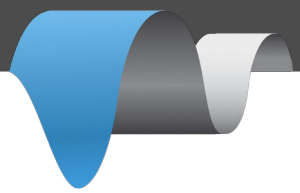
SimpleBGC 32bit 3-Axis Software User Manual

Board v. 3.x
Firmware v. 2.44
GUI v. 2.44

CONTENTS



1. Overview

This manual provides directions on how to connect, adjust and calibrate the SimpleBGC 32bit 3-Axis controller board by Basecam Electronics. To begin using the board the following are the components that are necessary to assemble. The controller board and additionally either one or two IMU units. A USB connection to the board or an optional Bluetooth converter (a standard TTL interface Bluetooth module- occasionally already integrated into the board but in any event readily available in the market). A computer to make and write settings to the controller via Basecam's software. And the Basecam software which runs on Windows, MacOS and Linux. The software is dowloaded from the Basecam website. Note that the gui software version should match (or be greater than) the firmware version deployed on the board.

Also necessary is a suitable physical apparatus to mount and orient the sensors for use during calibration- not a gimbal itself but rather a small cube of any material, cardboard foam, wood etc. that has true square sides which can be turned from side to side during calibration (further described later). Also needed ultimately is a gimbal with motors, that is well balanced in each dimension about its center point. The objective for gimbal design is that the center of effort be a fixed unmoving point- irrespective of the position of the gimbals arms and that the camera (the stabilized device) be centered- its mass centered- at that point. Additional optional components such as switches, joystick operation and interfaces to remote control devices (PWM or S.Bus from a standard RC receiver) are described in detail farther on.

SimpleBGC actively compensates for undesirable movement in the stabilized portion of the gimbal (which mounts a camera or other device) that requires precise positioning irrespective of movement in the surrounding frame of reference. The controllers high performance motion sensors (MEMS gyros) and ARM Cortex$_{(TM)}$ 32 bit core and additional capabilities to integrate PWM control (and other) signals directly to the stabilized devise makes the controller an ideal platform for applications from stabilized hand carried camera mounts to more complex objectives such as track and boom carried or aerial mounted applications.

Stabilizing is accomplished by directing energy to the gimbal motors in response to reception of repositioning data from the gyroscopic sensor(s). The primary gyroscopic sensor is mounted on the camera to register precisely any repositioning (to be compensated). Either one or two sensors can be used- a Primary IMU (sensor) which is attached to the camera and optionally a Frame IMU (sensor) which is attached to the frame in one of two positions). When two sensors are attached data from both is used by the controller board simultaneously for more precise system stabilization.

The controller itself is compact (17 grams) but directs 1.6 Amps at 20V to each axis, which gives it the power to drive large gimbal motors (80mm to 110mm is quite reasonable) when amperage and voltage levels are observed. This translates to a maximum payload of about 25lbs (the weight of a Red*(TM)* Cinema camera and prime lens) if properly mounted and balanced and depending upon the rate of correction expected and other operational factors. Correction rates (that result) are increased by novel and demanding applications such as mounting to moving or flying vehicles. Many factors must be taken into consideration when determining fitness for a proposed use, but in particular the payload weight, balance, gimbal quality, g-forces that will likely be encountered and the magnitude of wind speed and turbulence all contribute.

1. Overview

Introduction

The system controller board and software are designed and licensed by Basecam Electronics. You can purchase our controller directly from us at our web store (http://www.basecamelectronics.ru/store/) or you may purchase one manufactured under contract by one of our partners (the list of our official partners is available on our web site) http://www.basecamelectronics.ru/wheretobuy/). Different manufacturers may alter the controller slightly (for example, by adding an integrated Bluetooth component or by changing its size etc.). In either case note the the board version and relevant data published on the corresponding manufacturer's web site.

Some of our partners make just the boards available and others make finished gimbal products with preinstalled controllers (http://www.basecamelectronics.com/readytouse/). Gimbals are also available (both with and without motors) but without electronic stabilization system. In these case you will need to purchase a controller (from us as noted above or from one of our partners providing just the boards) and install it yourself. If you decide to assemble a stabilization system yourself, please visit our forum where you can find the necessary information (http://forum.basecamelectronics.com).

We describe in this manual both the controller board itself as well as the multi-platform (software) application for its adjustment. We call the software application (the) Basecam GUI. As noted it may be downloaded from our website and also as noted above it is necessary to get the version of it that is associated with the firmware version that is installed on the board (the versions should match).

The Basecam GUI software uses the java runtime environment and a virtual com port to aid in portability to other systems. Depending on the platform you may need to issue some commands to enable the port, and (on some platforms) it may be necessary to install a serial driver. Once running and connected the GUI looks and runs the same on all platforms. Note that when Bluetooth is employed as the serial bridge (rather than plugging the board into a computer with a usb cable) that it may be necessary to configure the bluetooth device separately from running our software. See below for more details.

Basic connections

The connection scheme for the basic controller board is shown in figure 1:

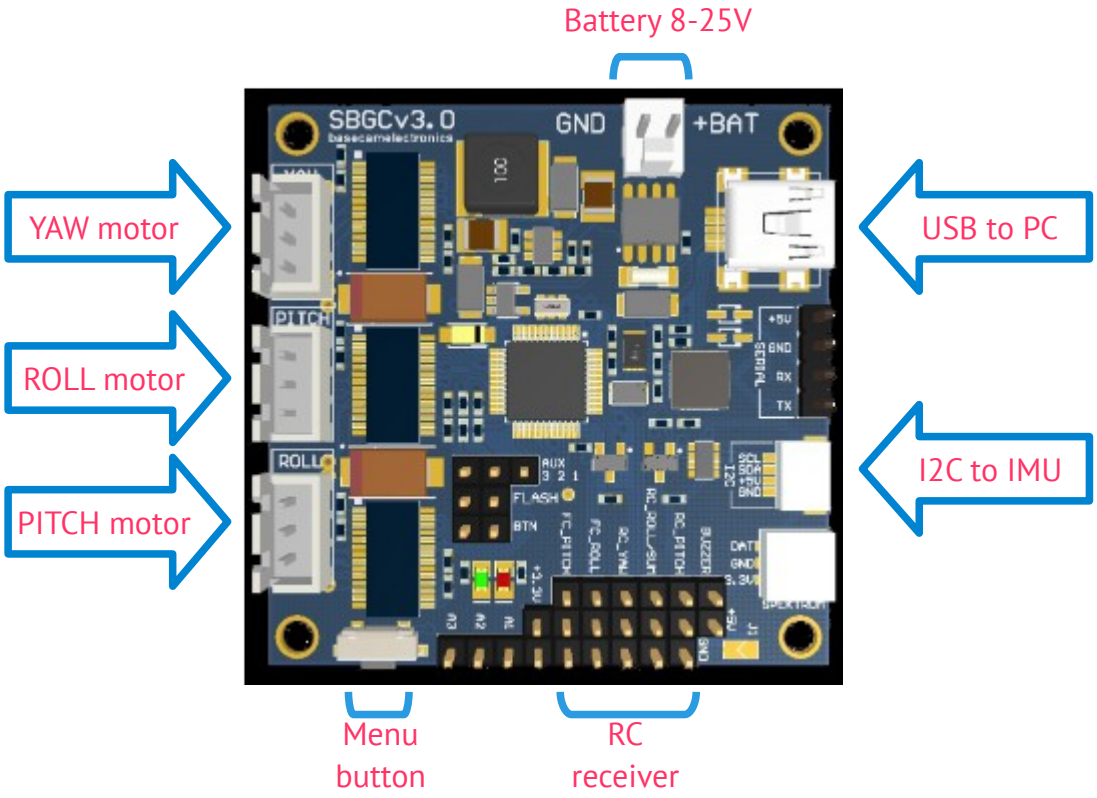


Fig.1 Basic connections

1. The **USB port** is used to connect the SimpleBGC 32bit stabilization board to PC.
2. Gyroscopic **sensor(s)** (IMU's) are connected to I2C slot. When there is a second IMU their outputs are combined with a joiner and in either case a single connection is made to the port as shown.
3. Each axis **motor** is connected to the corresponding motor connection. These outputs are connected directly to the brushless gimbal motors.

   NOTE: It is advisable to pull each motor cable through (and make at least one loop around) a ferrite ring to avoid high frequency interference from affecting the IMU sensors and other electronic devices (both on and connected to the board).

4. The controller board is equipped with a power cable for **connection to a battery.** To avoid connection interruptions it is recommended that you solder the wires to these pins from the corresponding connector to your battery and include some form of physical strain relief. Note Polarity at all times, do not make an incorrect connection. Even a brief (instantaneous) incorrect connection may damage (or destroy) the board (and perhaps the battery).

When handling batteries, never cross terminals, even momentarily. Particularly when handling lithium batteries, accidentally locking terminals may very definitely cause a fire or explosion! Use great care particularly when cutting and soldering battery leads to prevent any contact of opposite poles in a closed circuit.

NOTE: Battery voltage of 8 to 25V is acceptable. If you use a lithium-polymer battery (LiPo), 3S to 5S inclusive are acceptable, where S stands for the quantity of (standard 3.6v nominal) cells in a given battery. Note the voltage maximum for (most) such cells is 4.2V when fully charged. Consequently, a fully charged 3S LiPo is equal to 12.6V and 5S LiPo is equal to 21V. Heed all warning indications regarding safe handling of lithium polymer batteries. Remember that LiPoly batteries use only chargers specifically designed for this chemistry. Never connect a LiPoly battery to a charger not intended for this battery chemistry.

A detailed description of a controller connection within a complete stabilization system can be found in the detailed connection scheme.

GUI installation

First you need to download the latest version of the GUI application from our web site (http://www.basecamelectronics.com/downloads/32bit/). Unpack it in any folder. To start the application you need to have the Java Runtime Environment (managed by Oracle Inc) installed in your system. To obtain the product for your system see http://www.java.com. For each of the systems, in the unpack directory:

To run the Basecam GUI for **Windows**:

- run SimpleBGC_GUI.exe

To run the Basecam GUI for **MAC OS**:

- run SimpleBGC_GUI.jar

ATTENTION: The Basecam GUI uses a virtual COM port. To get that to work (on MacOS) a lock file will need to be created (it uses the lock file to control flow back and forth through the virtual COM port). Due to security constraints, you need to create the lock file yourself. Start terminal (Terminal is an application in the Utilities directory).

Into terminal- type- with great care if you are less experienced:

Make folder "/var/lock" by command:

1. sudo mkdir /var/lock

Change permissions by command:

2. sudo chmod 777 /var/lock

Either allow your system to run non-signed applications by setting this in:
System Preferences > Security & Privacy > General > Allow Applications downloaded from: Anywhere
Or you can allow just this one app to run by answering Open when prompted by the system dialog. In this case, as in the other navigate to the unpack directory and

3. double click (to run) SimpleBGC_GUI.jar

To run the Basecam GUI for **LINUX**:

- run run.sh

Connection to computer

The controller has either a Mini- or Micro-USB (depending on the version). To connect the board to the computer you will need to install a driver to first establish a connection. If the driver is not installed automatically, you can download it — for all operating systems - follow the link:
http://www.silabs.com/products/mcu/pages/usbtouartbridgevcpdrivers.aspx

The "Tiny" version of the driver for Windows can be downloaded here:
http://www.st.com/web/en/catalog/tools/PF257938

After you have installed the driver and connected the controller with USB you will see a new virtual COM port in the GUI in the Connection pulldown. Its name should appear upon connection.

You can connect the controller to a computer and supply power from a battery simultaneously. Again be careful and **observe polarity of battery terminals** because when a USB connection is established, the in-built reverse polarity protection is off (some versions are not equipped with such protection).

Wireless connection

To connect you can also use a wireless connection through a Bluetooth-to-Serial converter on the board side and USB-Bluetooth adapter from the PC side (your pc may of course have built in Bluetooth). On the board side working converters are, for example: HC-05, HC-06, Sparkfun BlueSMiRF and others. The converter should have at least 4 outputs: Gnd, +5V, Rx, Tx and it attaches to the controller at the corresponding slot (located near the USB port) marked with UART (or Serial). Regardless of the boards labeling the board's pins are TTL logic- not RS232.

Bluetooth module connection is described in Appendix B.

> **NOTE:** Bluetooth module should be set for baud=115200 and parity=None or Even. Under None the board can be connected to the GUI with parity set to either. However to update the board firmware through the Bluetooth connection parity on the device must be set to Even. To change Bluetooth module settings, see its manual. For some devices it may be necessary to run serial connection software on a machine and connect it to that machine through an adapter. As noted refer to the device's manual. But be aware it should hardly be necessary as updating firmware is likely better done through a wired connection (the USB cable) anyway. It is likely worthwhile to select a Bluetooth module that already has its baud rate set to 115200 as default as that will be required for a normal connection.

