

Automatic Calibration
Sean D'Epagnier
October 4, 2008

Table of Contents

1	Introduction.....	2
2	Filtering.....	3
3	StillPoints.....	4
4	Spherical Calibration.....	5
5	Rotated Ellipsoid Calibration.....	6
6	Quaternion Alignment Calibration.....	7
7	Box Alignment Calibration.....	8
8	Laser Alignment Calibration.....	9

1 Introduction

I came to the idea of auto-calibration when I realized the magnitude of my 3-axis accelerometers is constant when the device is not moving. This fact alone essentially allows me to auto-calibrate the sensors and recalibrate them when the device is in use.

2 Filtering

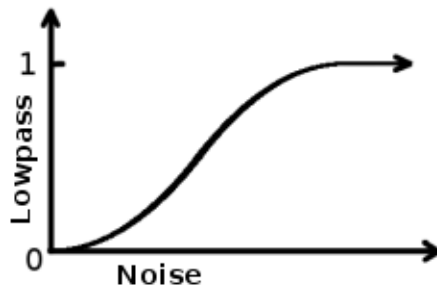
Without filtering, the data from the sensors is raw. This would be ok if the raw data were good enough to achieve the level of calibration desired; I have found with filtering I find a 10-20x improvement in calibration accuracy. This is because many samples can be averaged over a longer period of time to give a higher resolution result, however this is only useful when the device is still.

There are a lot of ways to filter data. A typical lowpass filter simply computes the weighted average of the newest sample with the running average, if y is filtered data, and $lowpass$ is the lowpass constant:

$$y_n = (1 - lowpass)y_{n-1} + (lowpass)x_n$$

The trouble with this filter is that if the $lowpass$ is too close to zero, it will react too slowly to large changes, and if it is too close to one, it will not effectively filter the data when the sensor is stable (device is not moving)

My calibration algorithms largely rely having the device perfectly still. This allows me to average a lot of samples to get a very good reading, but I also need to react quickly if the device is moving, so I devised the quadratic filter. This filter is the same as above, except the $lowpass$ constant is computed based on the quadratic equation:



It is essentially part of two parabolas and a line joined together so that the derivatives are continuous.

As the sensor noise goes up, we tend to just use the raw data, but when it is stable we average more and more samples. Having the derivatives continuous allows the data to make smooth transitions when the noise level changes. (This is especially important since it is the filter used to compute mouse coordinates for the pointer device)

3 StillPoints

A stillpoint is a raw measurement taken when the device is not moving. This is determined by looking at the noise levels on the sensors. When the device is still, there are 3 important assumptions:

- The accelerometers should only be measuring gravity.
- The magnetic sensors get a good low-noise reading for just the earth's magnetic field.
- The angle between the accelerometer and magnetometer readings is the inclination angle and is constant for a given geographic location.

The first assumption is used to calibrate the accelerometers, the second for magnetometers (note: the device measures just the magnetic field if it is moving or not.. the unfiltered data is used for the magfast calibration which works even if the device is moving) The last assumption is used to calculate the misalignment between the accelerometers and magnetometers. It is optional also to use the last assumption to also compute calibration for the magnetometer.

4 Spherical Calibration

Spherical calibration only computes biases for each of the 3 sensors, as well as a single scalefactor for all 3 axis. This is used for the accelerometer and magnetometer (magfast calibration) These algorithms are implemented in the source code. See (fit.c) for the actual implementation.

The equation used is: $(x - a)^2 + (y - b)^2 + (z - c)^2 = d^2$

The unknowns a, b, and c are the biases for the x, y, and z sensors. The unknown d is the overall scalefactor.

To estimate these unknowns, recursive least squares is used. This is a simplification of a kalman filter. The following equations are used:

$$K = P_{k-1}H^T(H P_{k-1}H^T + R)^{-1}$$

$$X = X_{k-1} + KZ$$

$$P = (I - KH)P(I - KH)^T + K R K^T$$

- X – The current state, so for the case of 4 unknowns, it is a 4-vector.
- P – The covariance, a 4x4 matrix.
- I – The identity matrix.
- R – The confidence factor of the new measurement.
- H – The partial derivatives of the above equation with respect to each unknown. For example, for the spherical truth equation: $H = [-2(x - a), -2(y - b), -2(z - c), -2d]$
- Z – The residule of the measurement. For the spherical truth equation it is $d^2 - (x - a)^2 - (y - b)^2 - (z - c)^2$

Whenever there is a new measurement, the above recursive least squares equations are applied.

