Determining Angle from Accelerometer and Gyro Data Using a Complimentary Filter

Objectives:

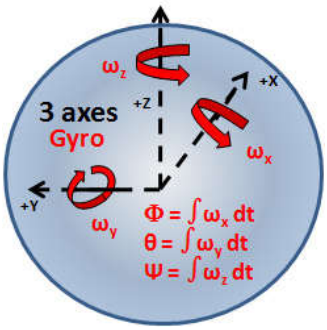
- Determine angle using a gyroscope
- Determine angle using an accelerometer
- Create a "Complimentary Filter" utilizing weighted input from both sensors
- Use the complimentary filter with an angle controller

Background:

In previous labs data from the gyroscope was used to track angular rotation, but the measured angle could "drift", even while the board was not moving. In this lab the accelerometer will be used to calculate angle, and both the gyro and accelerometer results will be combined in a complimentary filter to obtain the benefits of both.

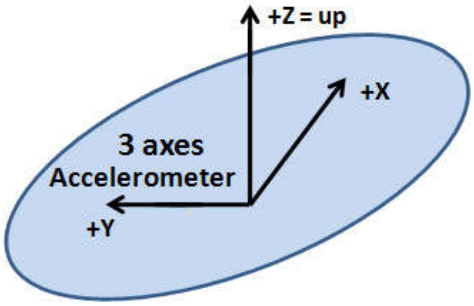
Gyroscopes

Gyroscopes utilize the principle of angular momentum to accurately measure rotation rate about an axis. Thus, multiple gyroscopes can be used to measure the angular rotational rate of an object in space. However, gyroscopes cannot inherently tell which was is "up" in the world, and techniques for computing position may result in "drift".



Accelerometers

Accelerometers measure the stresses caused by the movement of masses during linear acceleration or deceleration. Utilizing an array of these elements in three axes, we can effectively track linear movement in a 3D space. Because of the constant downward acceleration force of Earth's gravitational field, accelerometers can be used to determine "down" in the world.



Using both sensors in one application can leverage the strengths of both sensors, while eliminating the drawbacks. One real world example of the use both sensors is the Nintendo® Wii Remote. After its launch in 2006, many users observed that the "WiiMote" was great for sensing up waggle, but couldn't accurately track more complex motions including rotation. This was because the WiiMote only relied on a 3-axis accelerometer for motion tracking. In 2009, Nintendo released a "Wii Remote Plus" add-on that finally enabled "True 1:1 motion tracking" in 6 DOF. The core of this add on was a gyroscopic sensor.

Part 1: Computing Position Angle with Gyroscope Data

First, create a new program for angular measurement using the gyroscopic data.

Create the Simulink diagram below.

