# DIY Headtracker

Design and software by Anurag Pandey ([helpmeinproject@gmail.com](mailto:helpmeinproject@gmail.com))

Main RCGroups discussion (recommended reading): <http://www.rcgroups.com/forums/showthread.php?t=1677559>

Google Code repository: <http://code.google.com/p/open-headtracker/>

**Required hardware**

Arduino Nano

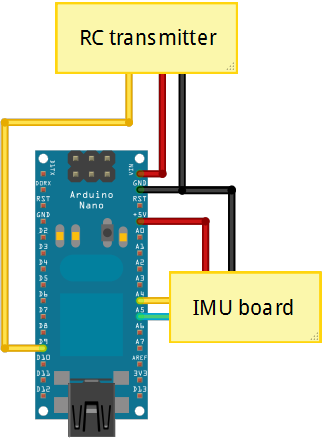
9-axis IMU sensor board 

**Sources for hardware:**

Arduino Nano:  
[http://www.dealextreme.com/p/nano-v3...-118037?item=8](http://www.dealextreme.com/p/nano-v3-0-avr-atmega328-p-20au-module-board-usb-cable-for-arduino-118037?item=8)  
[http://www.hobbyking.com/hobbyking/s...ler\_Board.html](http://www.hobbyking.com/hobbyking/store/__25086__Arduino_Nano_V3_0_Microcontroller_Board.html)  
[http://www.dealextreme.com/p/arduino...0-81877?item=2](http://www.dealextreme.com/p/arduino-nano-v3-0-81877?item=2)  
  
Sensor board (9-axis IMU):  
[http://www.dealextreme.com/p/gy-85-6...arduino-148436](http://www.dealextreme.com/p/gy-85-6dof-9dof-imu-sensor-module-for-arduino-148436)  
[http://www.ebay.com/itm/ws/eBayISAPI...E:L:OC:US:3160](http://www.ebay.com/itm/ws/eBayISAPI.dll?ViewItem&item=270962872404&ssPageName=ADME:L:OC:US:3160)  
[http://www.goodluckbuy.com/gy-85-sen...u-sensor-.html](http://www.goodluckbuy.com/gy-85-sensor-modules-6dof-9dof-imu-sensor-.html)  
[http://www.goodluckbuy.com/nine-axis...3l-module.html](http://www.goodluckbuy.com/nine-axis-dgree-of-freedom-imu-sensor-itg3200-itg3205-adxl345-hmc5883l-module.html)  
  
This project is made as user-friendly as possible while maintaining flexibility for developers and tinkerers. Only a little technical knowledge should be necessary. All necessary settings can be changed from a graphical user interface, making it easy for everyone to use.

A quick test-video can be found here. It's hard to show if a headtracker works well or not, so it's just a quick test-video.<https://dl.dropbox.com/u/3947315/Videogdp.3gp>

Connection pinout



**Figure: Connection diagram (centering button not shown between D11 and GND)**

Assembly and flashing quick guide   
  
You should have:  
1 x Arduino Nano or similar  
1 x Sensor board

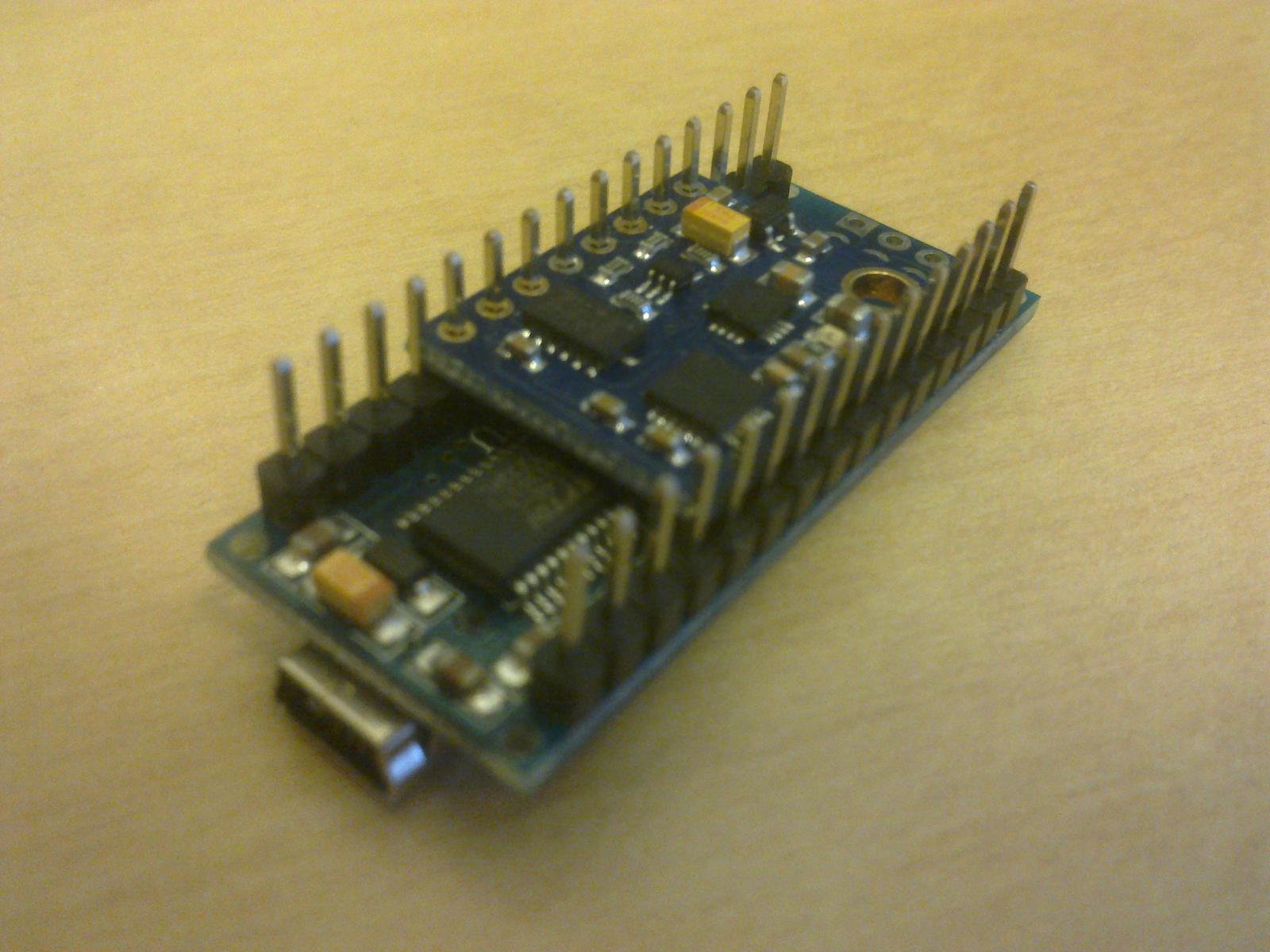
First, let’s assemble the hardware. It's pretty straight forward. Connect the pins as specified in the following table.

