Angle estimation using gyros and accelerometers

This version: February 14, 2020
Chapter 1

Introduction

The purpose of this lab is to illustrate sensors, measurements, filtering and simple sensor fusion. Our goal is to estimate the tilt-angle of the MinSeg. By extracting information from both the accelerometers and the gyro and combining these using low-pass and high-pass filters, we will create an improved estimate of the angle of the MinSeg, compared to a naive approach of using only the gyro, or only the accelerometer.

Figure 1.1: Newton Nordic AB in Linköping develops a camera stabilizer intended to, e.g., hang from cables and fly over soccer fields. For precise aiming through three controlled motors, it is important to know the orientation (pitch-, roll- and yaw) of the camera. Gyros and accelerometers are used to extract as much information as possible, with minimal sensitivity to calibration and noise errors.
1.1 Hardware set-up

The lab is based on three main hardware components.

To begin with, we have a standard desktop computer. This computer is used to automatically develop and deploy code using MATLAB and SIMULINK models.

We use a board which is equipped with an accelerometer and a gyro. The board is built around an Arduino micro-controller which runs the auto-generated code. The Arduino computer also communicates with the desktop computer and thus allows us to look at the measurements.

The Arduino board together with motor and wheels is called the MinSeg.

1.2 Trouble shooting

Complaints about COM port or connection when downloading to board

- Make sure USB cable is firmly attached on both ends.
- Disconnect USB-cable and connect it again.
- Make sure you only have one Simulink model open.
- Restart MATLAB.
- If it still does not work after several tries, follow the instructions in fixComport.pdf.
Chapter 2

Preparation

The questions below, and all questions throughout the document marked as Preparation must be done before attending the lab. Note that there are additional preparation exercises in Chapter 3.

Solutions to all questions should be available upon request from the lab assistant.

The scheduled time spent with the laboratory equipment is only a small part of the complete lab, as a major part is spent on the theoretical material during preparations.

**Preparation 1**  Read Chapter 1 and 2 in the auxiliary course compendium.

**Preparation 2**  A gyro allows us to measure the angular velocity, which we can integrate (numerically in practice) to obtain the rotation angle. The problem is that the angular velocity measurements have errors.

We typically divide errors into a fixed constant error (called bias or calibration error) and a faster varying error term (noise) which is 0 on average. Consider the measurement of a true angular velocity \( \omega(t) \) which we want to integrate to obtain the true rotation angle \( \theta(t) = \int_0^t \omega(\tau) d\tau \). Let the measurement be \( \omega_m(t) \). A simple description uses a bias \( b \), noise amplitude \( \epsilon \) and noise frequency \( N \),

\[
\omega_m(t) = \omega(t) + b + \epsilon \sin(Nt) \tag{2.1}
\]

Integrate this expression analytically and derive an expression for the estimated angle \( \hat{\theta}_m(t) = \int_0^t \omega_m(\tau) d\tau \) (you assume initial conditions to be 0).
The solution will consist of the true angle, a linear term, and a periodic term. Verify that the term in the error \( \theta_m(t) - \theta(t) \) that depends on \( b \) will grow as time increases (called linear drift), but the term coming from the high-frequency noise will be limited and decrease the larger \( N \) is.

**Preparation 3** To obtain an angle \( \theta(t) \) from angular velocity \( \omega(t) \), we have \( \theta(t) = \int_0^t \omega(\tau) d\tau \). However, we do not have the function \( \omega \) but only a finite number of samples, and must use an approximation. One such integral approximation is a rectangle approximation based on Euler backwards. With measurements \( \omega(kT_s) \) for \( k = 0, 1, \ldots \) and sample-time \( T_s \), the approximation is given by

\[
\theta(kT_s) = \theta((k-1)T_s) + T_s \omega(kT_s)
\]  
(2.2)

Consider the case in Figure 2.1 where a signal \( \omega(t) \) has been sampled with \( T_s = 1 \) second. Illustrate graphically in the figure how the integral approximation from \( t = 0 \) to \( t = 4 \) is computed, and confirm that it leads to the estimate \( \theta(4) = 9.5 \). Note that the value \( \omega(0) \) never is used in the backwards approximation. You initialize the angle estimate to \( \theta(0) = 0 \).

