 6

Conclusion

While the fusion of the IMU's sensors works well for attitude estimation, we have not managed to fully integrate them with the preexisting SMR control system. It was our initial goal to fuse the IMU data with the odometry data in order to get the best of both types. This way the heading of the IMU could be used as a means of correction in case of odometry error such as wheel-slip which can easily cause an error in the angle. The position determined from the accelerometer data can also be used if correctly fused with the odometry. If the wheel encoders says that the robot is standing still, the most likely cause of change in the position from the accelerometer is the drift from double integration, whereas if the accelerometer implies that the robot is moving fast in one direction, it is likely that the robot is currently being "stolen" (moved by a force not being the wheels), as in the kidnapped robot problem (as described on page 306 of Autonomous Mobile Robots, 2. Edition, by Siegwart).

A longer test period would have been preferable, as we had initially planned for, but the unforeseen problems with the I^2C to USB converter set us back a whole week of our three week plan. Further tests and implementation of the positioning Kalman fusion would require more time, but is definitely an intriguing idea for further development of the combined sensor fusion.