Running the application

1. Attach USB cable (or, if connection over Bluetooth, pair the devices).
2. Run GUI, select COM port from the list in the left corner pulldown of the main window and press **Connect**.
3. When the connection to the board is established all profiles will be read and downloaded and the GUI will display the current profile settings. You can read the board settings again any time by pressing the **READ** button.
4. **Make sure to have installed the latest version of firmware**. To check: open "Upgrade" tab and press "Check update". Update if a new version is available. Note that after updating the firmware you will need to re-download the corresponding version of the GUI and revisit this connection scenario. See section "Firmware Update" for more detailed information.
5. After you have finished editing parameters, press **WRITE** to save them to the persistent memory of

the controller (EEPROM). Only the current selected profile will be saved. To restore the factory settings go to **"Board" — "Reset to defaults"**. All the parameters of the current profile will be set to defaults except for general settings and calibration data. In order to erase the settings of ALL profiles, general settings and calibration data, go to menu **"Board" — "Erase EEPROM"**.

6. To switch over to the settings of another profile, choose the desired profile from the list in the upper right corner (pulldown labeled Profile). It is not required to read the parameters by pressing **READ**. You can save different settings in 5 different profiles. Profiles can be switched over through the GUI or by operating the menu button on the board (the button is implemented as a pair of pins on the board (in the center of the board, part of a group of pins operated by jumpers). A button push corresponds to connecting (briefly) the two pins. Repeating this equates to 'two' button presses. The button logic cycles through four values- each button press causing the load of the next profile. Please note that some settings are shared by all profiles. These settings concern hardware component configuration in particular, as well as sensor orientation, RC inputs, outputs to motors and some others. You can assign random names to profiles. They will be saved on the board and will remain unchanged when you connect to the GUI from a different computer.

2. Step-by-step setup sequence

1. Adjusting the mechanics

Mount the camera on the gimbal's tray and balance the gimbal in all three axes. Stabilization quality strongly depends on balance quality. To check your balance, take the (turned off) gimbal in your hands. Make fast motions along all axes's - try to catch any resonance point by swinging the gimbal back and forth. If it is hard to do - gimbal is balanced correctly.

> **NOTE:** Good balance and low friction allows reduced power levels and still keeps good quality of stabilization.

If you rewound motors by yourself, it's recommended to check electrical resistance and connectivity of your work! With motors removed from gimbal, connect them to controller and set parameters P=0, I=0.1, D=0 for each axis and set enough POWER. Connect main power supply. Motors should spin smoothly, while rolling the sensor. A little jitter is normal due to magnetic force between rotor and stator ("cogging" effect).

Pay great attention to sensor installation. Its axes must be parallel with motor axes. Pay attention to mechanical links. They must be a VERY RIGID and backlash-free. The sensor provides feedback data for stabilization, and even any little freedom or flexibility will cause delays and low-frequency resonances. This can complicate setting of PID and cause unstable work in real conditions (frame vibrations, wind, etc).

2. Calibrating the sensor

The Gyro is calibrated every time you turn the controller on, and it takes about 4 seconds to complete. Try to immobilize the camera sensor as hard as you can in first seconds after powering on while signal LED is blinking. After powering on you have 3 seconds to freeze the gimbal before calibration starts.

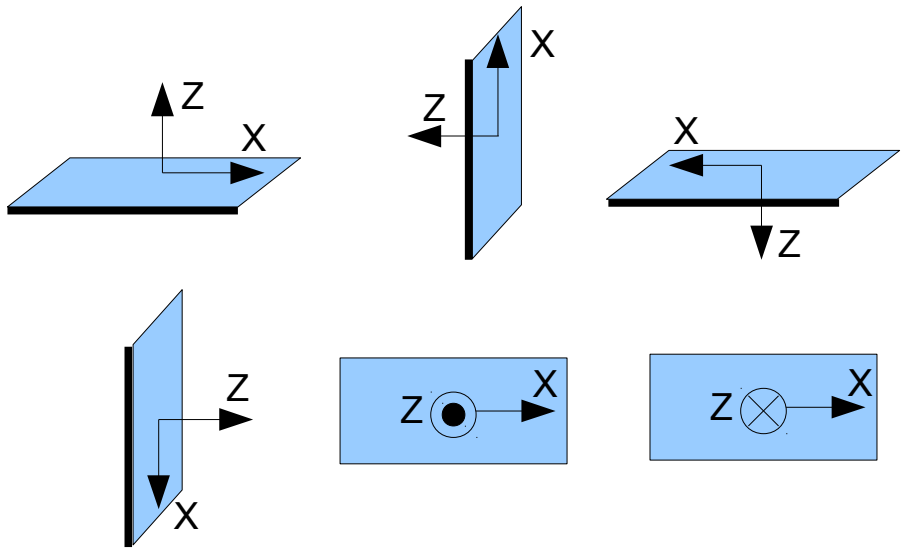
If you activated option "Skip gyro calibration at startup" then the gyro is not calibrated each time and the controller begins operating immediately after powering up. Be careful and recalibrate the gyro manually if you notice anything wrong with IMU angles.

Calibrating Accelerometer

You must perform ACC calibration only once, but it's recommended to recalibrate it from time to time or when the temperature significantly changes. Alternatively you can make a temperature calibration through a full range of possible working temperatures (see Temperature Sensor Calibrating).

- **Simple calibration mode:** set the sensor horizontally, and press CALIB.ACC in the GUI (or the menu button, if it's assigned). The LED will blink for 3 seconds. Be sure not to allowthe sensor to move during calibration. At this step it does not matter how the camera is oriented. You are calibrating the sensor, not the camera!
- **Advanced mode (recommended):** to begin perform calibration in simple mode as above. Then turn sensor successively in order such that each side of the sensor looks up (6 positions at all, including base one). To do this fix the sensor in each position, then press **CALIB.ACC** button in the GUI, and wait about 3-4 seconds (until the LED is stops flashing). The order does not matter but the base position always goes first (because the simple calibration cancels a result of advanced calibration). *You do not have to press the WRITE button at each step, calibration data is written automatically (the data is written when the LED stops flashing for each orientation performed).*

> **NOTE:** Precise accelerometer calibration is a very important for horizon holding during dynamic flying or YAW rotation.



4. Tuning basic settings

- Connect the main power supply.
- Set **POWER** according to the motor configuration (see recommendations below)
- Auto-detect number of poles and motors direction. Do not proceed to next step until proper direction is detected!
- Run auto-tuning for PID-controller, using default settings the first time.
- Adjust PID controller settings if required. To check stabilization quality use the peak indicator in the control panel (shown by the blue traces and blue numbers). Incline the frame by small angles and try to minimize peak values by increasing P, I and D to its maximum. You may use gyro data from the Monitoring tab to estimate stabilization quality too.

  It is better to tune PID with the "Follow Mode" turned OFF for all axes.

  Suggested algorithm for manual PID tuning:

  1. Set I=0.01, P=10, D=10 for all axes. Gimbal should be stable at this moment. If not, decrease P and D a bit. Than start to tune each axis sequentially:
  2. Gradually increase P until motor starts to oscillate (you may knock the camera and see on the gyro graph, how fast oscillation decays). Increase D a little – it should dampen oscillations, and decay time decreases. The lower is decay time, the better.
  3. Repeat step 2 until D reaches its maximum which is when high-frequency vibration begins to appear (you may hear it or feel it in your hands and see noisy lines on the gyro graph). When this begins current P and D values are at maximums for your setup. At this point decrease them a little and go to step 4.
  4. Increase I until low-frequency oscillation starts. Decrease I a little to keep gimbal stable. Now

you have found a maximum for all PID values for selected axis. Repeat from step 1 for other axes.

5. When all axes are tuned in static, try to move gimbal's frame, emulating a real working environment. You may notice that cross-influence of axes may make gimbal unstable. In this case, decrease a little PID values from their maximum for axes that are animating.

Good tuning results in stabilization error of less than 1 degree when you slightly rock the gimbal's frame.

Further steps to improve the precision of stabilization:

- Connect and calibrate external flight controller (see Advanced Settings, External FC Gain).
- Connect, setup and calibrate second (frame) IMU (see Second IMU sensor).

5. Connecting and configuring RC

- Connect one of the free receiver's channels to the RC_PITCH input, observing the correct polarity

In the RC Settings tab:

- Set **SOURCE**=PWM.
- Assign RC_PITCH input to PITCH axis.
- Leave all other axes's and CMD's as "no input".
- For PITCH axis, set **MIN.ANGLE**=-90, **MAX.ANGLE**=90, **ANGLE MODE**=checked, **LPF**=5, **SPEED**=10 (not used in angle mode).
- Connect the battery to the main controller and receiver, and check that RC_PITCH input receives data in the "Monitoring" tab (slider should be blue filled and reflects stick movement).

Now you can control the camera from your RC transmitter, from -90 to 90 degrees. If you are not satisfied with the speed of movement, adjust the **I-term** setting for PITCH in the "Basic" tab.

Try the SPEED mode and feel the difference compared with the ANGLE mode.

Connect and tune remaining axes the same way, as required.

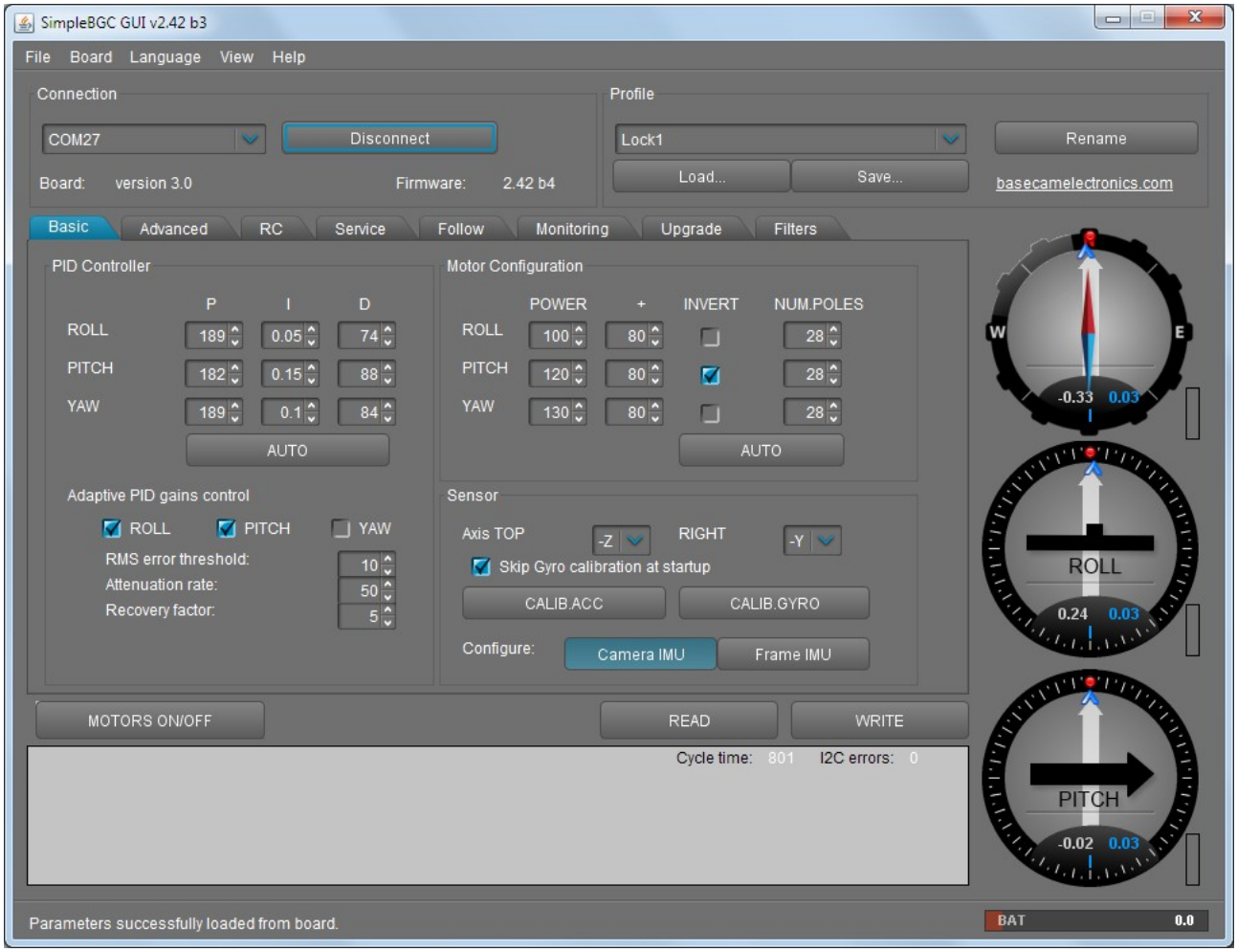
6. Testing gimbal in real conditions

For flight on multi-rotors, connect controller to the GUI and turn ON the vehicle's motors, holding it above your head (and away from your face and hands). Check the vibrations on the camera by using the Monitoring tab / ACC raw data. Try to decrease the level of vibrations using soft dampers.