**Preparation 4** When working with sampled signals, we often use the z-transform for notational convenience. Instead of working with \( \theta(kT_s) \) we introduce its transform \( \Theta(z) \) and the unit delay-operator \( \frac{1}{z} \) to express \( \theta((k-1)T_s) \), i.e., the transform of "previous value" \( \theta((k-1)T_s) \) is \( \frac{1}{z} \Theta(z) \). Introduce this notation for (2.2) and show that the integral approximation algorithm (2.2) can be represented as

\[
\Theta(z) = \frac{T_s z}{z-1} \omega(z)
\]  
(2.3)

In other words, the Euler backwards integral approximation is described by the discrete-time transfer function \( \frac{T_s z}{z-1} \).

**Preparation 5** Read the complete lab-pm. There are more theoretical questions in the pm which you are supposed to complete as preparation.
Figure 2.1: A time-continuous signal (blue curve) sampled with $T_s = 1$ second (blue dots). The blue curve is not known to us.
Chapter 3

The lab

The lab will primarily consist of experimentation and data collection, using theoretical results and strategies derived during your preparation.

Items labeled Preparation are questions you are supposed to solve before attending the lab.

Items labeled Task are questions you answer and solve when attending the lab and have access to the hardware.

3.1 Gyro and accelerometer

The board is equipped with a gyro measuring angular velocity around 3 axes, and an accelerometer measuring linear acceleration in the same axes. The accelerometer and gyro hardware are placed on the blue board as indicated in Figure 3.1. The measurements are done in a body-fixed coordinate system as indicated in Figure 3.2.

The gyro

Although the gyro is the natural sensor to use for estimating an angle (as it measures the derivative of an angle), it has some problems. To begin with, the gyro measurements can only give us angles relative to initial orientation, as we simply integrate the angular velocity. In many applications, we...
are interested in the absolute angle of the device, such as knowing if the MinSeg is standing straight up or not. In addition to this, the sensor has bias errors, meaning that the angle estimate will drift. Even if we leave the device completely still, the preparations showed that any bias in the angular velocity measurement will lead to a growing error in the angle estimate.

In the lab, we are only studying the tilt-angle of the MinSeg, i.e., the balancing angle of the MinSeg relative to the surface (0° degree when lying flat on a table, and −90° when standing straight up). Consequently, we are only interested in rotations around the x-axis (rotations in the y−z plane) and thus only use one of the gyro measurements.

The accelerometer

The accelerometer might seem unrelated to the orientation (i.e. angle) of the device, as it measures the three linear accelerations of the object. However, the acceleration is measured relative to free-fall, meaning that when
Figure 3.2: Body-fixed coordinate system used for accelerometer and gyro.

the device is kept still, the total acceleration should be $\sqrt{a_x^2 + a_y^2 + a_z^2} = 9.8\,m/s^2$. The way this magnitude is distributed on the three body-fixed accelerations $a_x(t)$, $a_y(t)$ and $a_z(t)$ will give us information about the absolute rotation.

Consider the coordinate system of the setup in Figure 3.2. When the device is lying flat on a table with the battery holder facing down, we should see the measurements $a_x = 0$, $a_y = 0$ and $a_z = 9.8$ (The device is accelerating up from the table, relative to free-fall. An alternative view is that there is a force in the positive $z$-direction). When in perfect balancing mode, i.e., tilted up and standing on its wheels, the measurements should be $a_x = 0$, $a_y = -9.8$ and $a_z = 0$. The device is still accelerating up from the table relative to free-fall (normal force up from table), but $y$ is pointing downwards, hence the negative sign.

From this, it is clear that when we are balancing the device on its wheels and it is sufficiently still, the tilt-angle is related to the relative size of the acceleration components $a_y$ and $a_z$. No movement should occur in the $x$-directions, and thus $a_x$ will be 0.

In practice the MinSeg is never completely still when balancing, so $a_y$ and
$a_z$ will contain parts that come from actual movements, which will influence the computed angle, i.e., we will see noise on the angle estimate. Another problem is that the accelerometer might have been installed slightly tilted, which will give us a bias on the acceleration signals. However, since we never integrate the acceleration signals in this application, this bias will not lead to any kind of drift, but just a constant error on the angle estimate.

