# Accelerometer Calibration Guide

The following guide shows how one calibrates the accelerometer. This guide assumes

1. You’ve installed the support package
2. Tested out external mode and everything works fine

This guide follows this calibration technique described in an application note from digi-key.

<http://www.digikey.com/en/articles/techzone/2011/may/using-an-accelerometer-for-inclination-sensing>

Essentially, the raw accelerometer data is the true data multiplied by a gain factor and adding it by an offset. We need to find what the gain and offset values are for all three axes. To do this, we orient the drone in different positions and measure what the accelerometer value.

**Why is accelerometer calibration important:**

The accelerometer is used for the Attitude Determination System, it’s vector is used as a point of reference to help guide the drone into knowing which way is pointing down – this is helpful for roll-pitch determination. The rate gyros can only provide the rate of change of orientation, integrating this raw value will result in massive drift. If the accelerometer is slightly off then the ‘true’ angular roll-pitch reference point is not the actual reference point and you may see the drone slightly biased to pitch or roll in a certain direction. This could also lead to slow instability as the attitude control integrator term will eventually saturate imposing more control effort than what is actually needed.

**Calibration code:**

Inside Calculate\_Offset\_Values.m, there is some equations which calculate gain/offset values. In each section of code, we orient the drone in a certain direction such that the accelerometer sensor is either parallel with gravity (ie: its direction faces the same way as gravity) or anti-parallel. Here’s a snippet of code

%+-1G for the X-axis

up\_data = [0.9453]; %parallel (axis facing the same way as gravity)

down\_data = [-1.03]; %anti-parallel (axis facing opposite of gravity)

Offset\_calc = 0.5\*(up\_data + down\_data);

Gain = 0.5\*(down\_data - up\_data) ;

Accel\_X\_Offset =Offset\_calc;

Accel\_X\_Gain =Gain;



For the ‘down\_data, we orient the drone this way, anti-parallel to the gravity vector. The red arrow is the positive X-axis. For the ‘up\_data’ we orient the drone in the exact opposite way with the drone pointing down (ie: red arrow facing down).

Similarly for the y-axis



%+-1G for the Y-axis

up\_data = [1.039]; %parallel (axis facing the same way as gravity)

down\_data = [-0.9867]; %anti-parallel (axis facing opposite of gravity)

Offset\_calc = 0.5\*(up\_data + down\_data);

Gain = 0.5\*(down\_data - up\_data) ;

Accel\_Y\_Offset =Offset\_calc;

Accel\_Y\_Gain =Gain;

Holding the drone in this orientation will give us the down\_data value. Holding it in the opposite direction (ie: red arrow facing down) will give you the up\_data value.

Do this repeatedly until you have 6 values, 2 values for each axis which will give you the range of +g and –g of each accelerometer sensor. The z-axis is a special case in which it’s up\_data is actually positive instead of negative.

**Steps to record data and incorporate calibration values:**

1. Run Record\_accelerometer.slx model whole having the drone in a fixed orientation. This will record the accelerometer data into a MAT file called Recorded\_Accelerometer
2. Run PlayBack\_Data.slx which will play the data back and also do a time running average of the accelerometer data
3. Run the script to do the calculation
4. Update the accelerometer’s data dictionary on line 31 in the AR\_DRONE\_SCRIPT.m MATLAB file

Accel\_Calib\_Offset\_Vector = [-0.0423 0.0261 0.0388];

Accel\_Calib\_Gain\_Vector = [-0.9877 -1.0129 -1.0082]; %note, that this might need to be multiplied by -1

%depending on how the

%calibration. this is

%becaues -g is up and +g

%might be down if you

%look at the raw data