> **NOTE:** Brushless motors versus traditional servos provide faster reaction, but less torque. That's why it's hard for them to fight against wind and air flows from props. If you are developing multi-rotor frame try to avoid these influences (for example, lengthen arms a bit, or tilt motors away from the center or place the camera above props in case of H-frame). Also bear in mind, when copter moves with high speed, an air flow is deflected and this affects the gimbal as well.

3. The Basecam GUI overview

GUI Structure



The GUI contains different functional blocks:

1. A configuration block in the central part of the window, organized by 'tab':
    - Basic – Basic gimbal stabilization settings. Adjusting these settings is usually adequate to achieve good camera stabilization.
    - Advanced — More precise tuning options.
    - RC – settings to control the gimbal roll/pitch/yaw orientation with RC inputs.
    - Service – Specify the behavior of the MENU button (located on the controller board or mounted externally) and tune the battery monitoring service.
    - Follow – settings related to special mode of the camera control when it follows the frame.
    - Monitoring — real-time sensor data monitoring. This screen is extremely helpful in tuning your gimbal performance. Firmware Update — Firmware and GUI software versions and update options.
    - Upgrade – lets you to check the version of firmware and upgrade if necessary.

- Filters – settings to setup digital filters for PID controller.

2. Connection — COM-port selection and connection status.
3. Profile — Profile selection, loading, re-naming, and saving.
4. Control Panel — graphic visualization of gimbal orientation angles in three axes.
   - *Black arrows are displaying the angles, blue arrows are a 10x time magnification to provide higher precision. Red marks show target angles that gimbal should keep.*
   - *Thin blue lines shows the maximum (peak) deflection from the central, neutral point.*
   - *Blue digits show* peak *deflection amplitude. Using these numbers,* stabilization *quality* can *be estimated.*
   - *Vertical red bars to the right of the scales show actual power level from 0 to 100%.*
   - *Gray arrows shows the angle of a stator of each motor, if known.*

5. READ, WRITE buttons are used to transfer setting from/to board.
6. MOTORS ON/OFF button is used to toggle motors state.
7. At the bottom of the screen, tips, status or error messages (in red color) are displayed . Overall cycle time and I2C error count is also displayed.
8. Battery voltage indicator with warning sector.

Board menu

This menu encapsulates options to Read/Write settings (duplicating READ, WRITE buttons) to calibrate sensors, to reset parameters to their default values, or to completely reset board by erasing EEPROM.

Language menu

The GUI starts in the English version of the user interface. To change the interface language, choose the one desired in the 'language' menu and restart the program.

View menu

You can change a visual theme from the “View” menu. For example, when using GUI outdoor, better to switch to one of the high-contrast themes.

Further in this manual each tab is described in details. At the end of this manual, you can find additional step-by-step tuning recommendations.

4. Basic Settings

PID and Motor settings

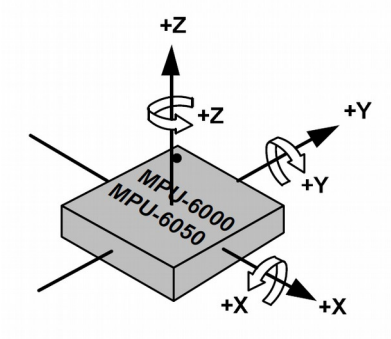
- **P,I,D – PID regulation parameters for all axes.**
  - P – describes the power of disturbance response. Higher values means a stronger response reaction to external disturbance. Raise this value until the stabilization quality of fast disturbances will be adequate. If the "P" value is too high, oscillations of the axis will start to be present. These oscillations will get worse if there are vibrations that reach the IMU sensor board. *If oscillations occur, raise the "D" parameter by 1 or 2 units, and then try to raise the "P" value again.*
  - D – The "D" value reduces the reaction speed. This value helps to remove low-frequency oscillations. A "D" value that is too high can cause high-frequency oscillations, particularly when the IMU sensor is exposed to vibrations. In special cases, it may be filtered out by digital filters (see below).
  - I – The "I" value changes the speed at which the gimbal moves to incoming RC commands and to move the gimbal back to neutral. *Low values result in a slow and smooth reaction to RC commands and to getting back to neutral. Increase this value to speed up the movement.*
- **POWER** – maximum voltage supplied to the motors (0 - 255, where 255 means full battery voltage). Choose this parameter according to your motor characteristics. *Basic tuning:*
  - **Motors should not get too hot!** Motor temperatures of over 80C will cause permanent damage to motor magnets.
  - A Power value that is too low will not provide enough force for the motor to move the gimbal and stabilize the camera adequately. A low power value will be most noticeable in windy conditions, when the gimbal is not well balanced, or if the gimbal suffers from mechanical friction. Slowly lower the Power parameter to find its optimal value. Find the lowest value that still provides good stabilization and adequate holding torque.
  - Raising the power equals raising the "P" value of PID settings. If you raise the POWER value, you should re-tune your PID values as well.
- **"+" - Additional power** that will be add to the main power in case of big error (caused by missed steps). It helps to return camera to the normal position. If main power + additional power is greater than 255, the result is limited to 255.
- **INVERT** – reverse motor rotation direction. It's extremely important to choose the correct motor rotation direction to not damage your gimbal. To determine the correct direction, set the P, I, and D values to 0 and the POWER values to 80 (or higher if your motors don't produce enough force to hold/move the camera). Level the camera tray horizontally and click the AUTO button in the "Motor configuration" settings. The gimbal will make small movement to determine correct motor rotation direction. Wait for the calibration procedure to complete. Then, re-set your PID values and tune your Power values.
- **NUM.POLES** – Number of motor poles. This value needs to be equal to the number of magnets in your motor's bell. During the "auto" calibration process described above, this value is automatically detected. However, this value is sometimes not correctly determined during the "auto" calibration process and will need to be verified and possibly corrected manually. Most brushless gimbal motors are built with 14 poles (or magnets) and utilize a DLRK winding scheme. Count your motor magnets and enter this value if the value is not correct in the GUI.

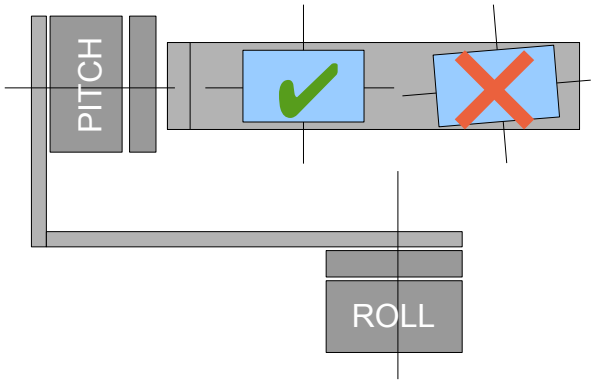
Main IMU sensor

Note: Before tuning your controller, install the camera into the gimbal firmly and ensure your gimbal's center of gravity is leveled as much as possible.

Specify your IMU sensor board's orientation and position on the gimbal . For a standard IMU sensor installation, look at the gimbal from behind just like the camera will view out from the gimbal. Viewing the gimbal in this way, the UP and Right direction will match the Z and X axis. You can place the IMU sensor in any direction, keeping its sides always parallel to the motor axis (be very accurate here, it is very important to precisely align the sensor and mount it firmly). Configure your IMU orientation in the GUI. The correct configuration should result in the following:

- Camera pitches forward – the PITCH arrow spins clockwise in the GUI.
- Camera rolls right - ROLL arrow spins clockwise in the GUI.
- Camera yaws clockwise - YAW arrow spins clockwise.

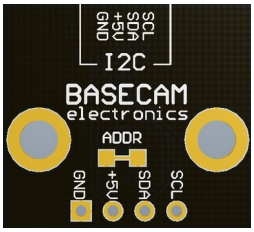




- **Skip Gyro calibration at startup** - With this option, the board starts working immediately after powering it on, using the saved calibration data from last gyroscope calibration call. However, stored calibration data may become inaccurate over time or during temperature changes. We recommend that you re-calibrate your gyro from time to time to ensure the best performance. As an alternative, you can perform a temperature calibration (see Temperature Sensor Calibrating).

Second IMU sensor

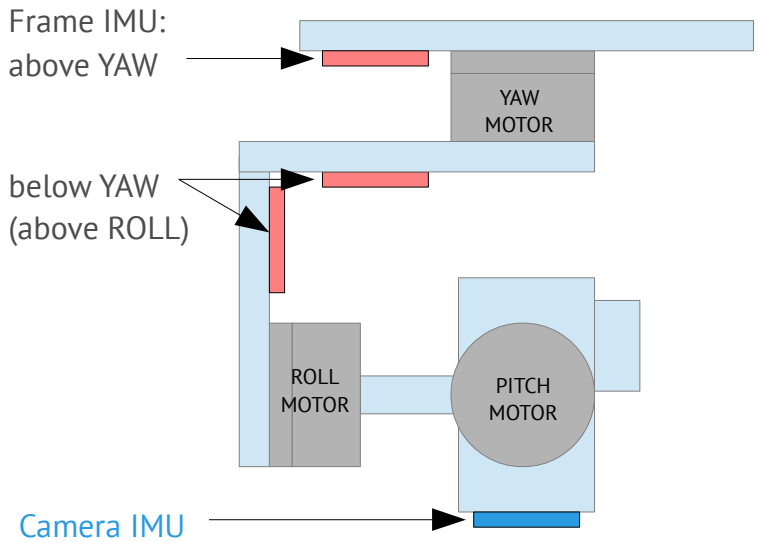
There is an option to install the second IMU sensor on the gimbal's frame. The advantage is more precise stabilization (you may use lower PID's to get the same quality) and knowing frame tilting greatly helps 3-axis systems to extend the range of working angles.



The second IMU should be connected to the same I2C bus as main (in parallel). Sensors should have different I2C-address (Main IMU – 0x68, Frame IMU – 0x69). On the Basecam IMU, address 0x69 may be set by **cutting the ADDR bridge**, located on the back side of the sensor.

Mounting the Frame IMU

There are two options where to place the second IMU: below YAW motor and above it. In case of 2-axis stabilization, there is only one option – above ROLL motor.



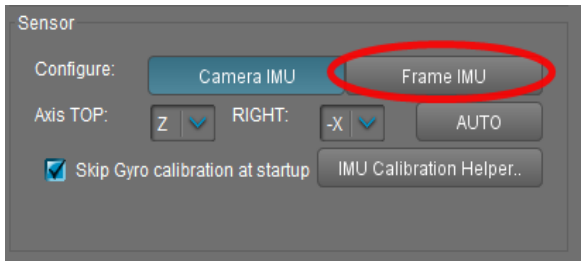
If the sensor is placed **above YAW** motor, it helps to stabilize ROLL, PITCH and YAW motors. But the system becomes less stable during long work (because the frame heading, estimated from the second IMU, may drift with time and auto-correction may not work in all cases).

If the sensor is placed **below YAW** motor, it does not help YAW axis stabilization, but its operation is more reliable. There is a particular option you can choose for this position from: "**Below YAW + PID source**". It means that if Frame IMU is mounted below YAW motor it can be used as a data source for the PID controller. In some cases this can give better result than the main IMU, because mechanical system's "IMU-Motor" becomes more stiff when its length is shorter and its closed-loop operation becomes more stable.

Like the main (camera) IMU, the frame IMU may be mounted in any orientation, keeping its axis parallel with the motor's axis.

Configuring the frame IMU

To configure the frame IMU, first of all set its location in the "**Advanced**" tab, "Sensor" area. Write settings to the board and go to the "Basic" tab. Press the button "**Frame IMU"**:



If the second IMU is connected properly, this button becomes active. It means that all IMU settings now affect the frame IMU. Change sensor orientation (axis TOP, RIGHT) and write setting to the board if necessary (board will be restarted). After restart, calibrate the accelerometer and gyroscope like you did for the main IMU. For the accelerometer you can do simple calibration or extended 6-point calibration.

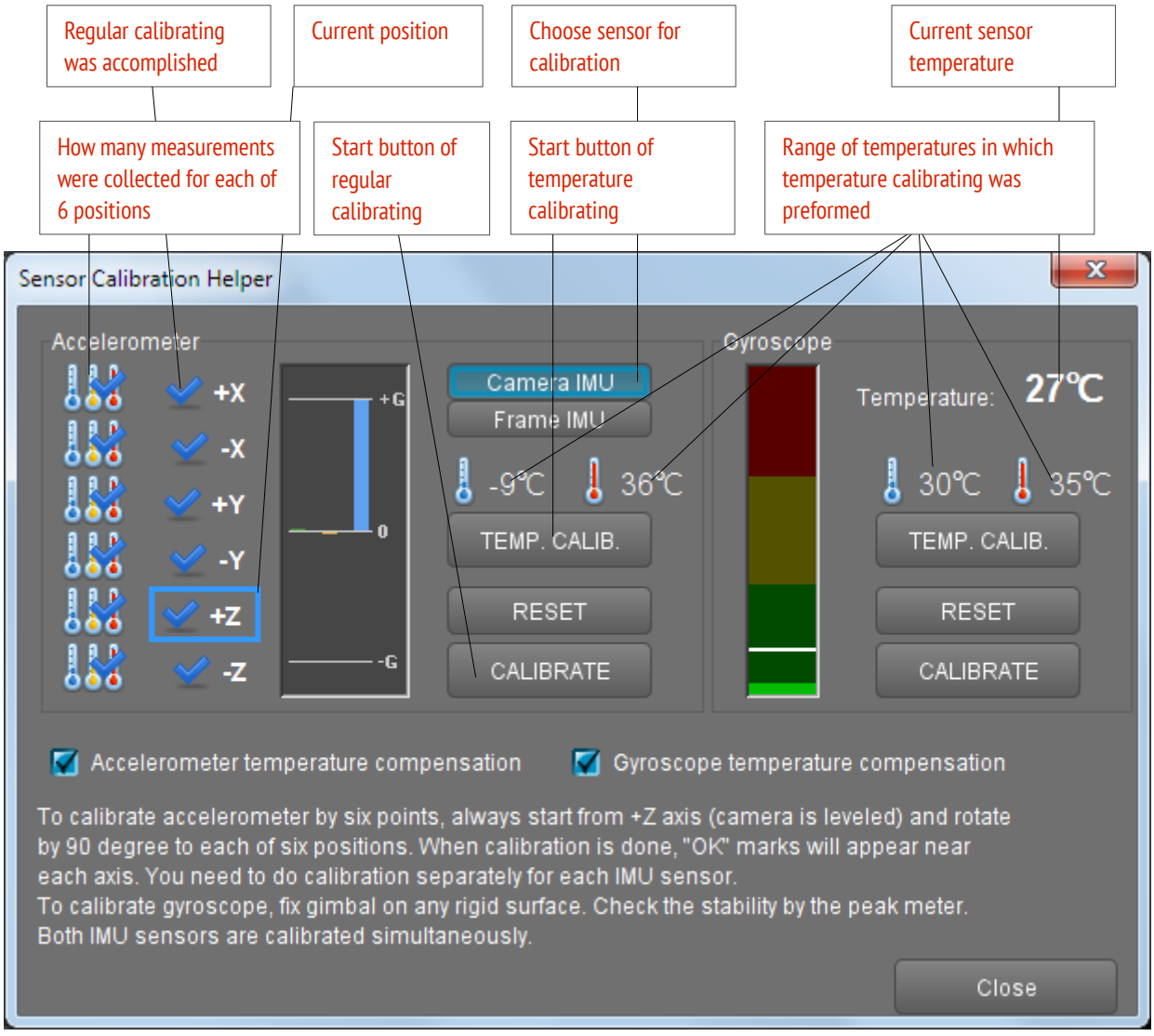
You may notice the right panels with arrows are displaying now angles not for the main, but rather for the frame IMU. Also, in the “Monitoring” tab, accelerometer's and gyroscope's data are for the frame IMU.

Temperature Sensor Calibrating

If the gimbal will be used in a wide temperature range, it is necessary to perform what is called a temperature calibration of the accelerometer and gyroscope. We suggest you do this procedure once properly for at least the temperature range you will be using the gimbal at. This will eliminate the need to repeat calibration due to each change of ambient temperature and results in increased stabilization accuracy for operation within the calibrated temperature range.

Temperature calibration is done through a computer connection with the use of the calibration assistant or offline by setting the corresponding commands for the board's menu button.

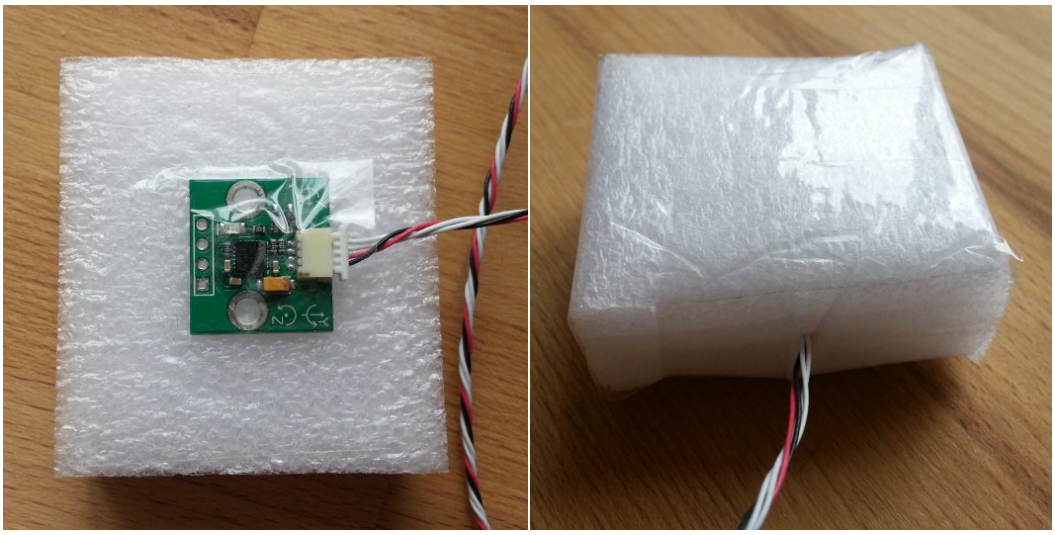
Calibration with the use of GUI is described below. Offline calibration is carried out similarly.



Temperature Calibrating Assistant

During temperature calibration it is important to ensure the slowest possible variation of sensor temperature so that all its parts have the same temperature. In order to ensure this condition the sensor can be protected by a heat insulating shell cut out of a piece of plastic foam. EPP foam or something similar is best- its common in high quality packaging (you will likely recognize it from the picture).

It is better to realize it in the form of a parallelepiped and align the sensor in accordance to its sides — this will make accelerometer calibration considerably easier.



Thermal insulation of the sensor

Temperature accelerometer calibrating

Calibration assignments are made for three values of temperature, starting with the lowest. The 6-position calibration is performed for each (of 3) temperature(s). The process is the same as for 6-point calibration, but you need to press the temperature calibration button instead of the usual calibration button. The steps should not be less than 10 degrees Celsius. For example, if the first six calibrations were carried out at -10°C, the next calibration series should be realized at a temperature not lower than 0°C.

Temperature accelerometer calibration procedure:

1. Connect to GUI, run calibration wizard.
2. Select a sensor (on the camera or on the frame).
3. Reset the previous calibration by pressing RESET and let it restart.
4. Cool the sensor to necessary temperature (for example, by placing it in a freezer), connect to GUI again, run calibration wizard and select the sensor. Check the current temperature indication of the sensor.
5. Calibrate in each of the six positions in a random order. Insignificant temperature variation is allowed during position switching (up to 5 degrees total), but it is desirable to realize the series as quickly as possible. Thermal insulation will help to slow down the sensor heating.
6. Make sure that each calibration (series) done is indicated by a new thermometer icon in a corresponding slot. If the difference to the previous calibration temperature value is less than 10 degrees, the new value will not be accepted and error will be indicated by the system with a flashing LED indicator.
7. Repeat steps 4, 5, and 6 for each of the higher temperature values so that the whole sensor working temperature range is covered.
8. Calibration results check: Accelerometer maximum values in each of the 6 directions are equal to 1G throughout the whole temperature range.

When the calibration assistant shows 18 thermometer icons, the checkbox for "Accelerometer temperature compensation" will switch on.

NOTE: Simple calibration by Z-axis or by 6 points switches the temperature calibration off, but its data is stored in memory. You can switch it back on through the calibration assistant.

Temperature gyroscope calibration

The gyroscope is calibrated under continuous temperature increase; the sensors of the frame and the camera are calibrated simultaneously. Choose the calibration temperature range so that the intended working temperature range for the gimbal is covered.

Temperature gyroscope calibration procedure:

1. Cool the sensors down to the required temperature below zero (for example, by placing them into a freezer), then put them in a place with high temperature above zero and secure. Provide total immobility (hold them perfectly still) and good thermal insulation. It is necessary to ensure slow uniform sensor heating to accomplish a sufficient amount of measurement.
2. Connect the controller to GUI and run the calibration helper. Check current temperature indication of the sensor.
3. Press the “TEMP. CALIB” button in the Gyroscope group. You can also start temperature calibration by pressing a hard button in menu or through the menu item: Board -> Sensor -> Calibrate Gyroscope (temp. compensation).
4. During calibration the green LED indicator is flashing uniformly. Calibration continues as long as temperature increases. If LED indicator starts flashing quickly, it means that the gyroscope is detecting some movement. Ensure total immobility of the sensors. If LED indicator is continuously flashing quickly it means that the sensor is not functioning properly.
5. As soon as the maximum temperature is reached, calibration is automatically finished and the board is restarted so that new parameters can be applied. The checkbox "Gyroscope temperature compensation" switches on.
6. Calibration results check: gyroscope reading when totally immobile equals to zero within the whole temperature range applied during calibration; drifting of axis arrows is absent or very low.

   NOTE: During normal calibration of gyroscope the temperature compensation is switched off, but its data is stored in memory. It can be switched on through the GUI in the calibration assistant.

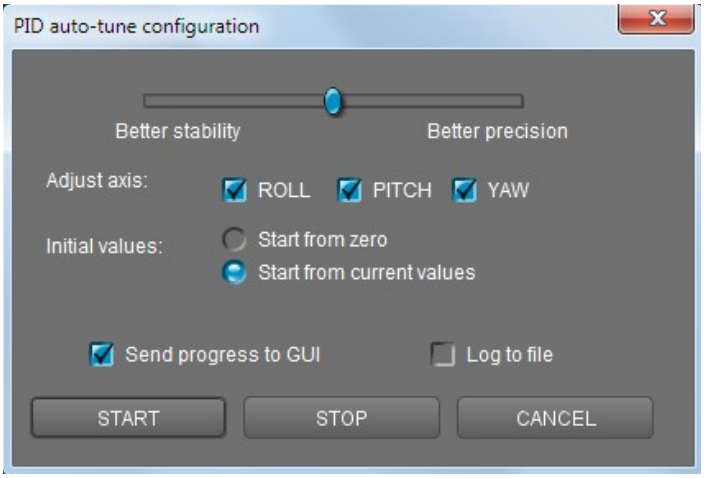
   If gyroscope calibration at system start is ON, it has a priority over the temperature compensation.

5. PID auto-tuning

This feature will be helpful for beginners who often experience difficulties with PID tuning.

Before you start automatic tuning, its very important to properly configure the hardware of your system: motor outputs, “Power”, “Inverse” and “Number of poles” (latest 2 settings may be detected automatically, as described in the user manual). Also, main IMU position should be configured and accelerometer and gyroscope should be calibrated.

Plug-in a battery, connect board to the GUI and press the “Auto” button in the PID parameters section. You will see a dialog window, where you can setup the auto-tuning process:



The slider at the top defines the target of tuning. If its close to “Better precision”, it will try to achieve maximum gain and keep it. If close to “better stability”, it will find maximum gain and then decrease it by 30-50% to make the system more stable.

You may chose which axis to tune. Best results may be reached only if you tune each axis separately. But for the first run, you can tune all axis at the same time.

If you want to use your current settings as start point, select “Start from current values”. Otherwise values will be set to zero in the beginning.

Select “Send progress to GUI” checkbox to see how PID values change in real-time during tuning process.

Select “Log to file” to write PID values together with some debug variables to the file “auto_pid_log.csv”. It may be analyzed later to better understand system behavior. There are a number of tools to plot data from log files, for example http://kst-plot.kde.org

How does it work?

The tuning process does a simple job: it gradually increases P,I,D values until system enters in self-excitation state. Self-excitation means maximum possible gains are reached. Then it rolls back values a bit and repeats the same iteration 2 times. Averaged “good” values are stored as PID settings.

During the process, you should firmly hold the gimbal in your hands. You can place it on a support but check that it provides strong hold, not less than your hands.

After about a minute of work, you can see that PID values have grown big enough and camera is stabilized. Now you can slightly tilt handles in all directions to emulate real-usage conditions. At this point try to find a point where self-excitation occurs (at some particular orientation of the gimbal motions tend to cause self-excitation), and continue tuning system in this point (starting with the “worse case” position).

It is normal that the gimbal starts to vibrate when PID values come close to their maximum. If any motor looses sync due to strong oscillations, you can help to restore it by hand without interrupting the process.

In some cases, you can get a better result (i.e. higher PID gains) if you remove high-frequency resonances before starting automatic tuning. See section "Digital filters" for more details.

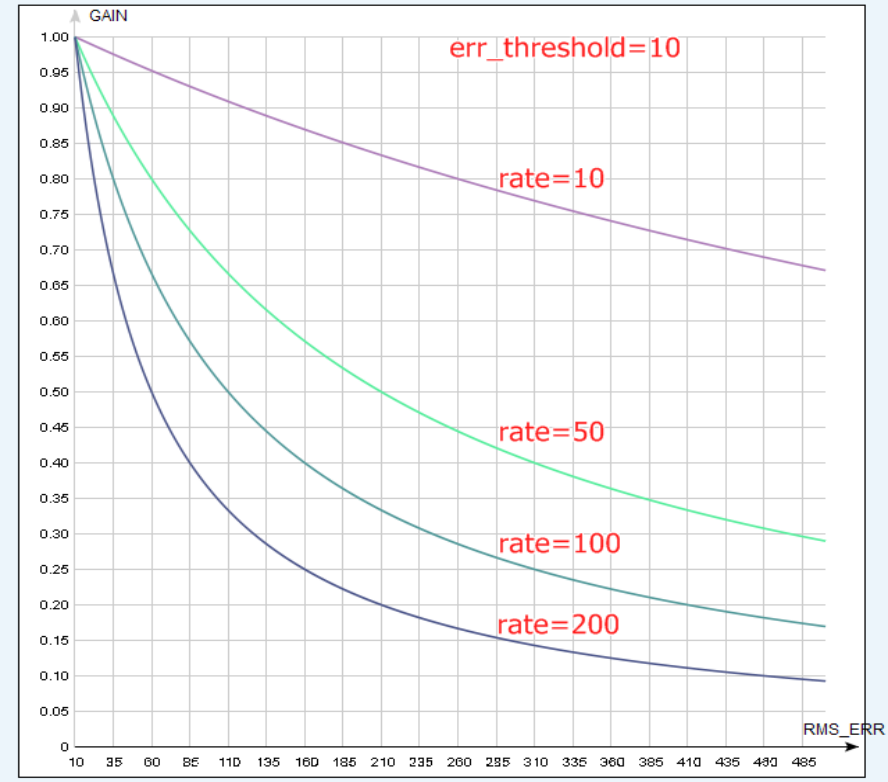
The boards LED is flashing during the tuning process. When the process finishes its job the LED will light ON and new PID settings will be transferred to the GUI.

There is also a corresponding menu command that can start PID auto-tuning without connection to the PC.

Adaptive control of PID gains

This settings group lets to adaptively decrease PID gains when the system becomes unstable due to high PID gains. For example the system may be tuned very well for certain positions, but it may become completely unstable in different position. Self-excitation may cause strong vibration that may negatively affect gimbal construction and may even become hazardous for the camera. For gimbals that have this problem a possible workaround is to use adaptive PID control (another possibility is to change the physical characteristics of the gimbal or its load, improve its balance or employ counter-balances etc) explained as follows.

- **RMS error threshold**, 0..255 - RMS (root mean square) error state variable effectively shows the level of vibrations. When it exceeds this threshold, adaptive PID algorithm comes into action. Recommended value is 10..15.
- **Attenuation rate**, 0..255 - the more this value, the more PID gains are decreased. Choose this value big enough to quiet system quickly. Effect of different rates is shown on the picture:



- **Recovery factor**, 0..10 - defines, how fast PID gains are recovered back when the system becomes stable. Too low of a value may increase a chance that vibration comes back in a short time. Too high of a value may cause worsen of operation (because lowered PID values are kept longer). Recommended value is 5..6

6. RC Settings

The SimpleBGC board provides very flexible configuration of a remote controller. It supports up to 5 digital inputs, including one that supports most popular serial protocols, and 3 analog inputs. It can also output an RC signal in pass-through mode or by Serial API commands. The full RC routing diagram can be found in the Appendix C of this manual.

- **RC Input Mapping** – here you can assign hardware RC inputs to target control channels. There are 5 hardware digital inputs provided on the board for RC Radio control connections and 3 analog inputs for connecting a joystick. Each input can be assigned to control any of three channels, one for each axes, and one command channel. If control for an axis is not needed, leave the option at "no input".
- **RC_ROLL pin mode** – Assigns format for the incoming signal on RC_ROLL pin:
  - **Normal** – incoming signal is in the PWM format which most RC-receivers generally output.
  - **Sum-PPM** - some receivers have this signal output format option. It is a PWM format modification, in which every channel transmits sequentially through one cable. In this case you do not need to connect other channels (read your receiver's user manual to check if it has SumPPM out- how to configure it to do this and which output (channel) it uses).
  - **Futaba s-bus** – receivers made by Futaba may transmit data in a special digital format, up to 16 channels by one wire. Connect it to RC_ROLL pin.
  - **Spektrum** – another digital multi-channel protocol, that is used to communicate Spektrum's satellite modules with the main module, and in its clones. There is a dedicated socket on the board (marked Spektrum) that matches the standard connector.
    Starting from firmware ver. 2.43b7, **you can bind** a satellite (remote) receiver connected to the "spektrum" port, directly from the SimpleBGC board. It will be bound as the stand-alone (master) unit. To start binding assign action "Bind RC receiver" to the hardware menu button and execute this action, or execute the same action from the "Board – Execute command" menu in the GUI. You can select any of 4 different modes prior to start binding, in the "RC" – "Other settings" tab:
    - DSM2/11ms
    - DSM2/22ms
    - DSMX/11ms
    - DSMX/22ms

    Choose a mode that a combination of your transmitter and receiver supports (10- or 11-bit modification does not matter at this moment). Switch to Auto-detection mode after binding is done. If channels are read incorrectly, select 10bit or 11bit modification manually.
  - **SBGC Serial API 2nd UART** – in this mode, RC_ROLL input can handle Serial API commands. It lets us expand the board functionality by connecting external devices, implementing SBGC Serial API protocol. If *RC_YAW* pin is not occupied, it acts as **TX** pin of this UART, allowing to use bi-directional communication. If *RC_YAW* pin is occupied, only **RX** functionality is possible (in other words, external device can send commands to the board, but can't read answers).
    Port settings: 115200 baud, 8N1 or 8E1 - 1 stop bit, 8 data bits, parity 'none' or 'even' (auto-detected after several incoming commands).
- For each control target you can choose appropriate hardware input from the drop-down list.
  - **RC_ROLL, RC_PITCH, RC_YAW, FC_ROLL, FC_PITCH** – are the hardware inputs on the board that accept a signal in the PWM (Pulse Width Modulation) format (excepting RC_ROLL, see above).

Most RC receivers output this signal type.

- **ADC1, ADC2, ADC3** — dedicated analog inputs, marked on the board as A1, A2, A3 and accepts analog signals in the range from 0 to +3.3 volts. For example, joystick variable resistor provides such a signal. Connect A1..A3 to the center contact of variable resistor, +3.3V and GND to side contacts. See Connection Diagram for more info.
- **VIRT_CH_XX** – In case of RC_ROLL pin mode is set to multi-channel signal format, you can chose one of the virtual channels.
- **API_VIRT_CH_XX** – Channels that may be set by Serial API command.

- **Control targets:**
  - **ROLL, PITCH, YAW** - controls the position of the camera
  - **CMD** allows you to execute some actions. You can configure a 2- or 3-position switch on your RC transmitter for a specified channel, and assign it to the CMD channel. Its range is split into 3 sections : LOW, MID, HIGH. When changing the position of your RC-switch, signal jumps from one section to another and the assigned command is executed. The full list of available commands is described in the section "**MENU BUTTON"** of this manual.
  - **FC_ROLL, FC_PITCH** – is used to mark any of PWM inputs to be a signal from the external flight controller. See "External FC gain" section for details.
- **Mix channels** - you can mix 2 inputs together before applying to any of ROLL, PITCH or YAW axis. It provides control of the camera from the 2 sources (joystick and RC for example). You can adjust the proportion of the mix from 0 to 100%.
- **ANGLE MODE** — RC stick will control the camera angle directly. The full RC range will cause a camera to go from min to max angles, as specified above. If RC stick doesn't move camera stands still. The speed of rotation depends on the "SPEED" setting and the acceleration limiter setting.
- **SPEED MODE** — RC stick will control the rotation speed. If stick is centered- camera stands still, if stick is deflected, camera starts to rotate, but does not exceed min-max range. Speed is slightly decreased near min-max borders. Speed of rotation is proportional to stick angle and the **SPEED** setting. RC control inversion is allowed in both of control modes.
- **INVERSE –** Set this checkbox to reverse direction of rotation relative to stick movement.
- **MIN.ANGLE, MAX.ANGLE** – range of the angles controlled from RC or in the Follow mode. For example, if you want to configure a camera to go from a leveled position to down position, set min=0, max=90. To disable constraints, set min=max=0. For ROLL and PITCH axis angles are absolute (i.e. relative to ground) for both "Lock" and "Follow" modes. For YAW axis limits are not applied in the "Lock" mode, and are applied relative to frame in the "Follow" mode. For example, if you set min=-30, max=+30 for YAW in the "Follow" mode, you will be limited by the range +-30 degrees relative to frame when controlling camera from RC sticks or joystick, and not limited when controlling camera by the rotation of frame.
- **LPF** – Signal low-pass filtering. The higher the value is, the smoother the reaction is to stick commands. This filter cuts fast stick movements but adds some delay as a consequence.
- **INIT.ANGLE** – if RC control is not configured for any axis (or there is no signal on the source) the system will keep initial angle specified in this field.
- **RC Sub-Trim** – correction for transmitter inaccuracy.
  - **ROLL, PITCH, YAW trim** – central point trimming. Central point here is PWM 1500. It's better to trim it in the transmitter. But in case of it is not possible (when using joystick, for example), you can use AUTO function in the GUI. Just place stick in center, and press AUTO button. Actual data

becomes new center point. Press WRITE button to apply settings.

- **Dead band** — adjusts dead band around the neutral point. There's no control while RC signal is inside this range. This feature works only in SPEED mode. It helps to achieve better control by eliminating jitters from unintended movement of the stick around neutral point.
- **Expo curve** – adjusts the curvature of an exponential function. Applying more expo means that movements around the center are slower (more precise) but movements of larger values are much greater- with the two extremes transitioning from one to another 'exponentially'. This gives precise control from RC in the range of the small values but rough and strong control near endpoints. Works only in SPEED mode.

- **Limit Accelerations -** this option limits angular accelerations in case of hard RC or Serial control (useful to prevents jerks or skipped steps, smoother camera control, less impact on the multirotor's frame). The lesser the value is the smoother the camera rotation under control is.
- **PWM Output** – a mapping that allows you to pass any virtual channel, decoded from serial input signal, to special pins that can output PWM signal. This signal can be used to drive a hobby servo or IR camera trigger, for example. On the SimpleBGC 3.0 boards, these pins share PWM output function with other functions:

  Servo1 – FC_ROLL
  Servo2 – FC_PITCH
  Servo3 – RC_PITCH
  Servo4 – AUX1

  To enable servo output on any of these pins, make sure that its not specified as RC input in the GUI.

  This feature may be useful if you connect RC receiver by single wire and want to decode signal to the separate PWM channels.

  When connecting regular hobby servo to these ports, there are two options to get +5V to supply them:

  - Connect external power (for example from +5V BEC) to the central pin of any of RC inputs. and **cut (de-solder)** jumper J1 that passes 5V from internal voltage regulator to them.
    WARNING: two power sources joined together, will likely burn each other out because a *switching* DC converter is used to provide 5V supply for the board and it may conflict with the external power source.
  - **Close (solder)** jumper J1 and get +5V from internal voltage regulator.
    WARNING: before connecting servos, check their total maximum current rating, and compare it with the current rating that the board can provide on the 5V line (you can find it in the hardware specifications of the board, for regular "Basecam SimpleBGC 32bit" the version is 1A).

7. Follow Mode Settings

Follow Mode is a special control mode that makes the camera "follow" a tilting of the outer frame, but at the same time eliminates small frame jerking. Several modes of this operation are possible:

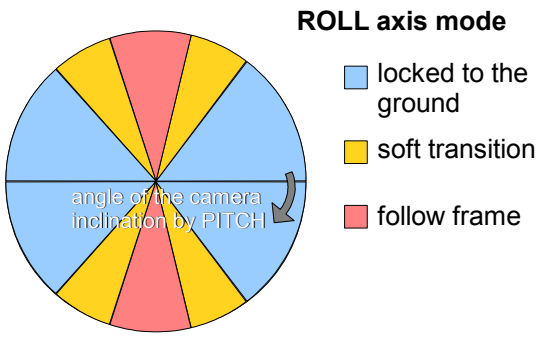
- **Disabled** – camera is locked to ground and may be rotated only from RC.
  - **Estimate frame angles from motors** - this uses the motor's magnetic field for rough estimation of frame tilting, and helps to increase the range of the frame angles where the gimbal's operation is stable. For proper operation in this mode, it is strictly required to calibrate **Offset** setting (see below). Like with the Follow mode, its not recommended to use this option in flight, it is dedicated for hand-held systems only.

    NOTE: This option is ignored if you connect second IMU mounted on the frame, because the data from the second IMU is more precise than from motors.)

- **Follow Flight Controller** – camera is controlled from RC together with the mixed signal from an external flight controller (FC). Almost every FC has servo outputs to drive a gimbal. It feeds the information about the frame angles to these outputs in the PWM format (that servos use). SimpleBGC can get this information and use it to control a camera. It is necessary to connect and calibrate the external flight controller (see **EXT.FC GAIN** settings). After calibration you can setup the percentage values for ROLL and PITCH axis so the camera will follow frame inclinations.
- **Follow PITCH, ROLL** – this mode is dedicated to hand-held systems. FC connection is not required. In this mode, the position of the outer frame by PITCH and ROLL is estimated from the motor's magnetic field. This means that if motor skips steps, position will be estimated incorrectly and operator should correct camera by hands, returning it to proper position.

  WARNING: you should use this mode carefully for FPV flying, because if the camera misses its initial direction, there is no chance to return it back automatically.

  - **Follow ROLL start, deg.** - Set the angle (in degrees) of the camera PITCH-ing up or down, where the ROLL axis enters follow mode. Below this angle, ROLL is in lock mode.
  - **Follow ROLL mix, deg.** - Set the range (in degrees) of the camera PITCH-ing, where the ROLL axis is gradually switched from the 'lock' mode to 'follow' mode (see picture)



  HINT: To completely disable follow for ROLL, set these values to (90, 0). To permanently enable follow for ROLL (regardless of the camera PITCH-ing), set values to (0, 0).

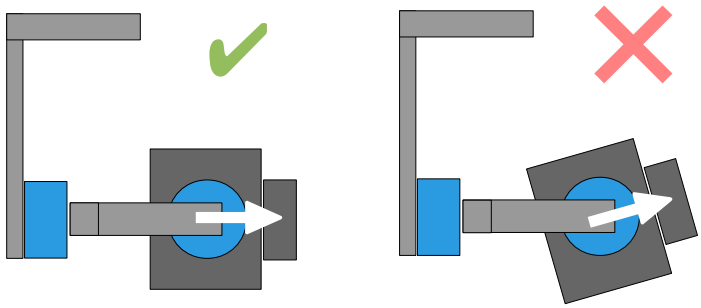
- **Follow YAW** – the same as above, except it can be enabled only for YAW axis. For example, you can lock camera by ROLL and PITCH axis by selecting "Disabled" option, but still control camera by YAW by enabling "Follow YAW" option.

There are additional settings to tune follow mode:

- **Dead band, degrees**: you can set a range where the rotation of an outer frame does not affect the

camera. It helps to skip small jerks when you operate gimbal by hands.

- **Expo curve:** when the expo curve is enabled (i.e. is not flat) a small or medium declination of an outer frame from neutral allows makes only very fine control. But the strength of control exponentially grows when angles of declination become greater. This feature gives considerable freedom in camera operation, from fine and smooth control to very fast movements.
- **OFFSET**: this is a setting that allows you to properly configure the exact initial position of the gimbal. For YAW axis it allows fine adjustment of the camera heading relative to a frame heading. For PITCH and ROLL axis there is an option to calibrate offset automatically. To do this power on the system, hold frame leveled and press **AUTO** button. Don't forget to write setting when finished. If the camera after power on is not leveled, you need to adjust the offset setting.



- **SPEED -** adjust the speed of the camera rotation in the follow mode. Don't set big values that motors can not handle (if motor does not produce enough torque, it will skip steps and synchronization will be broken). In this case an acceleration limiter may help to have high speed but miss steps.
  IMPORTANT NOTE: For high SPEED values (above 50-100) its strongly recommended to set "LPF" parameter greater than zero (set it to 2-3 for example) and "Expo curve" parameter greater than 50. Otherwise, wrong system operation is possible, like vibrations and jerks under follow control, and overshoot of target.
- **LPF** – adjust the low-pass filter applied to the control signal. If this value is set high, fast movements of the handle will be smoothed. But it requires careful operation near the stop point, otherwise the camera will overshoot after you stop rotation of the handle. Its recommended to not set it below 2.

Operation in the Follow Mode

At system startup in the follow mode, keep the frame horizontal and manually adjust the camera to the horizontal position, and adjust it's heading. Camera easily "jumps" between the magnetic poles. Rotate the camera by hands to desired horizontal position- it will stick to the nearest magnetic pole.

Gently rotate and tilt the frame. Turns within ± 45° will control the speed of the camera from 0 to 100%. Camera rotates in accordance with the **SPEED** settings until it's angles are not equal the frame's angles, or until its given restrictions will be achieved.

If the camera moves unpredictably, perhaps its the wrong direction of rotation of the motors and you need to change the **Reverse** flag in the 'Basic' tab .

To achieve smooth motion, increase the **LPF** parameter, increase the **Expo curve**, and decrease **SPEED a**nd **Acceleration limits**. For more dynamic control, change these settings in the opposite direction.

7. Follow Mode Settings

In case of failure of stabilization due to external disturbances the camera can completely lose synchronization with the frame. In this case it is necessary to return it to the proper position by hands. IT IS VERY IMPORTANT to keep the frame horizontal because it is at this point the frame's zero angles are calibrated.

You can switch between modes on-the-fly by activating different profiles, during this the camera will keep its position between modes.

8. Advanced Settings

- **AHRS** - options influencing camera angle determination accuracy.
  - **Gyro trust** – A higher value, gives more trust to the gyro data compared with the accelerometer data when estimating angles. It can reduce errors caused by accelerations during movement, but also decreases gyro drift compensation resulting in horizon drift over time. For smooth flying it is recommended to set lower values (40-80) which will give a more stable horizon longer. For aggressive flying it's better to set higher values (100-150).
  - **Accelerations compensation** – enable use of a physical model of a multi-rotor to compensate for accelerations during flight. This option works only when external FC is connected and calibrated.
- **Serial port speed** — changes baud rate used for serial communication. Decrease it when using over-the-air serial adapters that can't use the maximum speed. The GUI can auto-detect the baud rate configured in the board.
- **PWM Frequency** — sets the PWM frequency used to drive the gimbal motors. Two basic modes are available : **Low** Frequency (in audible range) and **High** Frequency (~22kHz outside audible range). Recommended mode is **High**. There is also third option present: Ultra-high (~30kHz).
- **Motor outputs** — you can assign hardware motor outs for any stabilization axis. For example, you can use the second controller for YAW stabilization and set it up this way: ROLL=disabled, PITCH=disabled, YAW=ROLL_OUT, and connect a YAW motor to hardware ROLL_OUT.
- **Sensor**
  - **Gyro LPF** – adjusts filtering gyro data. It's not recommended to set values different than 0, because it will make adjusting PID controller harder. You can experiment with this.
    Starting from 2.42 this setting is replaced by Low-pass filter in the "Filters" tab.
  - **Gyro high sensitivity** - Increases gyro sensitivity twice. Use this option for big-sized DSLR cameras, in case your PID settings are close to their upper limits, but stabilization is still not good. Increasing gyro sensitivity equals to multiplying P and D values by 2.
  - **I2C Pullups Enable** - turns ON built in I2C pull-up resistors for SDA and SCL lines.
    Use function only if sensor doesn't work properly.
    This option has no noticeable effect for 32bit boards.
  - **Frame IMU** – set the location of the frame IMU. See Second IMU sensor section of this manual.
- **External FC Gain** – Gain value for matching the gimbal data from your flight controller (optional). For better stabilization and utilization of some additional features, knowledge about the frame inclination angles is required. In SimpleBGC the IMU doesn't provide such information. Most FC's have servo outs for connecting gimbals. These outputs should be connected to SimpleBGC controller through EXT_ROLL and EXT_PITCH inputs.
  - Activate gimbal outs in FC and set range limits for angles you generally fly (for example +-30 degrees of frame inclination should equal a full servo range of about 1000-2000).
  - Deactivate all filters and smoothing of FC gimbal settings (if present).
  - In the **RC** tab make sure that inputs EXT_ROLL and EXT_PITCH aren't used to control gimbal. (i.e. are not chosen as a source for any other RC control task).
  - In **Monitoring** tab check availability of EXT_FC_ROLL, EXT_FC_PITCH signals and make sure they are assigned to axes correctly. (Frame roll angle tilting should cause EXT_FC_ROLL change in

approximately the 900..2100 range. The same for pitch.

- Connect power supply and setup stabilization as described above (tune POWER, INVERT, PID).
- Push **AUTO** button in **External FC Gain** group and smoothly incline aircraft's frame to different directions by all axes for 10-30 seconds.
- Push **AUTO** button again to complete calibration. (Calibration will stop automatically after some time too). New gains will be written into EEPROM and shown in the GUI.

NOTE: You may skip this step and leave zero values at initial setup.

9. Service Settings

Menu Button

If you've connected the menu button to BTN connector on the controller you can assign different actions to it. Action is activated by pressing 'button' several times sequentially (1 to 5 clicks) and by pressing and holding (long press).

Available actions:

- **Use profile 1..5** — loads selected profile.
- **Calibrate ACC** – the accelerometer calibration, works the same way as the button selection in the GUI.
- **Calibrate Gyro** – gyroscope calibration.
- **Swap RC PITCH – ROLL** — temporarily swap RC inputs from PITCH to ROLL. In most cases only one PITCH channel is enough to control a camera in 2-axis systems. Before a flight you can assign control from pitch channel to roll, and make a camera precisely leveled. Activating this function again swaps channels back, and saves roll position in static memory.
- **Swap RC YAW – ROLL** — like the previous point.
- **Set tilt angles by hand** – motors will be turned off, after that you can take the camera in hands and fix it in the new position for a few seconds. Controller will save and hold the new position. This function may be useful to correct camera position before flight if there is no RC control connected.
- **Motors toggle, Motors ON, Motors OFF** - commands to change the state of the motors.
- **Reset controller**
- **Frame upside-down** – configures system to work in upside-down position. New configuration is stored to EEPROM and applied even after restart. To switch back to the normal position, execute this command again.
- **Look down** - points camera 90 degree down (or maximum allowed limit under 90 as configured by the *MAX.ANGLE* parameter in the RC tab).
- **Home position** – returns camera to the initial position that is configured by the *INIT.ANGLE* parameter in the RC tab.

WARNING: There is a special action if you press 'menu' button 10 times in series: **full erase of all settings**. Use this option for recovery only, if board is not accessible from the GUI.

Battery Monitoring

On all 32-bit boards (and some 8-bit boards) there is a voltage sensor installed to monitor the main battery voltage. It is used to apply voltage drop compensation (to ensure PID's remain stable during the full battery life-cycle), and to provide power for low-voltage alarms and perform motor cut-off when the battery becomes discharged.

- **Calibrate -** adjusts the rate of the internal multiplier to make measured voltage more precise. You need a multimeter to measure the real voltage, then enter this value in the calibration dialog.
- **Low voltage - alarm** - set the threshold at which to issue alarms.
- **Low voltage - stop motors** - set the threshold at which to stop motors.
- **Compensate voltage drop** - set this option to automatically increase the POWER parameter (which

controls the output power to the motors) which is applied when the battery looses voltage due to the normal discharge process. This becomes unnecessary if the gimbal is fed from a voltage regulated power source.

- **Set defaults for** - select the **battery type** to fill the fields above with the default settings for selected type.

  NOTE: you can add the voltage sensor to old boards in a DIY way, by soldering a voltage divider 33k/10k: 33k goes to the battery "+", 10k goes to the GND and common point goes to pin 19 of the 328p MCU (if this pin is grounded, de-solder it first).

Buzzer

On some boards there is an output to the buzzer (or a buzzer is installed on-board) that is triggered on some events, like notification on errors or confirmation for user actions. Events are configured (turned ON or OFF) in the GUI.

You can connect an active buzzer only (which has an internal sound generator), working from 5V and current below 40mA (check this Digikey product search, for example)

If you have no buzzer connected there is an option to beep by motors. Note that motors can emit sound only if they are powered and turned on.

Status LED

There are 2 LEDs on the board. The **Red** LED lights when the power for MCU is present. The other LED (which is either **green** or **blue** depending on the boards manufacture) gives more specific information about the state of the system:

- **LED is off** — pause before calibration, allows time to take hands off of or to level gimbal.
- **LED blinks slowly –** calibration is in action. Keep the gimbal absolutely still throughout this process.
- **LED blinks fast** — system error, stabilization cannot be performed. To check error description, connect to the GUI.
- **LED blinks fast for short time** – confirmation for user action.
- **LED is on** — normal operation mode.
- **LED is on, but blinks irregularly** – there are I2C errors. Check in the GUI I2C errors counter.

Also, additional LEDs may be present to signal serial communication on RX and TX lines.

10. System Monitoring

In this tab you can see the raw sensor data stream, logical RC input levels and some debug information.

- **ACC_X,Y,Z** – accelerometer data.
- **GYRO_X,Y,Z** – gyroscope data. Helps to determine the quality of P and D settings- for example by disturbing the gimbal by hand and observing the trace. If it looks like a sine wave the D setting is too low and the gimbal tends toward low-frequency oscillations. If some noise is always present even without any disturbance the D setting is too high and the gimbal tends toward high-frequency self-excitation.
- **ERR_ROLL,ERR_PITCH,ERR_YAW** – the stabilization error graph. This is the same as the peak value indicators on the control panel and shows maximum deflection angle.

  NOTE: Each graph can be turned on or off and scale can be adjusted for the Y axis. You can pause the data transmission at any time.

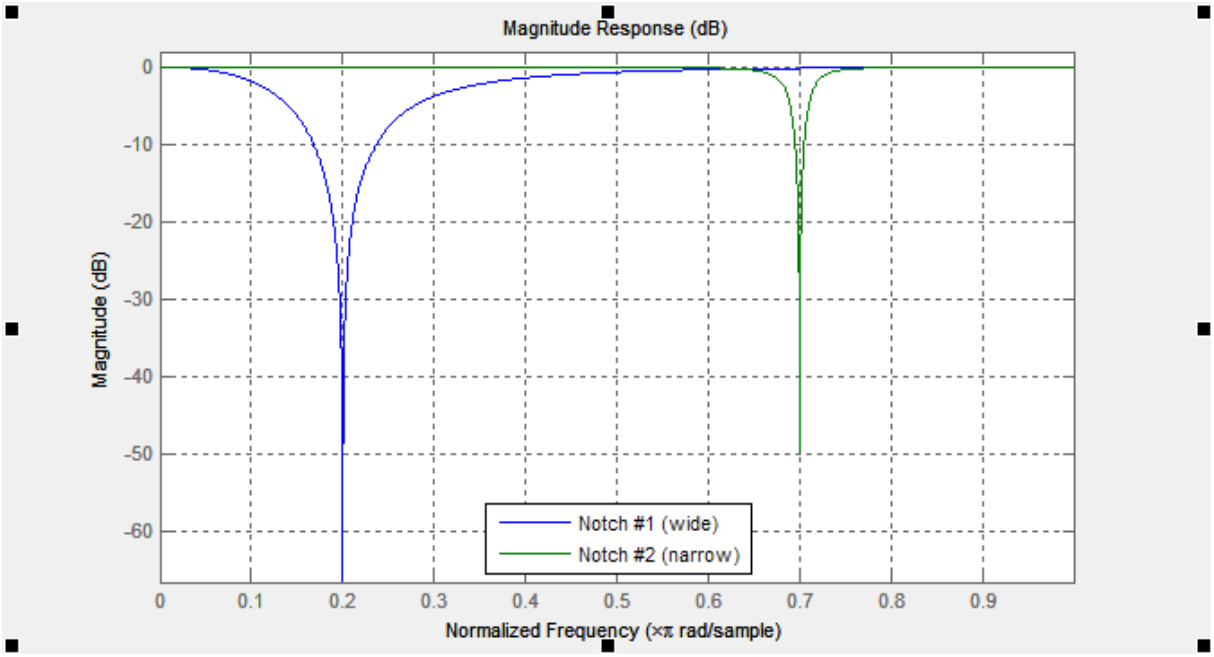
You can receive extended debug information from the board by selecting the checkbox "Receive extended debug info". Useful information you can get from the board:

- **RMS_ERR_R, RMS_ERR_P, RMS_ERR_Y** – RMS amplitude of gyro sensor data. In case of oscillations, it helps to clarify which axis is unstable. It may be not so clear from raw gyro data, because oscillations may have high frequency, far above a frame rate that GUI can receive and display.
- **FREQ_R, FREQ_P, FREQ_Y** – the main frequency of oscillation. If RMS_ERR is too small, this parameter's usefulness is limited.

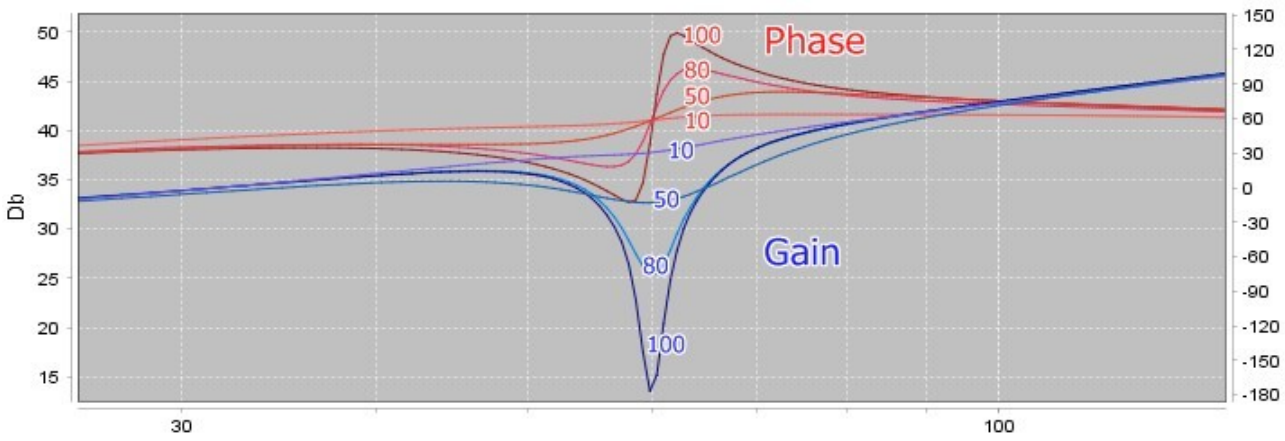
11. Digital Filters

This tab contains settings to configure digital filters that can help to improve the quality of PID controller operation.

Notch filters



These filters can reject narrow bandwidth (resonance). They can help in cases when the system has a pronounced mechanical resonance. Raising the extant feedback gain, oscillations will appear first on the mechanical resonance frequencies and does not depend on variations of P,I,D settings. In this case using one or more notch filters can help to increase feedback gain and get more accurate and stable work of the PID regulator. But this filter will be useless if oscillations appear in the broad frequency range. In this case it is better to use a low-pass filter. With the parameter **Gain** you can control the effect of the notch filter. Set it equal to 100 to get maximum effect, set it <50 to compensate only light resonances. The image below shows different values of the "gain" parameter in the Bode plot build for the PID controller with a single notch filter at 60Hz:



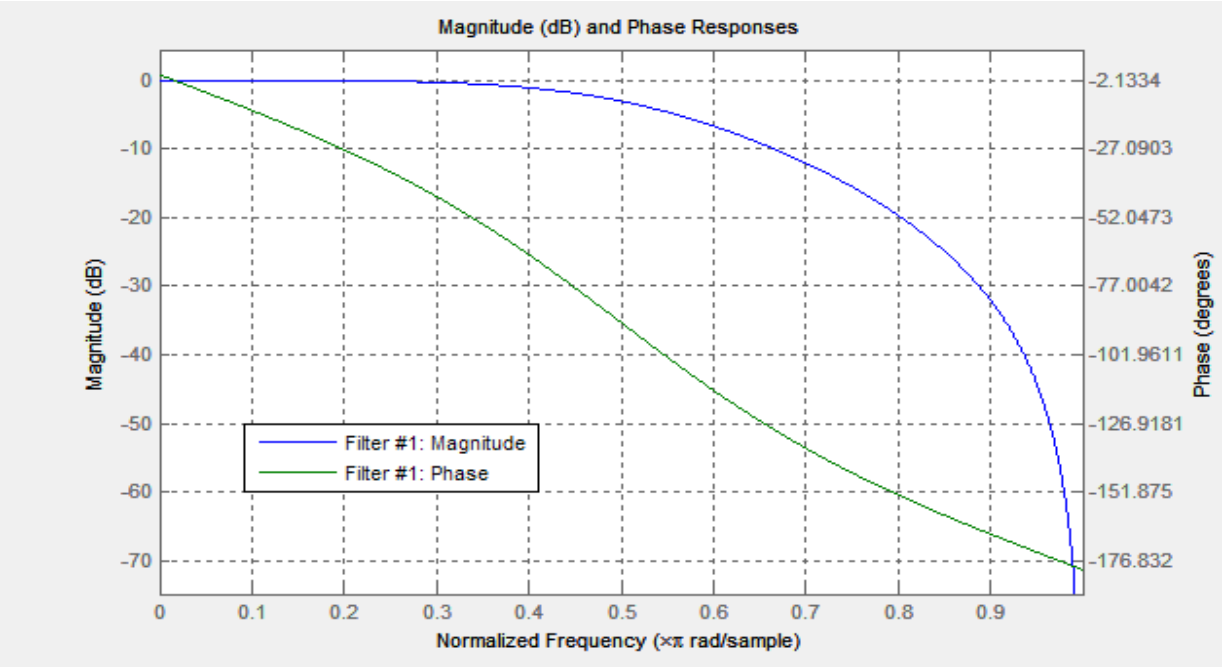
Different values of Gain parameter for single notch filter

Example: gimbal behavior is stable, but when camera tilts downward 60 degrees a strong vibration occurs and this effectively prevents an increase gain of PID (which might otherwise be desired to improve traction).

1. First detect which axis is causing vibration (most). To do this, in the GUI go to the tab "Monitoring" and switch on the following graphics: RMS_ERR_R, RMS_ERR_P, RMS_ERR_Y. Slowly tilt the camera downward until vibrations occur. The axis which shows the greatest growth will show the primary axis responsible. In the example it is RMS_ERR_P, the Pitch axis. A more precise way is to make a test of amplitude vs. frequency response in the Analyze tab.
2. When in steady state vibration mode, look at frequency indication: check another variable in the same tab: FREQ_P. It shows the main frequency of vibration (in our case it has the value 100). Another way is to use a spectroscope (for example, an application for a smartphone that takes an audio signal from mic), but this works only if the vibration is well-audible.
3. On the tab "Filters" fill out the parameters for the first notch filter for Pitch axis: Frequency: 100, Width: 10, Gain: 80 and checkbox "Enabled" is switched on.
4. Write the parameters to board. Now try to re-establish the oscillation. For our example the vibration has been significantly reduced and its frequency shifted to 105Hz. Change the frequency of the filter to 105 Hz. Now the frequency is shifted to 95 Hz. Set back value of the frequency to 100 and increase the bandwidth to 20. Now vibration on this resonance frequency is completely gone. Note, you need to set the bandwidth as narrow as possible. Too broad bandwidth can result in decreased PID efficiency.
5. Having closed one resonance, continue to increase gain of PID (responsible for gain are the parameters P, D). Second resonance occurs on frequency 140 Hz, when we tilt the camera upward. Fill in values for the second notch filter for PITCH axis to cancel this band too, the same way as above.

In this example we have not needed to set filters for the other axes. But it can happen that resonance occurs on more than one axis. Then you will need to set filters on both axes (possibly at the same frequency).