### 3.1.1 Gyro experiments

Double-click the icon **Copy minseg files** on the Desktop. Template files will be copied to a directory locally on the computer and a link to this directory will be created on the Desktop. Open this directory and go to the directory **Sensor**. Double-click the file `template1.slx`. If you are asked which version of MATLAB to use, you can pick either one. This **Simulink** scheme graphically describes the code that will be generated and downloaded to the Arduino micro-controller.

The first part of the lab will concentrate on the gyro, and in particular the gyro signal for rotation around the $x$-axis in the $y–z$ plane.

**Preparation 6** The largest possible angular velocity the gyro can measure is $250^\circ/s$. When the gyro senses this rate, it outputs the value $32768$ ($2^{15}$, the largest possible number that can be encoded using 16 bits, 1 bit is used for the sign). How should the measurement be scaled to go from the measured integer value, to the unit deg/s.

**Task 1** Check the value in the gain block to confirm your preparation on the measurement scaling.

**Task 2** Assemble the MinSeg as shown on the front-page, and place the MinSeg flat on the table with the battery holder facing down, and connect the USB cable. Compile, download and run the code by pressing the green run button. Study the plot scope with the uncompensated raw gyro signal. You will have to press the scale button ($\pm$) to see the signal. An alternative to repeatedly pressing the scaling button is to right-click in the plot, select **Configuration parameters**, go to the **Main** tab, and change **Axes scaling** to **auto**.

Since the MinSeg is stationary, the true angular velocity is 0, so everything you see is measurement errors (and extremely small vibrations in the table).
Make a rough estimate of the bias level, i.e., the constant error $b$ in Equation (2.1).

**Preparation 7** A simple way to reduce the bias error is to estimate its value, and then modify the measurement so that the signal varies around 0 instead. Of course, this only works if the bias really is constant (it never is in practice, it might change with temperature etc), but it is a good start. As an example, if you see the gyro signal varying around the level 70 when lying still, what value should be used in the bias compensation block in Figure 3.3?

**Task 3** Update the constant in the Bias compensation block such that the bias level found in Task 2 is compensated for. Confirm the change by looking at the plot of the compensated signal to see that it fluctuates around 0. Also study the plot showing the angular velocity in degrees per second. Approximately how large is the amplitude (in deg / s) of the fast varying measurement errors?
We are now ready to start integrating the bias compensated angular velocity measurements. To obtain the angle estimate, we integrate the signal using the approximation in preparation 3.

**Task 4** Extend your model according to Figure 3.4. The *Discrete-time Integrator* block can be found under *Discrete* in the library browser. Double-click the discrete-time integrator block in your model and change the Integrator method to Integration: Backward Euler, and change the sample-time to \(-1\) (which means it uses the sample-time 0.04 seconds specified in the model configuration). The Gain should be left at 1.0. Apply and close. You have now defined the integral approximation to be computed exactly as in preparation 3.

![Figure 3.4: Integrated bias-compensated gyro measurements.](image)

**Task 5** Start the code, and study the angle estimate plot. Note that it does not stay around 0, despite the MinSeg being completely still. This is due to a remaining bias error and what you see is the linear drift (and effects from...
noise). Find out how long it takes for the angle to drift $1^\circ$ (the time can be seen in the bottom right corner of the scope). This is a typical performance measure on a gyro implementation. The longer it takes, the better. Your solution will probably be in the order of $5 – 30$ seconds, which means that after some minutes, the angle estimate is completely wrong.

Here you definitely want to change the plot to have automatic scaling of the axis as described above.

**Task 6** Finally rotate the MinSeg so that it stands on its wheels. Confirm that the angle estimate changes $-90^\circ$.

### 3.1.2 Accelerometer experiments

Let us now turn to angle estimation from the accelerometer signals. Study Figure 3.5. If the device is stationary at an angle, the total acceleration is $g = 9.8 m/s^2$ and it is pointing downwards, and thus the accelerometer measures $g$ upwards (as it measures relative to free-fall). This acceleration signal will be picked up in the body-fixed coordinate system through the acceleration measurements $a_y(t)$ and $a_z(t)$. By geometry, we have $\theta(t) = \arctan(a_y(t)/a_z(t))$. Note that $a_y$ will be negative from the definition of the coordinate system in Figure 3.2, hence the angle will be negative in the current definition.