|  |  |
| --- | --- |
| **Sensorboard Pin** | **Arduino Pin** |
| Vcc\_in | 5 volt |
| GND | Ground/GND |
| SCL | A5 |
| SDA | A4 |

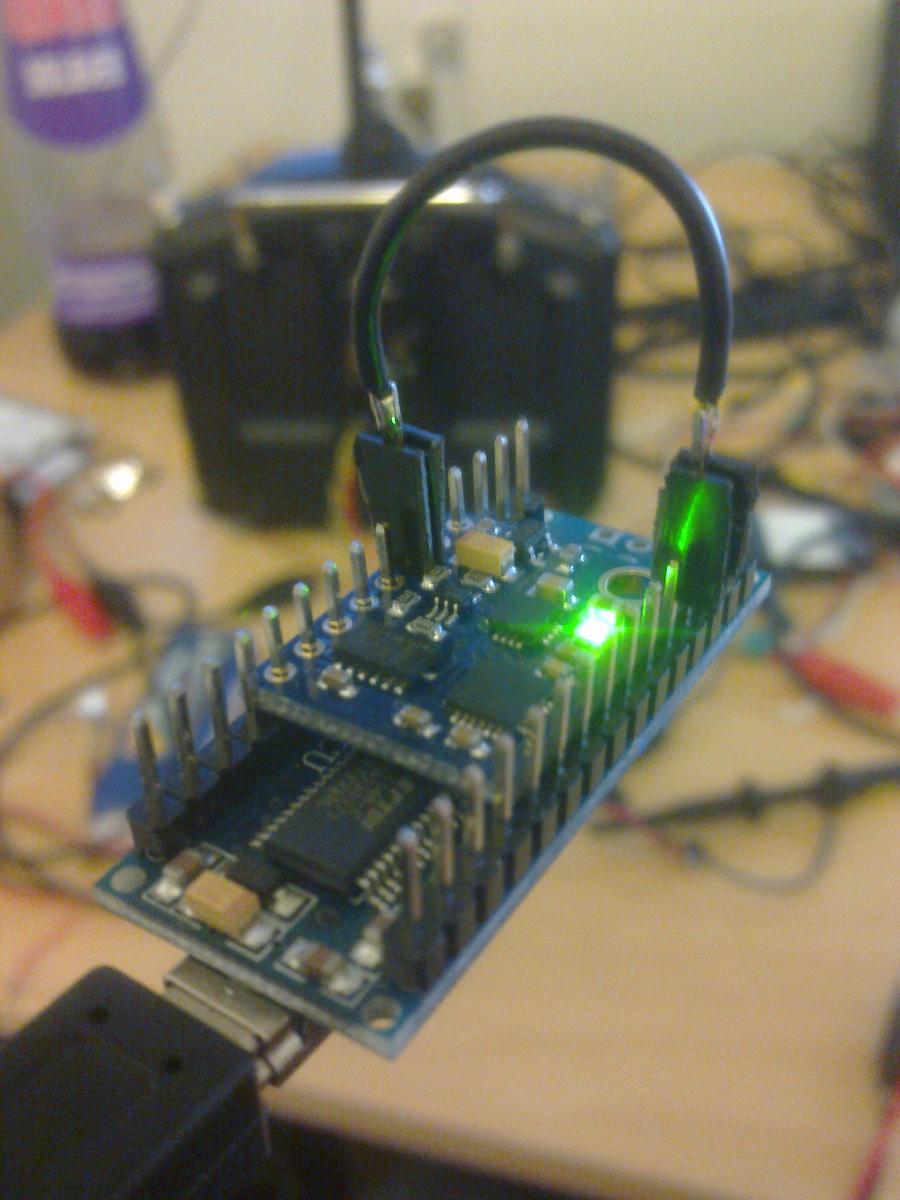
|  |  |
| --- | --- |
| **Transmitter** | **Arduino Pin** |
| PPM\_IN | D9 |
| V\_out | V\_in |
| Ground | Ground/GND |

Optionally, a reset button can be placed between Arduino pin D11 and ground. You can use any momentary pushbutton or toggle switch for this. This button can then be used to re-center the tracker if you change orientation while you’re flying. It can also be used to pause/unpause head tracking by holding the button for 1.5 seconds, if your setup requires this.  
  
The sensor board pins match pretty well with the Arduino pins, so you can just connect it directly. The only pin that doesn't match is ground. I recommend soldering a short wire between one of the Arduino ground pins and the Arduino pin A6(which is connected to the sensor board’s ground pin). The Arduino pin will be set as an input, so connecting it to ground will not cause any issues with the Arduino’s microprocessor.  
  
The recommended configuration for the sensor board is to place it underneath the Arduino board, like in the following picture. It is, however, possible to remotely mount the sensor board away from the Arduino module, or place it in other configurations. Just make sure the pins match as specified. Please remember that the magnetometer is pretty sensitive. Even small currents nearby will cause a magnetic field. If it's a constant field, chances are you can calibrate it, but please be aware of this problem.

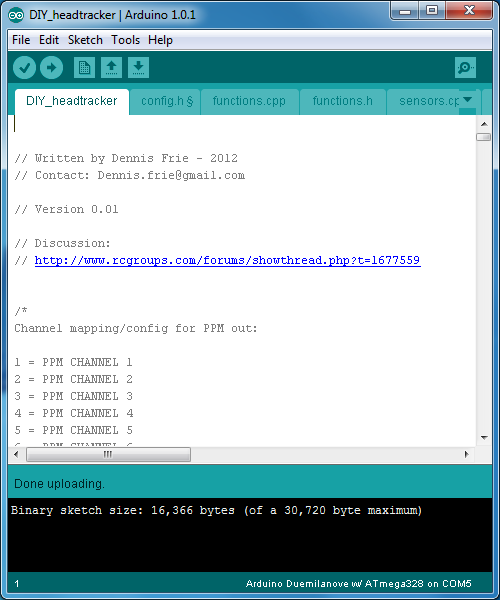
You can test your board without soldering the sensor board and using a temporary jumper between GND and A6.Once your tests are complete and you have the tracker working, solder the sensor board to the Arduino, and solder in a permanent jumper between the Arduino’s GND pin and A6. You can also (optionally) cut or desolder unused pins from the Arduino, including the 6-pin ISP header.



**Figure: Recommended hardware assembly configuration. Jumper between A6 and GND (underneath the Arduino) cannot be seen in this image.**

  
**Figure: For testing, a simple jumper can be used. Not recommended for final installation.**

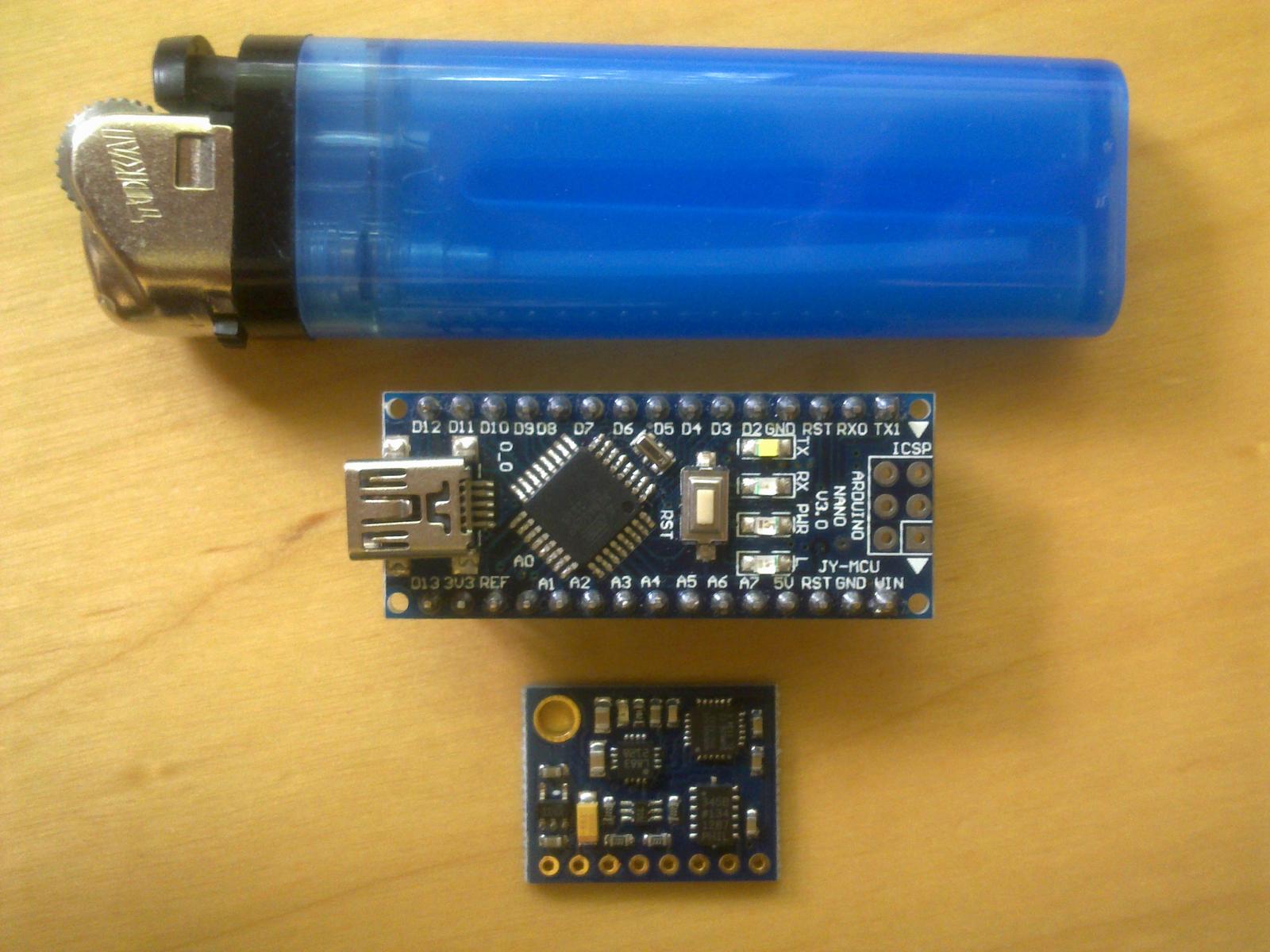
**Getting the Arduino software set up:**

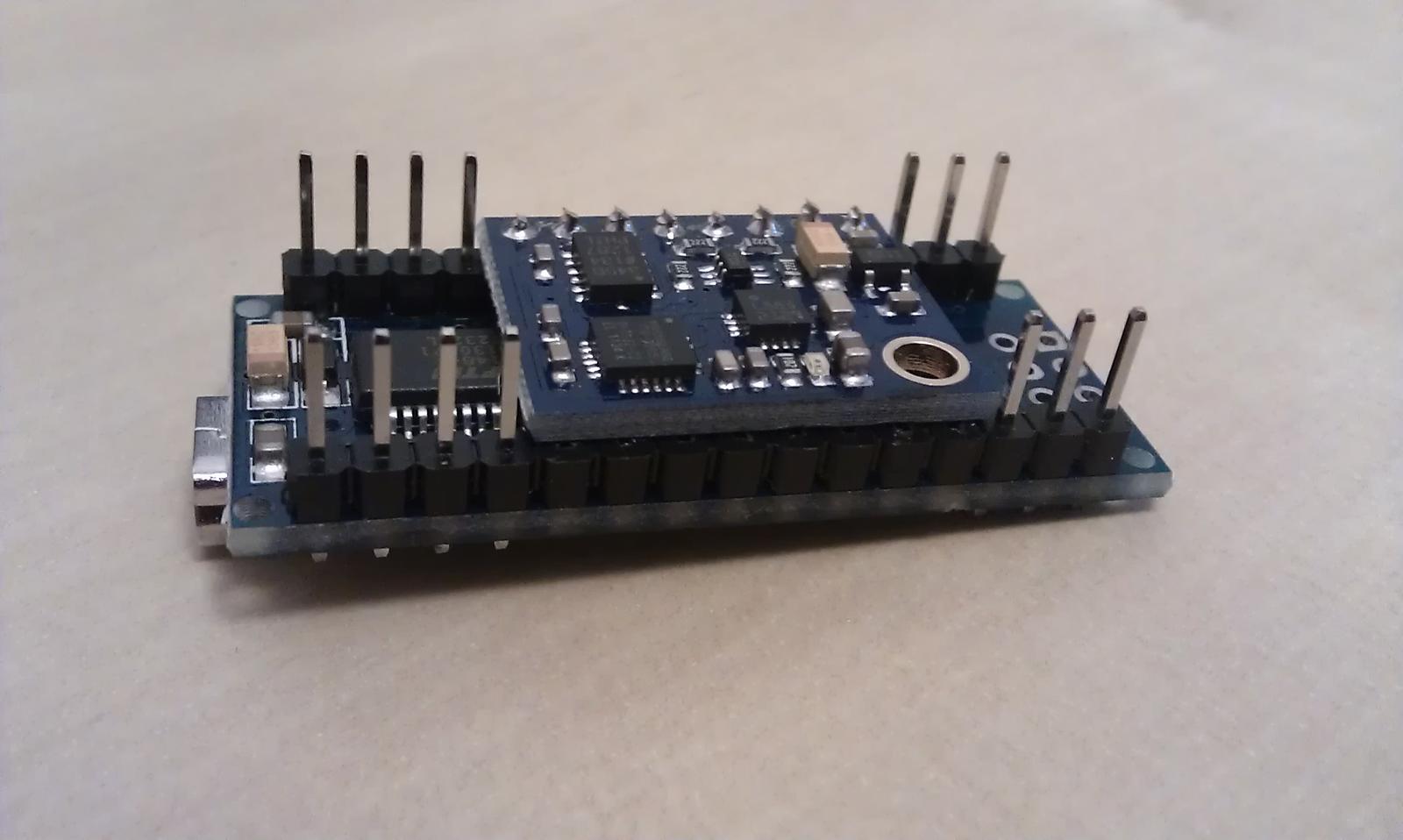
1. Download Arduino IDE from <http://arduino.cc/hu/Main/Software> (should be version 1 or higher)
2. Connect your Arduino board to your PC through the USB connector
3. If the driver is installed automatically, just continue. If not, select the folder called "Driver" inside the Arduino IDE folder
4. Download the latest headtracker-software from <http://code.google.com/p/open-headtracker/>
5. Open Arduino IDE (the program), Select file -> Open, find the headtracker software downloaded and open the file DIY\_headtracker.ino
6. You should now have a window looking pretty much like this:  
   
7. Go to tools -> Board and select "Arduino Nano w/Atmega328
8. Go to tools -> Serial port -> Select the com-port used by Arduino (you should remember the port-number for later use in the GUI).
9. Press the left-arrow in Arduino IDE to upload, or select file -> Upload, or press Ctrl + u.
10. Hopefully it starts to upload the firmware. It should flash the LED's quickly for   
    a while, and stop when the upload/programming is complete.   
      
    **Calibration**

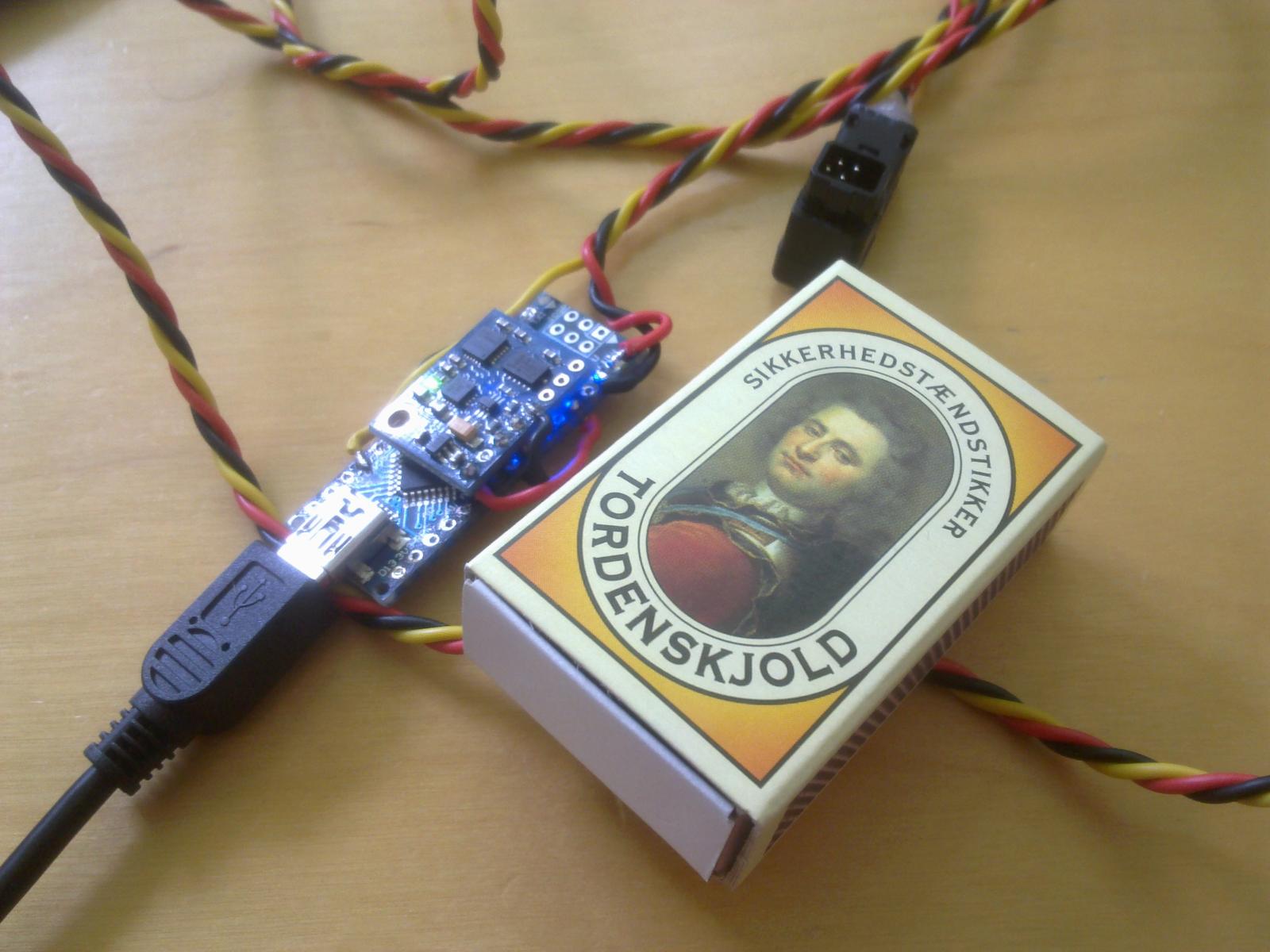
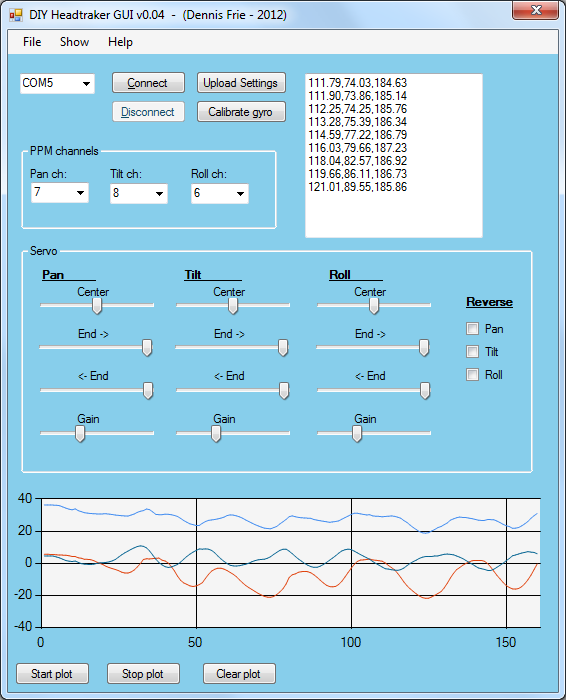
Before your headtracker can be used for accurate head tracking, you must first calibrate it. Calibration is done through thecalibration button (or menu option) in the GUI. After you connect the GUI to your headtracker, press the “Calibrate” button and follow the on-screen prompts and pictures.

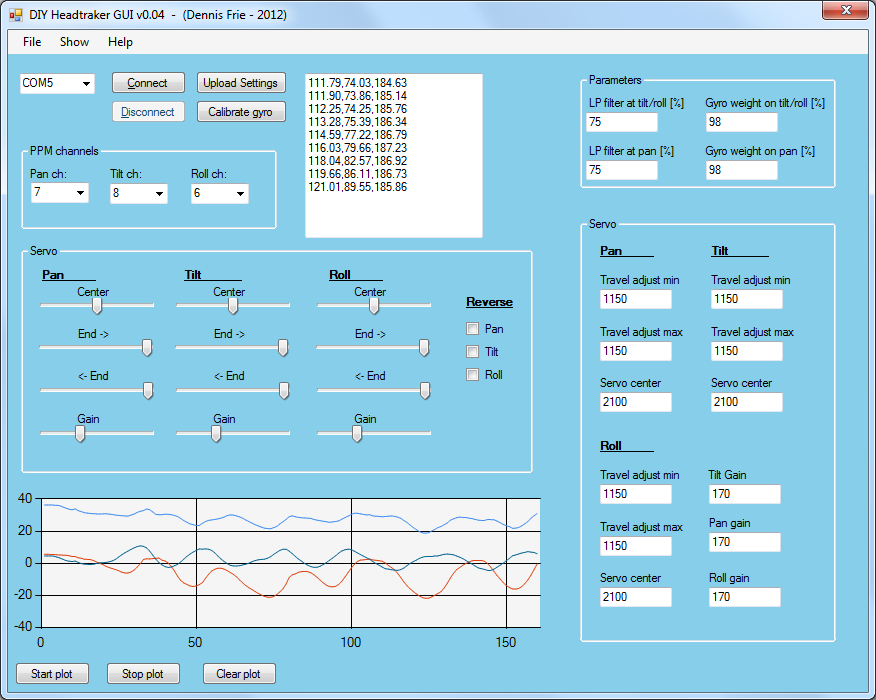
Hopefully that's it, and you are ready to go.   
  
Some known limitations:

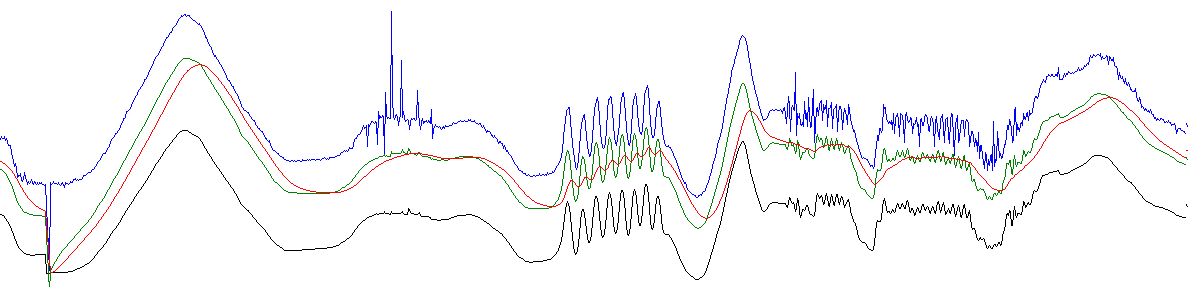
* The sensorboard must be orientated with the components up.
* Nothing in the GUI is "idiot proof". It will accept more or less everything and just do it.