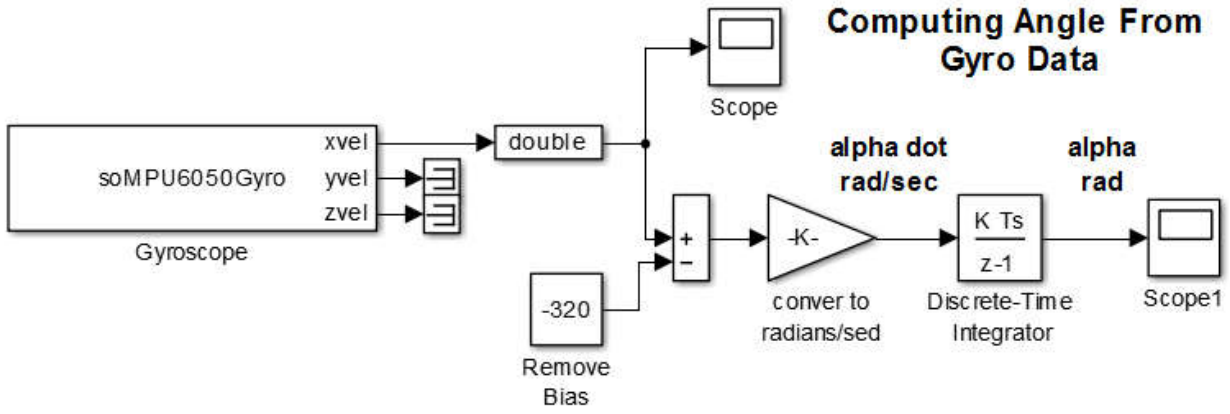
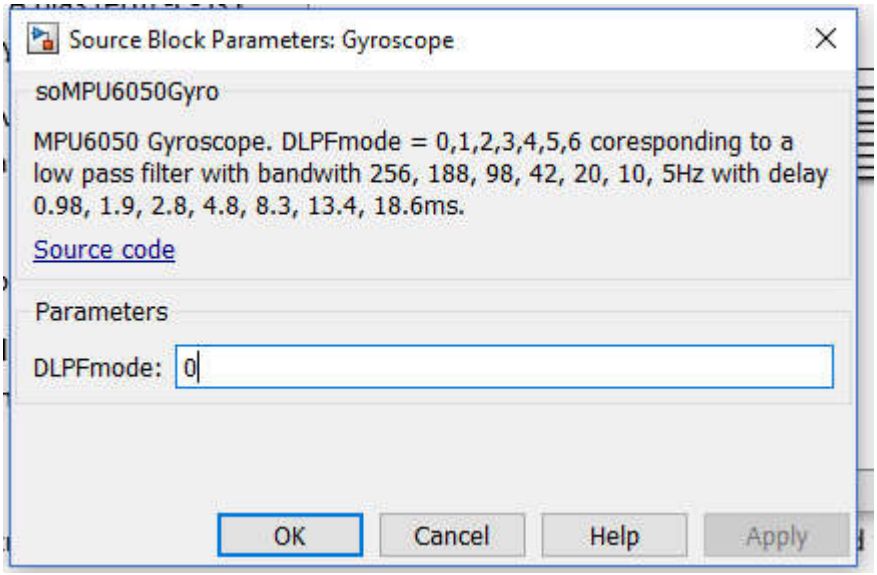


Figure 1: Calculating angle from gyroscope data – use the gyroscope block from your hardware system

Above, you can see that the input from the Gyro Driver is being integrated in order to produce the final Scope1 output. This also means that if the hardware does not read exactly zero while at rest, the small values will cause a steady increase or decrease in the reading. This is the gyro bias. A bias term -C- is used to removed before integration. You must manually "calibrate" the system by experimentally finding a value to remove this bias.

Even after removing the bias, over time, the initial bias will change. This is known as gyro "drift", and eventually something will need to be done to correct for it during longer periods of time (minutes).

The Gyroscope block is a special block. Double click the Gyroscope block:



Next open the file "MPU6050.cpp" in the "RASPlib\src" folder and search this file for the command "setDLPFmode" and see what the command "accelgyro.setDLPFmode(5)" does.

Checkpoint 1:

- Demonstrate a functioning system that accounts for drift. Rotate the board 90° around the x(??) axis, and observe the result in Scope1 to verify your angle calculation from the gyroscope.
- Repeately move the board from 0 to 90 degrees – is the angle measuement still accurate?
- Explain what “accelgyro.setDLPFmode(5)” does.

Part 2: Computing Board Angle with Accelerometer Data

In this exercise board angle is calculated using the accelerometer data by observing the components of the constant gravitational force (which is always "down") on each of the accelerometers axes.

Data from 2 acceleometer axes (Y_acc and Z_acc) are used to calculate the angular position. The downward acceleration from the earth will always be fairly constant: if the board measures a Z-component of 9.81m/s^2 but nothing in the Y-component, it knows it is lying flat. If the Z-acceleration decreases, but the Y accelerometer measures some portion of this downward, we can calculate the angle using an arctangent block.

Create the Simulink diagram below.