Of course, if the MinSeg ever was in the angle illustrated in Figure 3.5, it would not be stationary but had to be moving. This means there might be additional forces and resulting accelerations acting, but the idea here is to see those effects as noise.

**Task 7** Close the model **template1** and open the model **template2**. Double-click the gyro angle estimator and update the bias compensation to the value you found earlier. Download the code and study the angle estimate from the accelerometer signals, and compare it to the angle estimates from the gyro,
Figure 3.5: The acceleration relative to free-fall will be picked up in a stationary situation on the two coordinates $a_y(t)$ and $a_z(t)$ allowing us to compute the tilt-angle $\theta$

both displayed in the same plot, when you have the MinSeg completely still and when you tilt the MinSeg back and forth slowly. How do they compare in terms of noise and drift?

Preparation 8  To understand what you will see in the practical experiment below, consider a car analogy. You have a car with a speedometer which cannot show velocities higher than 90 km/h. You drive for 1 hours at 120 km/h. You have thus moved 120 km, but according to the measurements you have only moved 90 km. You now travel back at 90 km/h, and will thus end up 30 km away from the start. What position would computations based on the speedometer say that you are at after completing the drive?

Task 8  See what happens with the angle estimates if you rotate the MinSeg quickly and go beyond the measurement limit 250°/s in the gyro (hold the MinSeg firmly in both the motor and the board so you don't break it in the connection between the board and motor and tilt it quickly 45 degrees or so
and then back slowly). Observe the new constant level the gyro angle estimate ends up at after every tilting.

**Task 9** Move the MinSeg around while keeping the tilt angle constant (such as standing straight up and moving it left to right quickly). How do the two angle estimates perform now?

### 3.1.3 Sensor fusion with complementary filter

What you should have seen now is that the angle estimated using the gyro behaves well on a short time-frame, but drifts on a long time-frame (suffers from low-frequency errors which are integrated). The angle estimate from accelerometer signal on the other hand has no drift (good in a long time-frame), but has more noise (bad on a short time-frame, high-frequency errors). We would like to combine the merits from the two sensors, while avoiding the drawbacks of them. In other words, we trust the long-term average of the accelerometer angle estimate but not any of its fast changes, while we have no confidence in the long-term value of the gyro estimate.

Recall the estimate generated by the gyro signal. Call this estimate $\theta_g$ and consider the computation using the approximated integral.

$$\theta_g(kT_s) = \theta_g ((k-1)T_s) + T_s \omega(kT_s) \quad (3.1)$$

If the estimate ever goes bad (due to drift for instance), it will never get better as we keep using the bad value. If all movements end, the angle estimate will stay at its last value. What we could do is to bring it back to a somewhat correct value periodically. For instance, we could reset it to the value obtained from the accelerometer estimator every 10 seconds, or similar heuristics. A more clever idea is to continuously use the angle estimate from the accelerometers. Let the accelerometer angle estimate be called $\theta_a(kT_s)$, and introduce a new estimate $\theta_c(kT_s)$ which is a slight variation of
the gyro integration

\[ \theta_c(kT_s) = (1 - \alpha)\theta_a(kT_s) + \alpha(\theta_c((k - 1)T_s) + T_s \omega(kT_s)) \]  \hspace{1cm} (3.2)

The constant \( \alpha \) is typically close to 1. When it is 1, we obtain the gyro solution, and when it is 0 we obtain the accelerometer solution. What we are doing is that we are slowly pushing the estimate \( \theta_c(kT_s) \) towards \( \theta_a(kT_s) \), while still using the high-precision gyro measurements to detect rotations. Effectively, we will eliminate the drift, while retaining the good short time-frame qualities of the gyro. Using multiple sensors with different characteristics like this is called sensor fusion.