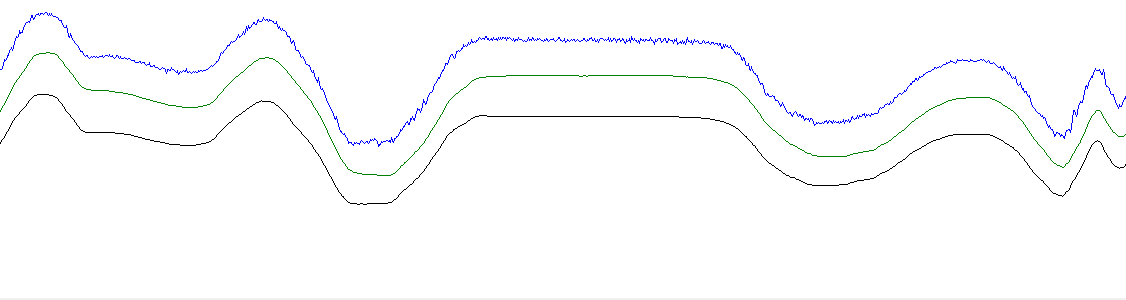
Here’s a shot of the two pieces of hardware as they appear out of the package:  
  
  
Assembled hardware - just missing connections to RC-transmitter. It’s not visible in this picture, but there’s a wire jumper on the bottom of the Arduino that connects the Arduino ground to the sensor board’s ground.

  
  
Early test-version:

  
  
Graphical user interface  
  
  
- GUI showing advanced settings too:

  
  
  
  
Headtracker sensor plots, raw, filtered etc:



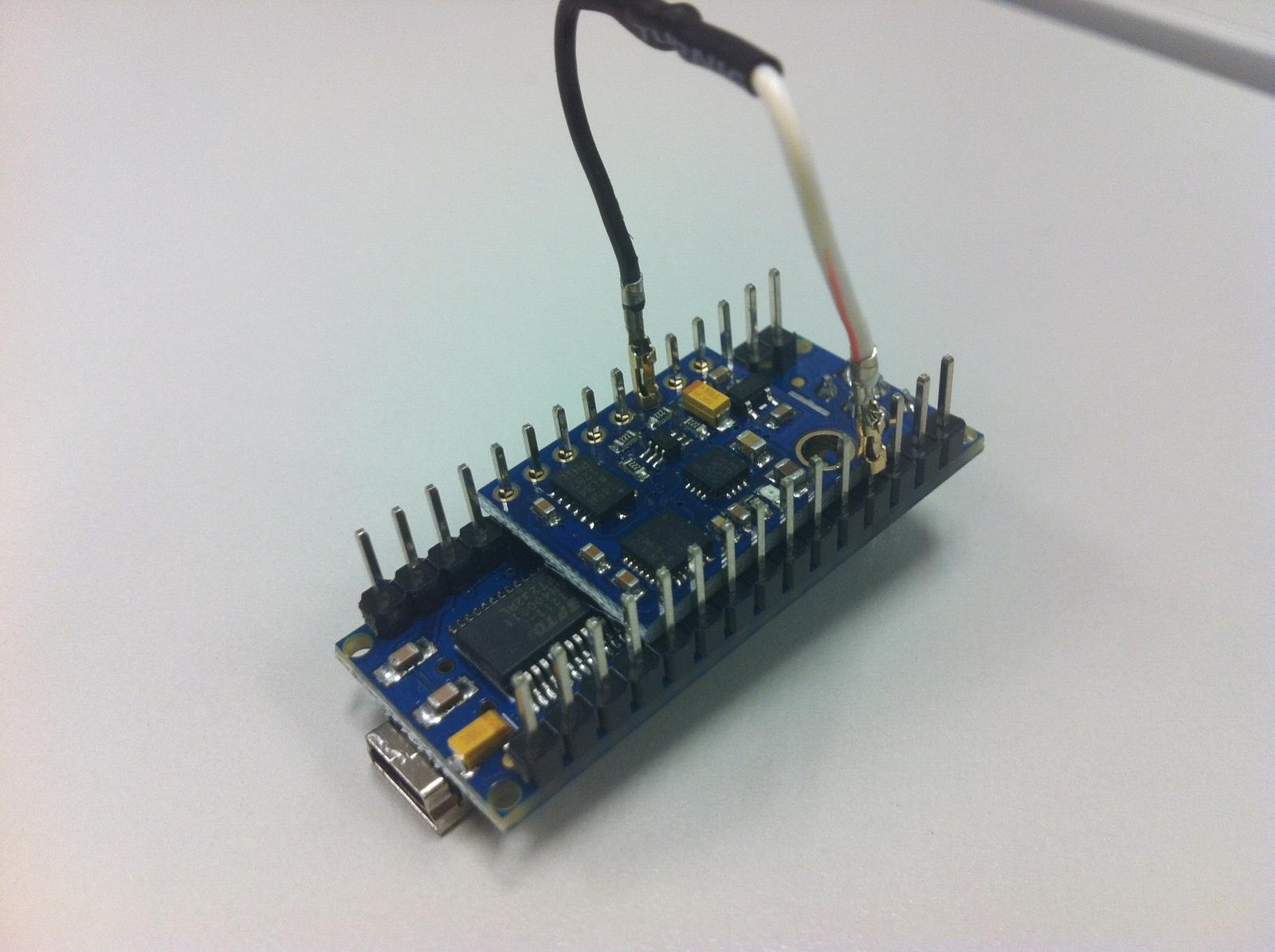
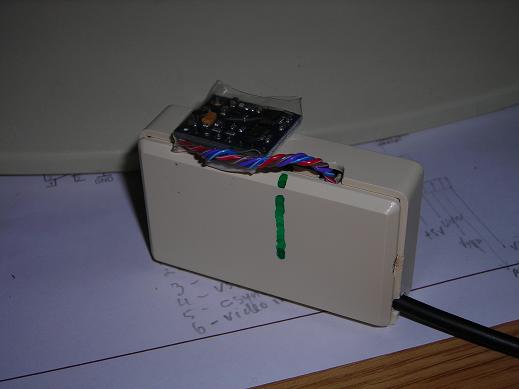
  
  
  
  
Using the GUI  
  
  
**PPM channels**  
Used to set the PPM-channel pan, tilt, and roll are assigned to.  
  
**Center**  
The center position of the servo for each axis.  
  
**End**  
Servo travel endpoints (maximum and minimum)  
  
**Gain**  
The gain determines how much the servo should move for a given amount of tracker movement. Default is 170 which pretty much gives 1:1 (90 degrees head-movement = 90 degrees servo movement).   
  
  
**All the "advanced settings"**  
- - - -   
**LP filter at tilt/roll [%]**  
Lowpass-filter of the final tilt/roll output.  
1 = max lowpass/time constant. Will give very smooth, but also very slow change.   
100 = lowpass off.   
  
**LP filter at pan [%]**  
Lowpass-filter of the final pan output.  
1 = max lowpass/time constant. Will give very smooth, but also very slow change.   
100 = lowpass off.   
  
**Gyro weight on tilt/roll [%]**  
How much do we thrust the gyro compared to the accelerometer on the tilt/roll axis? Accelerometer is noisy and should only be used to slowly compensate for drift.   
100 = only use gyro.  
0 = only use accelerometer  
  
**Gyro weight on pan [%]**  
How much do we thrust the gyro compared to the magnetometer on the pan-axis? The gyro is a lot more accurate, so magnetometer should only be used to slowly compensate drift.   
100 = only use gyro  
0 = only use magnetometer  
  
**Servo pulse variation**  
In short: Servo travel adjustment.  
  
Used to set the max variation of the pulse-width in the PPM signal. The unit is microseconds\*2 - matching the Atmega timer   
  
**Servo center**  
In short: Servo neutral position  
  
Used to set the center/default length of the pulse-width in the PPM signal. The unit is microseconds\*2 - matching the Atmega timer   
  
**Tilt and roll gain**  
Gain of the servo. You decide if you want to turn the head 10 degrees or 360 degrees to get full travel.   
  
**Pan gain**  
Gain of the servo. You decide if you want to turn the head 10 degrees or 360 degrees to get full travel.   
  
**Reverse (Tilt, roll and Pan)**  
used to inverse the servo direction.   
  
  
General questions  
  
**Can I use other Arduino versions?**  
Sure, all Arduino boards should have the necessary pins available (as far as I know). I have choosen Arduino Nano as it's cheap, small, have FTDI onboard and the regulator can handle a bit higher current, compared to Arduino Pro etc. But pick whatever you prefer/have available.   
  
**Can I use other sensors?**  
It should be pretty easy to use other sensors, as long as it's I2C. Update the sensor-init, sensor-reading and the sensor-configuration and you should more or less be good to go. I'll maybe add support for another IMU later.   
  
**Can I buy a headtracker from you?**  
No, it's meant as a DIY project. It's easy to make, and it should be pretty straight forward. I like to develop, but care little about sale etc. http://static.rcgroups.com/forums/images/smilies/wink.gif  
  
**Can I sell this?**  
Feel free. I can imagine a few people would like to buy a nice ready-made unit, but it would be great to know if you use my software/work. But in general it's uploaded with no strings attached.   
  
**What about copyright/license etc?**  
It's uploaded as OpenSource, and I expect it to be used and abused - don't really see much sense in "protecting" my work. So feel free to use it for whatever purpose.

**A few pictures from other peoples build**, for reference

SpookiePower



  
  
SubSonic:

  
  
  
Jalves:  
  
  
BlueAngel