5 Rotated Ellipsoid Calibration

The magnetometer differs from the accelerometer in that it suffers from soft-iron distortion, crosscoupling, large differences in scale factors between axis (typically 10%) as well as sensor misalignment.¹ The soft-iron calibration strategy taken is simplified as it essentially treats the sensors as soft-iron, not the metals around it. This means the soft-iron calibration is not complete, and the overall accuracy is reduced with additional iron. Soft iron distortions show up as cross-coupling errors, so these are both handled by estimating cross-coupling coefficients. I believe that it is possible to do a more complete model of the soft-iron distortions using tensors, however I would need a faster microprocessor, and gyros to attempt this calibration. The current calibration is significant; for example, with a given amount of iron, the heading errors are as much as 10 degrees with basic calibration (only bias and overall scalefactor) where with the calibration errors are reduced to 1 degree with scalefactor ratios, cross-coupling, misalignment factors.

The equations used:

$$m_x^2 + m_y^2 + m_z^2 = 1$$

$$m_x = X_0(x - X_6)$$

$$m_y = X_0(X_1(y - X_7) + X_3(x - X_6))$$

$$m_z = X_0(X_2(z - X_8) + X_4(x - X_6) + X_5(y - X_7))$$

By plugging in for m_x, m_y, m_z into the first equation, the overall equation is calculated. The same technique of recursive least squares is used, except there are 9 unknowns instead of only 4.

In addition to recursive least squares, normal least squares is used as well which gives better results. Whenever the device is still, the raw sensor measurements are stored. Once enough measurements are stored, the state is updated with the equation:

$$X = X + (J^T J)^{-1} J^T R^T$$

- X – the current state
- J – the matrix with the partial derivatives for each point
- R – the vector of residues for each point

¹ It would be possible to solve for crosscoupling and scalefactor ratios in the accelerometer, but due to the extremely low values estimated, they are below the noise level of the sensors, therefore it is unhelpful.

6 Quaternion Alignment Calibration

The above chapters describe how to compute normalized orthogonal measurements for both magnetometer and accelerometer given enough good datapoints. Due to manufacturing imperfections the accelerometer is not in the same coordinate space as the magnetometer. They may be misaligned typically as much as 2-3 degrees. To correct for this error, we must calculate a rotation that needs to be applied to the magnetometer readings to put it into accelerometer (or sensor) coordinates.

A unit quaternion can be used to represent a rotation in 3-space. Because of this, we solve for a unit quaternion for our misalignment. Given a quaternion q , and vector m , we can apply the quaternion rotation to the vector with the quaternion triple product:

$$qnq^t$$

n is a quaternion which is $\langle 0, m_0, m_1, m_2 \rangle$

The result is a quaternion, the last 3 parts is the rotated vector.

From this known fact, we can derive an interesting truth equation:

$$g.qmq^t = d$$

- g – the calibrated accelerometer measurement (gravity)
- m – the calibrated magnetometer measurement
- q – the unknown rotation to get from magnetometer to sensor coordinates
- d – the sin of the inclination (constant for a given magnetic latitude)

The unknowns are q and d . q is a quaternion with 4 parts, but we force it to be unit, so the first part is computed from the last 3. This leaves us with only 4 unknowns, and the recursive least squares method and regular least squares are used to compute the coefficients.

Note: It is also possible to use this equation to estimate all but overall magnitude for magnetometer calibration, as well as the biases for the accelerometer. Typically this calculation is disabled because these coefficients can already be calculated without it, and it can have adverse effects since it is effectively using one sensor to calibrate the other.

7 Box Alignment Calibration

Box alignment calibration is used to align the sensors to whatever coordinate frame is being used. This is represented by a quaternion rotation. There are two methods used to compute this rotation:

- Rotation calculated when the sensors are told they are level and facing north. First the rotation to make the accelerometer point down is applied, then the rotation to rotate to make the x axis face north is measured and combined with the first rotation and this is the box alignment.
- The second method is if the direction of north is not known. In this case, the alignment to make the accels point down is applied, then we need data of the x axis pointed straight up. This allows us to compute the alignment for the yaw rotation using only the accels. These rotations are combined and this is the box alignment. Notice that the second method does not use the magnetometer, so it works even before the magnetometer is calibrated.

8 Laser Alignment Calibration

Laser alignment is a rotation to get from the sensor coordinates to align to an axis or vector. This means roll is ignored because we only want to align to the vector, the rotation around the laser is irrelevant. To compute laser alignment, we use the truth equation for multiple shots rotated around the laser axis:

$$a_x n_x + a_y n_y + a_z n_z = 1$$

- a – The Calibrated normalized accelerometer sensor reading for the shot.
- n – The rotation from this vector to $\langle 1, 0, 0 \rangle$ is the laser alignment calibration quaternion. In polar coordinates n is two angles plus a magnitude. The two angles are needed for the alignment rotation, the magnitude is $1/\cos(\text{incline})$. Incline is the incline angle that is used for the shots, and is arbitrary, it is not used.

Notice that the magnetometers are not used for laser alignment calibration, so the laser can be aligned before the magnetometer is calibrated. The recursive least squares method can be used to find n .