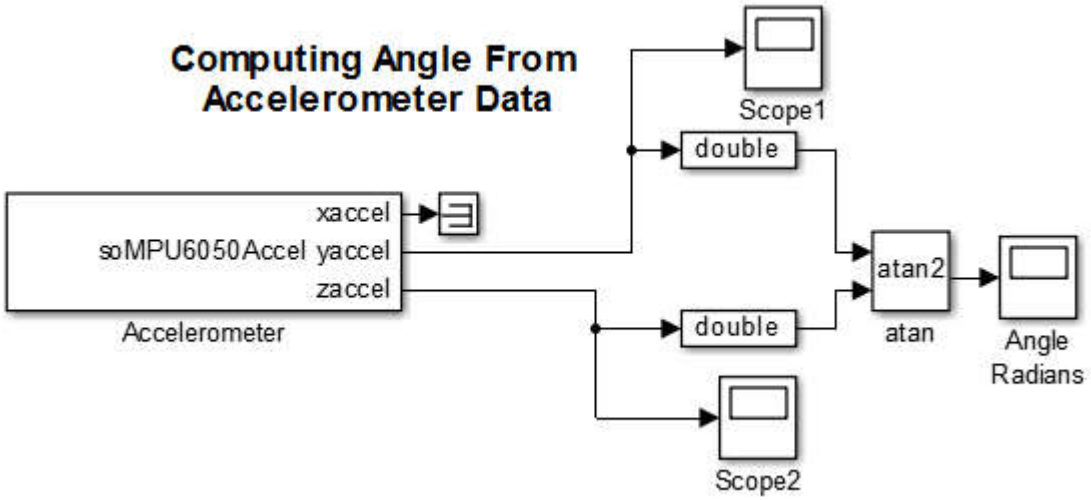


Figure 2: Calculating Angle from Accelerometer Data

Checkpoint 2: Run the system. Rotate the board 90°, and observe the result in the scope to verify that the angle is calculated correctly.

Discussion Questions:

- Do the measurements seem valid for all angles through a rotation?
- Repeately move the board from 0 to 90 degrees – is the angle measuement still accurate?
- How does the noise of the signal compare to that of the gyroscope function?

- If the accelerometers are being used to track the known gravitational acceleration, what happens to measurements when the board is accelerated in a linear direction?

Part 3: Creating the Complimentary Filter

The previous two measurement circuits can be combined into one measurement system to utilize the characteristics of the two sensors: the gyroscope can measure quick changes in rotation, but has steady state error. The accelerometer can accurately obtain the steady state angle. A simple method of combining these two measurements is called a Complimentary Filter.

Create the Simulink diagrams below.

Note: The "GoTo" blocks are used to connect signals without actually having the "wires" drawn on the Simulink diagram.