**Preparation 9** With \( z^{-1} \) denoting unit time-delay, show that (3.2) can be written as

\[ \theta_c(z) = \frac{(1 - \alpha)z}{z - \alpha} \theta_a(z) + \frac{T_s \alpha z}{z - \alpha}\omega(z) \]  \hspace{1cm} (3.3)

**Preparation 10** With \( \theta_g(z) = \frac{T_s z}{z - 1}\omega(z) \) denoting our original gyro estimate, show that the solution (3.3) alternatively can be written as

\[ \theta_c(z) = \frac{(1 - \alpha)z}{z - \alpha} \theta_a(z) + \frac{\alpha(z - 1)}{z - \alpha}\theta_g(z) \]  \hspace{1cm} (3.4)

Hence, the new solution \( \theta_c \) we propose is a sum of our two old solutions \( \theta_a \) and \( \theta_g \), each of them filtered. Let us look at the frequency response of those filters (here with \( \alpha = 0.9 \) and \( T_s = 0.025 \)s)

```matlab
z = tf('z',0.025);
Ha = (1-0.9)*z/(z-.9);
Hg = 0.9*(z-1)/(z-.9);
bodemag(Ha,Hg);legend('Ha','Hg');
```

The result is seen in Figure 3.6. We see that the filter \( H_a(z) \) used on the accelerometer angle estimate is a low-pass filter (removes fast variations and only keeps slow variations) and the filter \( H_g(z) \) for the gyro angle estimate is a high-pass filter (removes the slow drift and only keeps fast variations). The name complementary filter comes from the fact that the two filters sum to 1 (check in (3.4)!).
Figure 3.6: Amplitude gain for the two filters used in the complementary filter approach. The accelerometer solution is low-pass filtered, while the gyro solution is high-pass filtered.

Preparation 11  In the block diagram in Figure 3.7, values and connections are missing. Insert the values $\alpha$, $1 - \alpha$ and $T_s$ in suitable blocks, and draw the connection, to implement the computation in (3.2). As a hint, standard Euler backwards as in (3.1) is implemented in Figure 3.8.

Task 10  Close template2 and open template3 and set the two gains on the accelerometer signals as specified in the text next to them. This operation is done to account for the fact that we currently have two different types on IMUs, and is not related to any theory described here.

Task 11  Set the bias compensation in the gyro angle estimator.

Task 12  Open the Complementary filter block, and complete it according to the preparation exercise. Use $\alpha = 0.9$ and $T_s = 0.04$. Start the code and study the complementary filter estimate compared to the previous estimates. Try
Figure 3.7: Insert missing values and connection to implement the complementary filter (3.3)

![Diagram of complementary filter implementation]

Figure 3.8: Example showing how discrete-time integral approximation (Euler backwards) would be implemented manually,
\[
Out(kT_s) = Out((k-1)T_s) + T_s \text{In}(k)
\]

other values on \(\alpha\) such as 0.1 and 0.99 so see how this choice affects noise sensitivity and how well the complementary filter estimate follows actual movements of the robot.

Task 13  Test the robustness of the angle estimate (with a sensible choice of \(\alpha\) such as 0.9) by using a completely wrong value on the bias compensation (add an extra 50 or so).

The angle estimate from the accelerometer signals are based on the raw accelerometer signals. An acceleration of 1 g leads to a measurement of 8192
while no acceleration should lead to the measurement 0. In our computation of the angle based on these signals, we have not scaled the data to $m/s^2$, as the scaling cancels out when we perform the division before taking arctan. However, we should compensate for calibration error / bias in the accelerometer signals.

**Task 14** Extend the model in the **Accelerometer angle estimator** block to have bias compensation on both $a_y$ and $a_z$ as in the **Gyro angle estimator** block. When flat on the table, $a_y(t)$ should be 0, and $a_z(t)$ should be 8192 (the accelerometer is built such that it outputs an integer between $-2^{15}$ to $2^{15}$ when going from -4 g to 4 g, thus 1 g will be $2^{13} = 8192$). After calibration does the accelerometer based angle estimate and complementary filter estimate give $0^\circ$ when flat and $-90^\circ$ when standing straight up (hold it against the computer to get it straight)?
3.2 Summary and reflections

Summarize and reflect on what you have seen and learned in this lab.

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
</table>
| 1. A gyro measures | an angle  
an angular velocity  
an angular acceleration |
| 1. An accelerometer measures | an angle  
a velocity  
an acceleration |
| 2. A low-pass filter | removes high-frequency content  
removes low-frequency content |
| 3. A high-pass filter | removes high-frequency content  
removes low-frequency content |
| 4. Accelerometers are suitable for computing the angle | when moving very slowly  
when moving around a lot |
| 5. Gyros are good for | finding changes in angle  
finding the absolute angle |

Most unclear to me is still: