Sparton Navigation Modules

Integration and Calibration Guide

Rev: 2.11.16

Applicable to Sparton AHRS-8, GEDC-6E, and DC-4E

REVISION HISTORY:

DATE	REV	DESCRIPTION OF CHANGE	INITIALS
08/09/13	A	Original	RSW
12/03/14	B	Added apr_tare script and information on how to perform it.	RSW
06/08/15	C	Added NMEA timing and wording to describe gyro-only mode and resetting of the field calibration prior to starting field cal.	RSW
06/24/15	D	Added sample algorithms to Appendix C	RSW
11/17/15	E	Added wording for scripts while sending large files. General spelling/grammar errors.	RSW
01/20/16	F	Modified section s for clarity and fixed grammatical errors	RSW

Table of Contents

1 Introduction

This document is intended to aid a user in the integration and installation of the Sparton inertial sensor. This document outlines procedures for that purpose. This document applies to the AHRS-8, GEDC-6E and DC-4E devices. In this document, these devices will be referred to as inertial systems and all references shall apply to all devices except where noted.

1.1 Scope

This document covers the following topics:

- In-field Calibration of the Sparton Inertial Systems
- Virtual Mechanical Alignment of the Sparton Inertial Systems

This document is not a hardware or mechanical data sheet.

1.2 References

This document is an umbrella document in that some of the interface descriptions for the inertial systems are described in external documentation. This document describes the relationship of these documents. The additional documentation needed to complete the entire interface description for the inertial system family is listed below.

- *NorthTek System,Programming Manual*

1.3 Applicability

This document applies to the Sparton AHRS-8, GEDC-6E and DC-4E Inertial Systems. All portions of this document that refer to gyro functionality only apply to the AHRS-8 and GEDC-6E.

2 Overview

2.1 Protocol Introduction

The interface to the inertial system is a serial communication link (User Port). The link is an asynchronous character oriented interface that operates in a bidirectional mode. This serial link transmits and receives asynchronous characters using nominal "UART" type framing with 8 data bits, no parity and a single stop bit, with a default baud rate of 115200. The baud rate may be changed by software command. No error detection is used with the individual characters.

The serial link is full-duplex; however in practice the information flow is largely half-duplex in nature. Essentially the client and the server operate in a poll-response fashion. For example, in the above situation, the device that desires to use the inertial system would be the master (client) and the inertial system would be the slave (server).

3 How do I setup the sensor

Sparton Inertial Systems do not require typical setup to work but we do have evaluation software that requires installation. Each environment is different so there can be tweaks made to ensure the best possible accuracy if needed.

3.1 Where should I place the sensor inside my Device

Due to the nature of the Sparton Inertial System, placement of the sensor inside the host device may not be as easy as finding the space to fit. When selecting the location, it is necessary to find a spot where the magnetic disturbance is at a minimum.

To characterize the host system and to find a good location for placement of the Inertial System, it is recommended to take data from the sensor at different positions inside the host device. During this profile you will be determining the magnetic signature of the host device and its components. The following procedure describes how to collect the data for one position:

1. Prior to data logging please log the output of the following commands and save to a text file and the reset the device by removing power or sending the '*reset*' command..
   save_mask d.on<cr>
   db.print<cr>

2. Send the following commands and log the data:
   a. AHRS-8, GEDC-6E, DC-4E
      1 magp.p<cr>- Sending/Pressing CTRL-q and CTRL-S will start/stop the output.

   This is the magnetometer sensor data output running at gyroSampleRate. What you are looking for is that the data has a low standard deviation/variance and that the DC offset, if any, is static and does not drift over time. If the DC offset of the magnetic signature is time-varying, then the previous In-Field Calibration will no longer apply.

3. This dataset should be a profile of the host targeting system. Take care to note what is being done during each log. Turn on/off individual components if possible: camera zoom, camera focus, GPS acquiring/tracking. This will allow a user to know at which times the output is more accurate and which internal systems will affect the magnetometer readings.

4 Virtual Alignment Procedures

For an accurate measurement it is imperative that the Sparton Inertial System be aligned properly with the host system; in other words, to be pointing towards where the end device is pointing.

Sparton has the ability to adjust the Rotation Matrix (boresightMatrix) on the fly. This allows the sensor to be mounted into a near infinite number of orientations and be virtually rotated to match the host system. While the gyros do not need to be at the center of rotation for each axis, the accelerometers will experience centripetal accelerations which will affect the pitch/roll and effectively the heading. In static environments there is less of a concern with this issue.

Measurement due to misalignment is very problematic for some applications. An example of how misalignment error affects heading is as follows:

	misalignment		
	pitch	**roll**	**Yaw error**
10° Pitch / 10° Roll	0.05	0.05	0.017638
20° Pitch / 20° Roll	0.05	0.05	0.036418
30° Pitch / 30° Roll	0.05	0.05	0.057785
45° Pitch / 45° Roll	0.05	0.05	0.100124
60° Pitch / 60° Roll	0.05	0.05	0.173467
70° Pitch / 70° Roll	0.05	0.05	0.275197

4.1 AHRS-8, GEDC-6E, DC-4E

If using the GEDC-6E, you might need to perform an InvokeGyroOffsetCal before running the virtual alignment procedure. This will calibrate out any gyro offsets built up due to temperature and other effects (The AHRS-8 does not have this issue). Both compasses perform in-field gyro offset calibration consistently during normal operation but due to the sensitivity of the virtual alignment procedures, it is recommended that this be done before boresighting the GEDC-6E.

4.1.1 Tare Procedure using North Reference

This procedure is the simplest procedure but the user needs to know the North Direction to a high degree of accuracy without using the Sparton Device for Northing.

- Point the Host System to North and Zero degrees Pitch and Roll.
    - Do not use the Sparton sensor for this procedure as this procedure is meant to align the Sparton with the host system.
- Open the NDS-2 Software.
- Go to Device Settings->Tare Device
    - This will finish in about one second and the Virtual Alignment Procedure will be complete

- If you are using a terminal emulator or a microcontroller, after the system has been pointed to North and leveled you would send the command *InvokeTare 0 set drop* to run the tare procedure.

4.1.2 Tare Procedure using a Known Azimuth, Pitch and Roll

When using scripts this large you must set up the system as follows:

- **Insert a delay per line of 5ms.**
- **Ensure each line "to be processed" (not comment lines) is less than 60 characters (bytes).**

This procedure was created due to some customers not knowing North Direction but knowing another azimuth with a high degree of accuracy. The best results will occur if the roll is kept at 0.0°. Any error in the pitch or roll of the host system will compound as describe in the previous section.

- Point the Host System to a known azimuth, pitch, and roll.
- Open the NDS-2 software.
- Go to File-> Send NorthTek Script
- Browse to the location of the script (apr_tare.4th) and select the file.
- Click the Open button.
- Scan the terminal window to make sure the word "Huh?" does not show. This will allow us to make sure the script loaded properly.
- Type in the command with the know azimuth, pitch, and roll as follows:
    - For azimuth/pitch/roll of 191.4/49.5/0.0 type "f191.4 f49.5 f0.0 apr_tare<enter>"
- The Boresight Matrix will print out and the operation will be complete

4.2 AHRS-M1

4.2.1 Tare Procedure using North Reference

This procedure is the simplest procedure but the user needs to know the North Direction to a high degree of accuracy without using the Sparton Device for Northing.

- Point the Host System to North and Zero degrees Pitch and Roll.
    - Do not use the Sparton sensor for this procedure as this procedure is meant to align the Sparton with the host system.
- Open the NDS-2 Software.
- Go to Device Settings->Tare Device
    - This will finish in about one second and the Virtual Alignment Procedure will be complete

- If you are using a terminal emulator or a microcontroller, after the system has been pointed to North and leveled you would send the command *InvokeTare 0 set drop* to run the tare procedure.

4.2.2 Tare Procedure using a Known Azimuth, Pitch and Roll

When using scripts this large you must set up the system as follows:

- **Insert a delay per line of 5ms.**
- **Ensure each line “to be processed” (not comment lines) is less than 60 characters (bytes).**

This procedure was created due to some customers not knowing North Direction but knowing another azimuth with a high degree of accuracy. The best results will occur if the roll is kept at 0.0°. Any error in the pitch or roll of the host system will compound as describe in the previous section.

- Point the Host System to a known azimuth, pitch, and roll.
- Open the NDS-2 software.
- Go to File-> Send NorthTek Script
- Browse to the location of the script (apr_tare.4th) and select the file.
- Click the Open button.
- Scan the terminal window to make sure the word "Huh?" does not show. This will allow us to make sure the script loaded properly.
- Type in the command with the know azimuth, pitch, and roll as follows:
  - For azimuth/pitch/roll of 191.4/49.5/0.0 type "f191.4 f49.5 f0.0 apr_tare<enter>"
- The Boresight Matrix will print out and the operation will be complete

5 In-Field Calibration – Procedures and Protocols

In the presence of Hard and Soft Iron magnetic distortions, the device must be calibrated for use in each specific environment. For a hard or soft iron distortion that exceeds the Earth’s local magnetic field, the resolution of the heading will be reduced. The greater the field strength of the offending part, the less resolution and accuracy of the heading.

To explain more fully, this is a platform calibration. Static distortions that are rotating together with the Sparton Inertial System can be calibrated out provided that they are not saturating the magnetic field sensors. Only those distortions that have enough magnetic field strength to be detected by the sensors need to be calibrated out. Since we cannot “see” the magnetic field strengths of any material, you must allow the Sparton Sensors to “show” you what is affecting them. An easy way to perform this task is to log or watch the magnetic sensor readings while moving items in your device closer to the system and then slowly moving them away until the magnetic effects detected by the sensors are removed.

5.1 AHRS-8, GEDC-6E, DC-4E

After the first calibration point is taken, the previous magnetic values will be removed from the magnetic-based yaw/heading calculation. This will show up as a heading drift in the output. Since the magnetometer values are being calibrated at this point, it is assumed that these previous values are not correct. To alleviate this drift, you can do one of the following:

1. Reset the magnetometers back to their original value before performing the in-field calibration. To do this, modify the following variables prior to beginning the in-field calibration.
    a. restoreFieldCal set to ‘1’
    b. restartCompassCalcs set to ‘1’
2. Turn off the magnetometers and rely only on the accelerometers and gyros for during the calibration point selection. Then turn then back on once the calibration is complete.
    a. *km0 f0.0 set drop* to turn them off.
    b. After calibration reset them to their previous value. *km0 fx.x set drop*; where x.x is whatever previous value was stored in km0 as a float.
        i. example km0 = 0.01, *km0 f0.01 set drop*

In the following procedures, we describe how to calibrate out those distortions that cannot, for whatever reason, be removed from the host device.

5.1.1 Procedures

Field calibration involves multiple steps:

1) Select which mode: 3D or 2D.
2) Start the calibration
3) Capture Points
4) End point capture
5) Allow the calibration algorithm to converge.
6) Terminate calibration, causing values to be stored.

The compass is already factory calibrated in an environment free from magnetic distortions. When the compass is first used in the application, it must learn the local magnetic distortions.

5.1.2 Protocols

5.1.2.1 NorthTek

NorthTek is not a protocol per se. NorthTek is a programming language, a command interpreter, and an execution environment. NorthTek is described in detail in a separate manual. NorthTek provides the user with a command line interface that allows direct interaction with the inertial system. NorthTek also allows the user to load custom programs into the inertial system that will execute a custom user application. The user may create programs that cause some of the standard protocol outputs but at user defined points, or may create custom output depending on the specific need. The user specific algorithms may be used to filter the output data, control the reporting rate, create unusual mounting configurations, or select multiple calibration sets, for example.

Because NorthTek is an environment unto itself, an entire manual is dedicated to the NorthTek System as listed in the references. The user should refer to that manual for detailed descriptions of the commands being used in this manual. You can see each of the appropriate delays in the NorthTek scripts provided in this document regarding the in-field calibration procedure.

NorthTek provides a command line access to the internal database variables. NorthTek also provides raw and processed sensor data streamed at the acquisition rate. The sections that follow illustrate some of the NorthTek functionality that can be used on the inertial system. The reader should refer to the *NorthTek System Programming Manual* for detailed descriptions on syntax and semantics of the commands used in the examples that follow.

Due to the critical nature of the in-field calibration procedure the recommended delays between commands are 250-300ms.

5.1.2.1.1 2D Compass Calibration

When using scripts this large you must set up the system as follows:

- **Insert a delay per line of 5ms.**
- **Ensure each line "to be processed" (not comment lines) is less than 60 characters (bytes).**

This script performs 2D compass calibration. To perform calibration use a terminal emulator to send this to the Inertial system (if huh? is seen flying by, add some delay to each line transmitted or reduce the baud to 38400). Once the script has been loaded, send the command "cal2D". The script will then prompt the user to capture calibration points. The user should move the compass around and select between 4 and 12 points, then hit ESC. The script will then print the magnetic error at a 0.5 Hz rate. Observe the magnetic error until it converges sufficiently then hit the spacebar or any other key. The compass will now be field calibrated. The cal3D command may be re-entered as many times as desired without re-loading the macro. The macro needs to be loaded only once while the inertial system remains powered up. There is enough information in the basic programming section and the appendix to analyze this macro sufficiently should the user desire to extend this functionality to other custom user scripts.

```
// ***********************************************************
// NorthTek Script for 2D calibration.
// ***********************************************************
```

```
// This erases this script should it be reloaded.

forget cal2D

// ************************************************************
// This is the actual calibration program.
// ************************************************************
( -- )
: cal2D

  // ************************************************************
  // Init the calibration process by setting calmode to 2.
  // ************************************************************

  calmode 2 set
  250 delay // give compass time to process

  // ************************************************************
  // Give the user a heads up that we are starting.
  // ************************************************************

  ." Calibration starting" cr

  // ************************************************************
  // Tell the calibration logic to start calibration
  // ************************************************************

  calCommand cal_start set        // cal start
  250 delay // give compass time to process

  // ************************************************************
  // Give the user some instructions
  // ************************************************************

  ." Press any key to take next point, ESC to finish" cr

  // ************************************************************
  // The code now grabs a point, the user changes the
  // compass position and repeats the cal3DState
  // from 4-12 times total.
  // sit in a loop, taking points until
  // the user hits escape.
  // User hits spacebar to take a point, ESC to quit.
  // ************************************************************

  begin
     key 27 = 0=                         // until user enters ESC
  while
     // capture a point
     calCommand cal_capture set          // take another point
     250 delay                           // give compass time to capture
     // Print out the current point number
     calNumPoints di.
  repeat

  // ************************************************************
  // Now the points are captured,
  // Issue the command to start computing
  // the real time cal values
  //
  // ************************************************************

  calCommand cal_end_capture set        // cal computation
```

```
  250 delay // give compass time to process

  // ************************************************************
  // Some more user instructions
  // ************************************************************

  ." Starting error settling" cr
  ." Press any key to terminate" cr

  // ************************************************************
  // The user observes magErr to watch it settle
  // at a minimum value (NDS-1 can display every sec or so):
  // This runs until the user has decided that the value
  // has converged. See Software User Interface Manual
  // regarding the calibration process.
  // ************************************************************

  begin                   // keep printing magErr at .250 sec intervals
    ?key 0=               // till user hits a keystroke
  while
    magErr di.
    250 delay
  repeat
  key drop               // read and remove the key used
                         // to stop the loop

  // ************************************************************
  // Let the user know that we are all done.
  // ************************************************************

  ." Calibration done!" cr

  // ************************************************************
  // Send the cal_end command.
  // ************************************************************

  calCommand cal_end set        // cal computation
  250 delay // give compass time to process

  // ************************************************************
  // End calibration mode
  // ************************************************************

  calmode 0 set          // terminate calibration mode.
  250 delay // give compass time to process

  magFieldCalErr di.
;
```

5.1.2.1.2 3D Compass Calibration

When using scripts this large you must set up the system as follows:

- **Insert a delay per line of 5ms.**
- **Ensure each line "to be processed" (not comment lines) is less than 60 characters (bytes).**

This script performs 3D compass calibration. To perform calibration use a terminal emulator to send this to the Inertial system (if huh? is seen flying by, add some delay to each line transmitted or reduce the baud to 38400). Once the script has been loaded, send the command "cal3D". The script will then prompt the user to capture calibration points. The user should move the compass around and select between 4 and 12 points, then hit ESC. The script will then print the magnetic error at a 0.5 Hz rate. Observe the magnetic error until it converges sufficiently then hit the spacebar or any other key. The compass will now be field calibrated. The cal3D command may be re-entered as many times as desired without re-loading the macro. The macro needs to be loaded only once while the inertial system remains powered up. There is enough information in the basic programming section and the appendix to analyze this macro sufficiently should the user desire to extend this functionality to other custom user scripts.

```
// ************************************************************
// NorthTek Script for 3d calibration.
// ************************************************************
// This erases this script should it be reloaded.

forget cal3D


// ************************************************************
// This is the actual calibration program.
// ************************************************************
( -- )
: cal3D

  // ************************************************************
  // Init the calibration process by setting calmode to 1.
  // ************************************************************

  calmode 1 set
  250 delay // give compass time to process

  // ************************************************************
  // Give the user a heads up that we are starting.
  // ************************************************************

  ." Calibration starting" cr

  // ************************************************************
  // Tell the calibration logic to start calibration
  // ************************************************************

  calCommand cal_start set        // cal start
  250 delay // give compass time to process

  // ************************************************************
  // Give the user some instructions
  // ************************************************************

  ." Press any key to take next point, ESC to finish" cr

  // ************************************************************
  // The code now grabs a point, the user changes the
  // compass position and repeats the cal3DState
  // from 4-12 times total.
  // sit in a loop, taking points until
  // the user hits escape.
  // User hits spacebar to take a point, ESC to quit.
  // ************************************************************

  begin
     key 27 = 0=                         // until user enters ESC
```

```
while
   // capture a point
   calCommand cal_capture set          // take another point
   250 delay                           // give compass time to capture
   // Print out the current point number
   calNumPoints di.
repeat


// ************************************************************
// Now the points are captured,
// Issue the command to start computing
// the real time cal values
//
// ************************************************************

calCommand cal_end_capture set        // cal computation
250 delay // give compass time to process

// ************************************************************
// Some more user instructions
// ************************************************************

." Starting error settling" cr
." Press any key to terminate" cr

// ************************************************************
// The user observes magErr to watch it settle
// at a minimum value (NDS-1 can display every sec or so):
// This runs until the user has decided that the value
// has converged. See Software Interface User's Manual
// regarding the calibration process.
// ************************************************************

begin                   // keep printing magErr at .250 sec intervals
  ?key 0=               // till user hits a keystroke
while
  magErr di.
  250 delay
repeat
key drop               // read and remove the key used
                       // to stop the loop

// ************************************************************
// Let the user know that we are all done.
// ************************************************************

." Calibration done!" cr

// ************************************************************
// Send the cal_end command.
// ************************************************************

calCommand cal_end set        // cal computation
250 delay // give compass time to process

// ************************************************************
// End calibration mode
// ************************************************************

calmode 0 set          // terminate calibration mode.
250 delay // give compass time to process
```

```
  magFieldCalErr di.
  250 delay
;
```

5.1.2.2 NMEA

NMEA commands use ASCII text strings to communicate with the digital compass through the User Port. All commands should follow the command syntax exactly. No white space should be included in NMEA command strings. Data is formatted with the first character being "***$***" to signify the start of a NMEA command and include a **"*"** to signify the start of the checksum.

It is not necessary to include any checksum characters in commands sent to the digital compass. Strictly speaking, this is not standard NMEA protocol but manually entering checksums is not practical. However an option exists to enable NMEA checksums if additional error protection is desired. This option is off by default and may be enabled via a NorthTek or a NMEA command string (see the ***nmeaignorechecksum*** explanation in the Protocols section). All NMEA responses from the compass will contain a checksum followed by a carriage return and line feed.

The Inertial system by default at power up does not echo the NMEA character strings, however a NorthTek command exists that will enable echo of NMEA command strings as they are typed (***nmeaecho 1 set drop<cr>***). This mode is provided for evaluators that wish to use a serial terminal to hand type inertial system commands. Each command sent must be terminated with a carriage return ***<CR>*** and line feed ***<LF>*** (entered as ***<CTRL-J>*** in some terminal programs such as Tera Term). To turn off the character echo, use the command: ***nmeaecho 0 set drop<cr>***

NMEA commands must be completed in a timely manner. The terminating character must be received within 5 seconds after the reception of the last character. If the command is not received in the time allotted all characters received will be discarded and the command ignored. This is to prevent the Inertial system from getting stuck in the NMEA protocol.

Due to the critical nature of the in-field calibration procedure the recommended delays between commands are 250-300ms.

5.1.2.2.1 2-D Field Calibration

The 2-D calibration procedure is used for applications that stay within a plane. This means that the X/Y plane should be as near to flat and level with respect to the Earth's gravitational vector as possible. Any slight errors will result in errors in the magnetic calibration. The positions should be chosen to represent the extremes expected for the application.

The results are stored in magFieldCalX, magFieldCalY and magFieldCalZ with the first value being offset and the second being a divisor.

Send:

```
$PSPA,CAL=2D<cr><lf>
// delay 300ms
$PSPA,CAL_CMD=START_CAL<cr><lf>
// delay 300ms
```

Change position and repeat CAPTURE_POINT totaling 4 to 12 points

```
$PSPA,CAL_CMD=CAPTURE<cr><lf>
// delay 300ms
.
.
.
$PSPA,CAL_CMD=CAPTURE<cr><lf>
```

Stop the capture after a minimum of 4 points.

```
// delay 300ms
$PSPA,CAL_CMD=END_CAPTURE<cr><lf>
$PSPA,MAGERR,RPT=0.5<cr><lf>
```

Watch for the magnetic error quality indicator to settle to a minimum value

```
$PSPA,CAL_CMD=END_CAL<cr><lf>
// delay 300ms
$PSPA,CAL=OFF<cr><lf>
```

Response:

```
$PSPA,CAL=2D*3B
$PSPA,CAL_CMD=START_CAL*09
$PSPA,CAL_CMD=CAPTURE,POINTS=<int#>*<checksum in hex>
.
.
.
$PSPA,CAL_CMD=CAPTURE,POINTS=<int#>*<checksum in hex>
$PSPA,CAL_CMD=END_CAPTURE*0C
$PSPA,MagErr=<float#>*<checksum in hex>
$PSPA,MagErr=<float#>*<checksum in hex>
.
.
.
$PSPA,MagErr=<float#>*<checksum in hex>
$PSPA,MagErr=<float#>*<checksum in hex>
$PSPA,CAL_CMD=END_CAL*06
$PSPA,CAL=OFF*02
```

5.1.2.2.2 3-D Field Calibration

The 3-D calibration procedure is used for applications that move outside of a plane. It does **not** require full 360° pitch and roll capability for this calibration. The positions should be chosen to represent the extremes expected for the application.

The results are stored in magFieldCalX, magFieldCalY and magFieldCalZ with the first value being offset and the second being a divisor.

Send:

```
$PSPA,CAL=3D<cr><lf>
// delay 300ms
$PSPA,CAL_CMD=START_CAL<cr><lf>
Change position and repeat CAPTURE_POINT totaling 4 to 12 points
// delay 300ms
$PSPA,CAL_CMD=CAPTURE<cr><lf>
// delay 300ms
.
.
.
$PSPA,CAL_CMD=CAPTURE<cr><lf>
Stop the capture after a minimum of 4 points.
// delay 300ms
$PSPA,CAL_CMD=END_CAPTURE<cr><lf>
$PSPA,MAGERR,RPT=0.5<cr><lf>
Watch for the magnetic error quality indicator to settle to a minimum value
$PSPA,CAL_CMD=END_CAL<cr><lf>
// delay 300ms
$PSPA,CAL=OFF<cr><lf>
```

Response:

```
$PSPA,CAL=3D*3A
$PSPA,CAL_CMD=START_CAL*09
$PSPA,CAL_CMD=CAPTURE,POINTS=<int#>*<checksum in hex>
.
.
.
$PSPA,CAL_CMD=CAPTURE,POINTS=<int#>*<checksum in hex>
$PSPA,CAL_CMD=END_CAPTURE*0C
$PSPA,MagErr=<float#>*<checksum in hex>
$PSPA,MagErr=<float#>*<checksum in hex>
.
.
.
$PSPA,MagErr=<float#>*<checksum in hex>
```

```
$PSPA,MagErr=<float#>*<checksum in hex>
$PSPA,CAL_CMD=END_CAL*06
$PSPA,CAL=OFF*02
```

A Examples of 2D and 3D calibration orientations

A.1 AHRS-8, GEDC-6E, DC-4E

A.1.1 2D calibration

A.1.1.1 Perfect 2D calibration

When performing a 2D calibration, the user should always do the following:

1. Use the World Magnetic Model to improve the results
2. Start by finding the maximum magnetometer X-Value in the X-Y plane and use this as the starting point
3. Take the full 12 points instead of less. To get the best representation of the plane, using more points will be the best solution.
4. Space the points as diversely as possible.
    a. For a full 360° this means take a point every 30°.

A.1.2 3D calibration

There are an infinite number of possible solutions for the 3D calibration and only a select few will be described here. These are only a few and can easily be modified to improve the diversity over the sphere.

A.1.2.1 Perfect 3D calibration

The exact perfect solution for the 3D selection of points is the Icosahedron. http://en.wikipedia.org/wiki/Icosahedron

1. Taking points at the vertices of the Icosahedron will give the perfect diversity for the sphere.

A.1.2.2 3D calibration with Inverted points.

Sparton's 3D calibration algorithm is best used when the points are as diversely spread as possible for a given environment. Here is an example with inverted points.

1. Start by finding the maximum magnetometer X-Value in the X-Y plane and use this as the starting point
2. Take first point with Pitch increased to 45°.
3. Take the next five points at 60° yaw angles from the previous point.
4. Invert unit with Roll at 180° and take pitch to -45° and take point 7.
5. Take the next five points at 60° yaw angles from the previous point maintaining the 180° roll and the pitch at -45°.

A.1.2.3 3D calibration with non-inverted points.

Sparton's 3D calibration algorithm is best used when the points are as diversely spread as possible for a given environment. Here is an example without inverted points. The order of points taken does not matter. For example you can take the three points a 0° heading and 0° and +/- 45° pitch before moving to the next heading at 90°.

1. Start by finding the maximum magnetometer X-Value in the X-Y plane and use this as the starting point
2. Take first point on a Flat and Level Plane.
3. Take the next three points at 90° yaw angles from the previous point.
4. Increase Pitch to 45° and take point 5.
5. Take the next three points at 90° yaw angles from the previous point maintaining the 45° pitch.
6. Decrease Pitch to -45° and take point 9.
7. Take the next three points at 90° yaw angles from the previous point maintaining the -45° pitch.

B How and When to use the In-Field Calibration related variables

B.1possibleMagPointQF

As each calibration point is selected, a quality factor is provided (called magPointQF) that indicates how well the magnetic values in the points collected cover the expected possible range for each axis. By default, the expected possible range is 2 times 450 mGauss (world average). If the Auto Variance is computed (World Magnetic Model), then the magnetic field strength computed for the given location will be used which should yield a more consistent indicator for different world locations. As the AHRS-8 is maneuvered, a real-time value is provided that indicates what the magPointQF would be if the current position was added to the collected points (called possibleMagPointQF). By using possibleMagPointQF during an in-field calibration, the user can find positions that increase the quality factor.

- **Usage**:
  - See the NorthTek script Cal3d_PointQualityFactor.4th script in the website downloads for an example
  - When selecting calibration points using this script, pressing the space bar toggles between viewing possibleMagPointQF and selecting a point.

B.2magFieldCalErr

The magnetic data collected during field calibration will be analyzed for variability in field strength. This variability can be related directly to heading error providing instant feedback as to the quality of calibration (called magFieldCalErr). This variable gives an estimate of the error in degrees and is computed once at the end of the in-field calibration process.

- **Usage**:
  - Execute the 3D or 2D in-field calibration to completion
  - Read the variable magFieldCalErr (for example: magFieldCalErr di.)
  - If the value is too high, then perform the in-field calibration again with better point selection

B.3magErr

- **Usage**:
  - This variable in only used once the convergence process has begun after the point selection has ended.
  - Monitor this variable to see if the calibration algorithm has converged.

C Calibration algorithms

Introduction

This code is meant only as a primer and not to be used exactly as is. Must be modified to be used inside a microcontroller or on a host PC.

C.1 Algorithm in NMEA messaging format

$PSPA,CAL=3D
*$PSPA,CAL=3D*3A*

// delay here ~300mS

$PSPA,CAL_CMD=START_CAL
*$PSPA,CAL_CMD=START_CAL*09*

// delay here ~300mS

$PSPA,CAL_CMD=CAPTURE
*$PSPA,CAL_CMD=CAPTURE,POINTS=1*23*

*// delay here for 10ms*calPointsNumToAvg + 50ms*

$PSPA,CAL_CMD=CAPTURE
*$PSPA,CAL_CMD=CAPTURE,POINTS=2*20*

*// delay here for 10ms*calPointsNumToAvg + 50ms*

$PSPA,CAL_CMD=CAPTURE
*$PSPA,CAL_CMD=CAPTURE,POINTS=3*21*

*// delay here for 10ms*calPointsNumToAvg + 50ms*

$PSPA,CAL_CMD=CAPTURE
*$PSPA,CAL_CMD=CAPTURE,POINTS=4*26*

*// delay here for 10ms*calPointsNumToAvg + 50ms*

$PSPA,CAL_CMD=CAPTURE
*$PSPA,CAL_CMD=CAPTURE,POINTS=5*27*

*// delay here for 10ms*calPointsNumToAvg + 50ms*

$PSPA,CAL_CMD=CAPTURE
*$PSPA,CAL_CMD=CAPTURE,POINTS=6*24*

*// delay here for 10ms*calPointsNumToAvg + 50ms*

$PSPA,CAL_CMD=CAPTURE
*$PSPA,CAL_CMD=CAPTURE,POINTS=7*25*

*// delay here for 10ms*calPointsNumToAvg + 50ms*

$PSPA,CAL_CMD=CAPTURE
*$PSPA,CAL_CMD=CAPTURE,POINTS=8*2A*

*// delay here for 10ms*calPointsNumToAvg + 50ms*

$PSPA,CAL_CMD=CAPTURE
*$PSPA,CAL_CMD=CAPTURE,POINTS=9*2B*

*// delay here for 10ms*calPointsNumToAvg + 50ms*

$PSPA,CAL_CMD=CAPTURE
*$PSPA,CAL_CMD=CAPTURE,POINTS=10*13*

*// delay here for 10ms*calPointsNumToAvg + 50ms*

$PSPA,CAL_CMD=CAPTURE
*$PSPA,CAL_CMD=CAPTURE,POINTS=11*12*

*// delay here for 10ms*calPointsNumToAvg + 50ms*

$PSPA,CAL_CMD=CAPTURE
*$PSPA,CAL_CMD=CAPTURE,POINTS=12*11*

*// delay here for 10ms*calPointsNumToAvg + 50ms*

$PSPA,CAL_CMD=END_CAPTURE
*$PSPA,CAL_CMD=END_CAPTURE*0C*

// delay here ~100mS

$PSPA,MAGERR,RPT=0.5
*$PSPA,MagErr=0.148678*17*
*$PSPA,MagErr=0.148649*15*
*$PSPA,MagErr=0.148693*12*
*$PSPA,MagErr=0.148678*17*
*$PSPA,MagErr=0.148650*1D*
*$PSPA,MagErr=0.148677*18*
*$PSPA,MagErr=0.148665*1B*
*$PSPA,MagErr=0.148628*12*
*$PSPA,MagErr=0.148649*15*
*$PSPA,MagErr=0.148637*1C*
*$PSPA,MagErr=0.148640*1C*
*$PSPA,MagErr=0.148660*1E*
*$PSPA,MagErr=0.148672*1D*
*$PSPA,MagErr=0.148648*14*
*$PSPA,MagErr=0.148656*1B*
*$PSPA,MagErr=0.148634*1F*
*$PSPA,MagErr=0.148660*1E*
*$PSPA,MagErr=0.148653*1E*
*$PSPA,MagErr=0.148655*18*
*$PSPA,MagErr=0.148668*16*

// delay here ~300mS

// You don't have to send the $PSPA,MAGERR,RPT=0.5 command
// but you need to wait to send the next command until the calibration converges

$PSPA,CAL_CMD=END_CAL
*$PSPA,CAL_CMD=END_CAL*06*

// delay here ~300mS

$PSPA,CAL=OFF
*$PSPA,CAL=OFF*02*

// delay here ~300mS

$PSRFS,magFieldCalErr,get
*$PSRFS,magFieldCalErr,0.120008*73*

C.2 Algorithm in NorthTek

calmode 1 set dropOK
300 delayOK

calCommand cal_start set dropOK
300 delayOK

calCommand cal_capture set dropOK
300 delayOK

calNumPoints di.
calNumPoints = 1

calCommand cal_capture set dropOK
300 delayOK

calNumPoints di.
calNumPoints = 2

calCommand cal_capture set dropOK
300 delayOK
calNumPoints di.
calNumPoints = 3

calCommand cal_capture set dropOK
300 delayOK

calNumPoints di.
calNumPoints = 4

calCommand cal_capture set dropOK
300 delayOK

calNumPoints di.
calNumPoints = 5

calCommand cal_capture set dropOK
300 delayOK

calNumPoints di.
calNumPoints = 6

calCommand cal_end_capture set dropOK
300 delayOK

calCommand cal_end set dropOK
300 delayOK

calmode 0 set dropOK
300 delayOK

magFieldCalErr di.
magFieldCalErr = 1.648529e-02