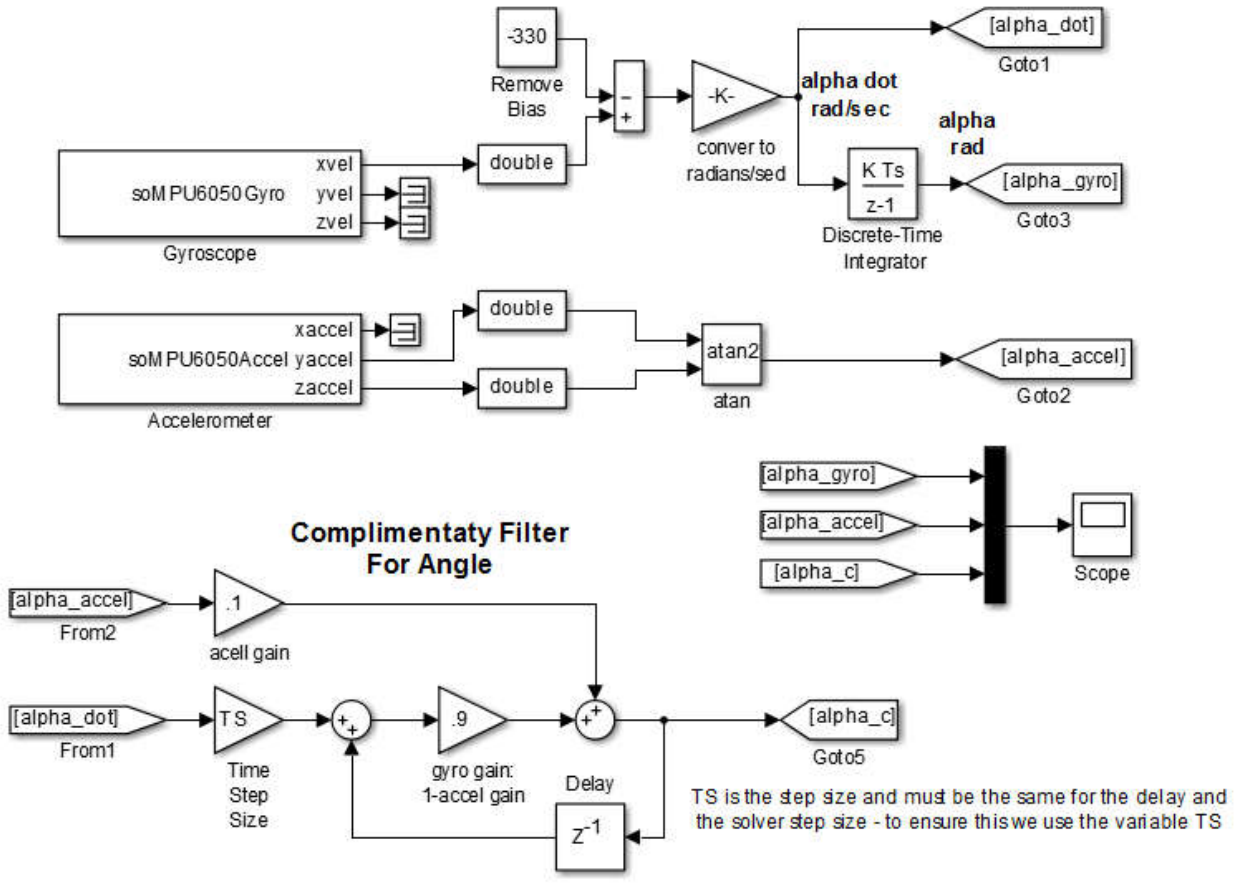


Figure 3: Components of a complimentary filter (detailed view)

The gyroscope only needs a small correction to compensate for drift accumulation. So most of the angle is measured from the gyroscope, in this case 90%. But to account for any drift that may accumulate 10% of each calculation comes from the accelerometer. The weights (0.1 and 0.9) can be adjusted to provide a trade off between speed of resposne, and steady state error, but their sum must add up to 1.

Now, select the components of the detailed diagram above, then right click and create "Create subsystem from selection". This will create a block that can be easily inserted into code for other applications.

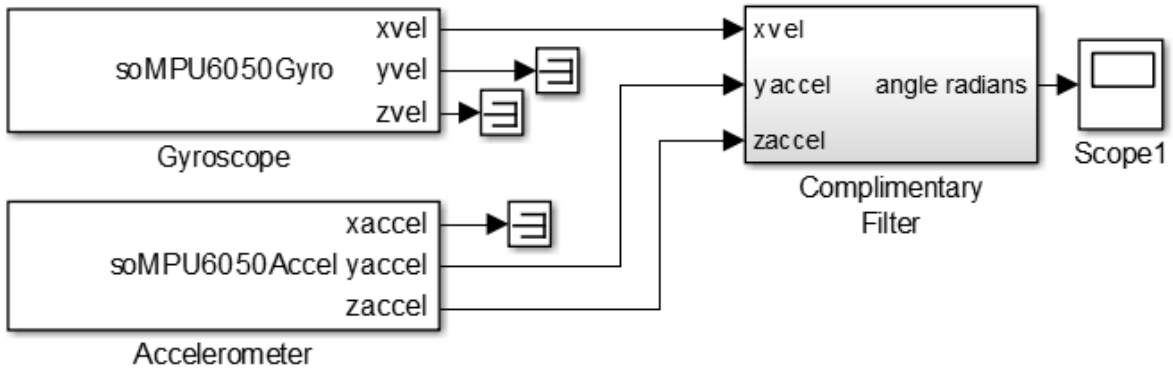


Figure 4: Complimentary filter with subsystem block

Checkpoint 3: Run the system. Rotate the board in several directions, and observe the outputs.

Discussion Questions:

- What happens to the readings as you adjust the accelerometer and gyroscope gains? What is the system like if their values are reversed (0.9 and 0.1 resp.)?
- Provide a screenshot of the scope showing all 3 angle calculations as you rotate the board.

Part 4: Using the Complimentary Filter for Position Control

Now, we will revisit a position control diagram and apply the Complimentary Filter you have just created. In this case, we will be measuring position, and then outputting this data to a PID controller and motor driver. What do you think the resulting system will accomplish?

Create the Simulink diagram for Closed Loop Position Control below:

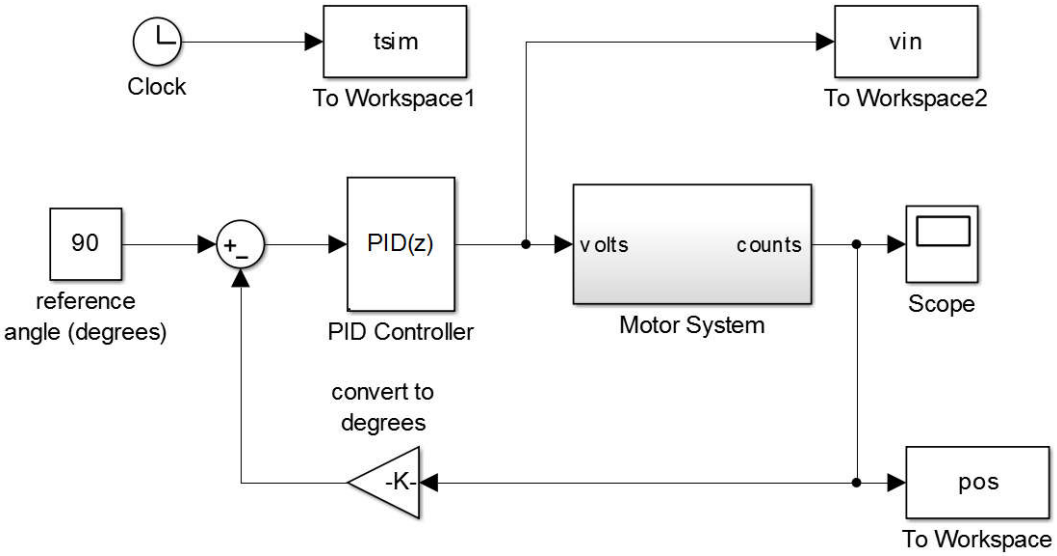


Figure 5: Closed Loop Position Control Diagram

Be sure to choose the correct feedback value K for converting from encoder counts to degrees.

Now, insert the complimentary filter you have created to complete the system below.

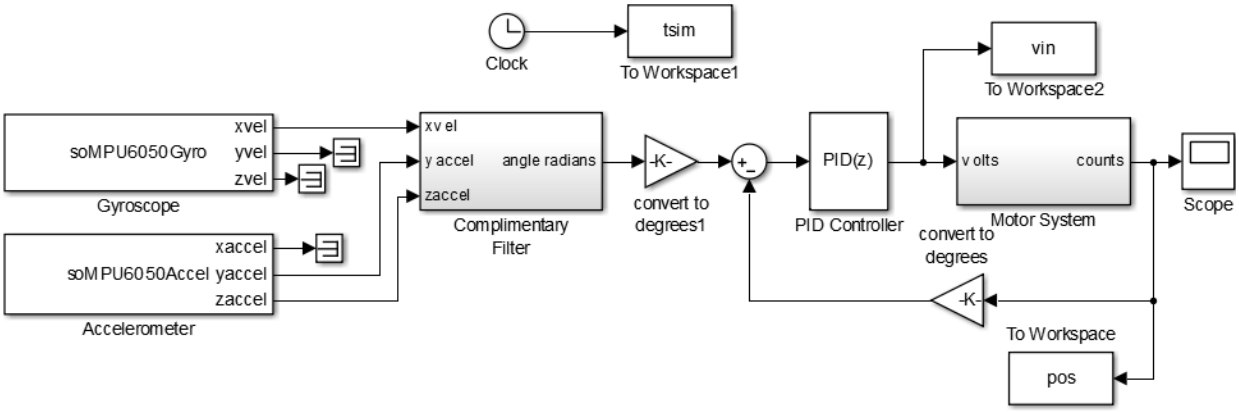


Figure 6: Final System. Closed Loop Position Control utilizing a Complimentary Filter

For this exercise, Proportional control will be sufficient. Double click the PID controller block, and enter a P value of 0.1. The (I) and (D) terms remain 0.

Checkpoint 4: Run the system. Rotate the board in several directions, and observe the resulting wheel behavior. Try sticking a post-it note or other marker on a wheel and test the system.

Discussion Questions:

- You can try modifying the Accel/Gyro ratio to observe the difference in behavior.
- This is also an excellent system to test out PID control. Note the system behavior as you change input values P, I, or D in the controller block.