Low-pass filter



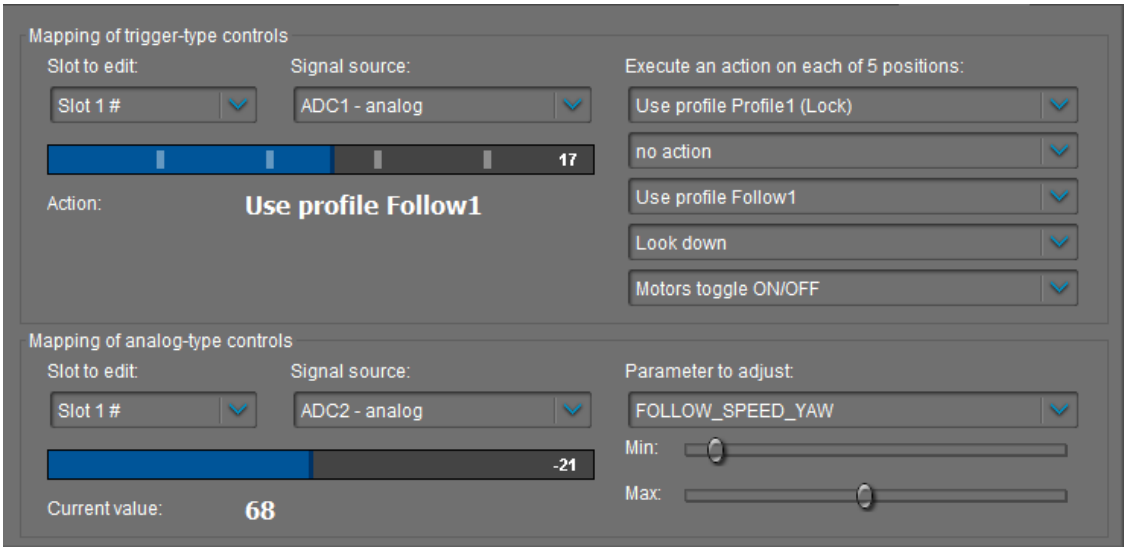
It may be necessary to apply this filter (low pass filter) for large gimbals (heavy cameras with high moment of inertia) or for gimbals with reduction gear. The working frequency range for them are lower than of the lightweight gimbals. But factor D of PID also increases feedback at higher frequency. At high frequencies the response of the mechanical system is typically not sufficiently precise because of many reasons: high-frequency resonances, propagation delay of mechanical impact, nonlinearity due to the backlash and friction, and so on. Due to this the system tends to self-excitation when gain increases. A low-pass filter reduces the gain at high frequency and increases stability of the system. But as drawback a low-pass filter results in phase delay which grows negative near the crossover frequency and can adversely affect the PID stability. This is the reason for the complexity of configuring this filter, and its usage is not always justified.

> **NOTE:** Up to version 2.42 the parameter Gyro LPF was responsible for LPF and provided a first order filter. Now it is not used and changed to a second order filter with more precise tuning of frequency and independent configuration for each axis.

12. Adjustable Variables

Beginning with firmware version 2.43 the controller supports not only remote control of camera angles but also that of a large number of system parameters, allowing their change in real time. Also it has expanded functions of various commands executed remotely - similar to channel CMD but with a much more flexible configuration.

The tab with these settings is displayed after connecting to a 32-bit board with a firmware that supports this feature.



There are two types of control: Trigger and Analog.

- Trigger control is designed for connecting the buttons and switches in such a way that each state of the button triggers a certain command pre-assigned to this particular state. The entire range of the RC signal is divided into 5 sectors whereby the transition from one sector to another triggers the action assigned. Up to 10 slots are available for matching the control channel set designed for 5 different functions.
- Analog Control is designed for fine adjustment of selected parameters by rotating the potentiometer on the remote control panel. It is also possible to switch between fixed values using a multi-position toggle switch that almost all RC transmitters have. Up to 15 slots are available for assigning the control channel to one parameter.

The source of a signal

For both types of Control, the signal source can be:

- **PWM inputs** on the board designated as RC_ROLL, RC_PITCH, RC_YAW, FC_PITCH, FC_ROLL. They take input from standard RC-receivers.
- **Analog inputs ADC1 - ADC3**. They can be connected to analog potentiometers with resistance value of 1-10 kOm (the end terminals are connected to GND and 3.3V and the central terminal is connected to the ADC input in question).
- **Virtual channels** from multi-channel RC. In the event of connection of RC-receivers with a large number of channels over a single wire virtual channels of RC_VIRT_CH1 - RC_VIRT_CH32 receiver can also be used. You can read more on this in the section "RC Inputs".

- **Virtual channels** operated through the Serial API from another device. API_VIRT_CH1 - API_VIRT_CH32.

  TIP: This type of input allows independent developers to create an external control panel with any set of buttons, switches and potentiometers, serviced by a simple microprocessor (for example based on the Arduino software), which reads and transmits the state of control devices data over the wired or wireless serial-interface. Since the tuning of control functions is performed through SimpleBGC_GUI, software for such control panel can be extremely simple. Documentation of protocol «SimpleBGC Serial API specification» is available for download on our website – http://www.basecamelectronics.com

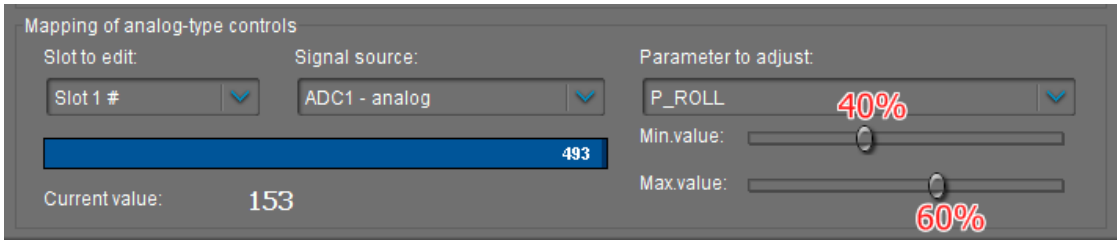
Setting control of the Trigger type

- Select a slot for tuning. Slots, where the signal source is already defined, are marked with '#' symbol.
- Select the signal source. One and the same source can be used for several slots simultaneously (but please make sure that the commands executed for individual slots do not interfere with each other).
- Assign actions to each sector. Possible actions are described in the section "Menu Button". You can leave any sector unused by specifying “no action”.

After activating parameters by pressing button "Write", you will see the current RC signal level on the selected slot (for convenience, the whole range is divided into sectors), as well as the last activated action. You can check in real time whether actions are performed correctly in the case when the level of the signal has changed.

Setting control of the Analog type

- Select a slot for tuning. Slots, where the signal source is already defined, are marked with '#' symbol.
- Select the signal source. One source can be selected to control the number of variables at the same time, which can be convenient to change the value of a group of parameters by single control function.
- Select the variable that must be changed. Decoding of names of variables is presented in Table 1.
- Specify the range of variation by means of the sliders Min. and Max. For example if the full variation range is 0-255, and you need to change it to the range 100-150, you will need to set the slider «Min.» at the mark close to 40%, and the slider «Max.» - at 60%, as shown in the picture:



In this case, the maximum control deviation corresponds to the parameter limit value of 153. Observing the parameter current value in real time it is easy to estimate the required range by moving sliders.

There is a possibility to invert a control, so that when a signal goes up, a variable goes down. To achieve this set Min. slider greater than Max slider.

You may notice that Min. and Max. sliders extend the range of a variable to ±10%. Its done for cases when the RC signal is limited in range and does not cover the full RC range (correction of up to ±500 – and on the screen the blue bar does not reach its limits).

After activating parameters by pressing button "Write", you will see the current RC signal level on the selected slot, as well as the current value of controlled variable.

Table 1. Decoding of names of controlled variables

Parameter's name	Description
P_ROLL, P_PITCH, P_YAW	Parameter of 'P' PID-controller
I_ROLL, I_PITCH, I_YAW	Parameter of 'I' PID-controller multiplied by 100
D_ROLL, D_PITCH, D_YAW	Parameter of 'D' PID-controller
POWER_ROLL, POWER_PITCH, POWER_YAW	Parameter 'POWER'
ACC_LIMITER	Acceleration limiter- unit of measurement: 4°/s (squared)
FOLLOW_SPEED_ROLL, FOLLOW_SPEED_PITCH, FOLLOW_SPEED_YAW	The speed of movement in the mode "Follow"
FOLLOW_LPF_ROLL, FOLLOW_LPF_PITCH, FOLLOW_LPF_YAW	Smoothing of operation in the mode "Follow"
RC_SPEED_ROLL, RC_SPEED_PITCH, RC_SPEED_YAW	Speed of movement when operating from the RC transmitter
RC_LPF_ROLL, RC_LPF_PITCH, RC_LPF_YAW	Smoothing of operation from the RC transmitter
RC_TRIM_ROLL, RC_TRIM_PITCH, RC_TRIM_YAW	Neutral point trimming for channels controlling the camera by ROLL, PITCH, YAW in the speed mode
RC_DEADBAND	The deadband of the RC signal for the camera control channels in the speed mode
RC_EXPO_RATE	Degree of exponential curve depth for the RC signal
FOLLOW_MODE	Follow mode by the PITCH, ROLL angles: 0 - off, 1 - follow the flight controller, 2 - follow the gimbal's frame

RC_FOLLOW_YAW	Follow mode by the YAW angles: 0 - off, 1, 2 - follow the gimbal's frame
FOLLOW_DEADBAND	The deadband for the deflection angle of the frame in the Follow mode- unit of measurement: 0.1 degree
FOLLOW_EXPO_RATE	Degree of the exponential curve depth for the Follow mode
FOLLOW_ROLL_MIX_START	The starting point of the zone transition to the Follow mode, degrees
FOLLOW_ROLL_MIX_RANGE	The length of the zone transition to the Follow mode, degrees
GYRO_TRUST	Trust to gyroscope compared to accelerometer

13. Firmware update

To check if a firmware upgrade is available connect the board and press “CHECK” button. You will receive information about all available versions of firmware and can choose version for upgrade. When selecting a version in the drop-down list its full description is displayed in the text area below. To upload the selected version to the board press the **“UPGRADE”** button. The uploading process will be started. Generally, it takes about 10..30 seconds to finish. **WARNING! Do not disconnect USB cable (or break wireless connection) while firmware is uploading!**

PLEASE NOTE:

- For non-windows operating system, additional steps may be required. See notes at the end of this section.
- For “Tiny” version of the board, you need to install the custom DFU device driver using Zadig utility. Driver installed by default by Windows, does not suit! Detailed instructions on driver installation are provided at the end of this section.

There is an option to configure the system to check for updates automatically. When a new version is issued, you will be prompted to upgrade to it.

If automatic upgrade fails just after downloading firmware from our server (for example, there could be problems upgrading when using a Bluetooth connection under Mac OS), you can try to upload firmware in the manual mode. You can find the downloaded firmware in the '*SimpleBGC_GUI/firmware*' folder and upload this file to the board in manual mode.

Uploading firmware in the manual mode.

This option is intended for special cases when the board becomes bricked (GUI cannot connect to it) and you need to upload special a “recovery” version of firmware, or when you experienced problems with automatic upgrade. *Use this mode carefully and only if you understand what you are doing!*

1. Disconnect any power source and USB cable.
2. Close (set) FLASH jumper on board (attach jumper to the 2 pins marked as 'FLASH', thus shorting them)
3. Connect board to PC by USB cable
4. Run GUI, select COM port (but don't connect!) and go to "Upgrade firmware", "Manual" tab but DO NOT PRESS "CONNECT" IN THE GUI, IF JUMPER IS CLOSED! If pressed, you need to repeat all steps from the beginning.
5. Choose firmware file (**.hex* or **.bin* format).
6. Select board version:
   - **v.3.x (32bit) through Virtual COM Port** – for a regular 32-bit board
   - **v.3.x (32bit) through USB in DFU mode** – for a “Tiny” type 32-bit version. You need to update DFU device driver before proceeding to the next step (see instructions below)
7. Press "FLASH" button and wait for process to finish.
8. Open (remove) FLASH jumper.

If board is alive (you can connect to the GUI), you can upload firmware in manual mode without setting FLASH jumper:

1. Connect to board in the normal way.

2. Choose firmware file.
3. Press "FLASH" button and wait for process to finish.

Upgrading under Mac OS and Linux

Starting from 2.42b7 its possible to upgrade firmware from the GUI under Mac OS and Linux (and virtually, any other OS). The open-source tool **stm32ld** (https://github.com/jsnyder/stm32ld) is used to upload firmware to the board.

> **NOTE:** If the tool failed to run under your OS, you can compile it from source (located in the 'SimpleBGC_GUI/bin/stm32ld-src' folder). Place the result in the 'SimpleBGC_GUI/bin' folder, renaming it to 'stm32ld_mac' for Mac OS, 'stm32ld_linux' for Linux family, and 'stm32ld' for any other OS.

Installing DFU device driver

This driver is required only for "Tiny" version of the board when connected by USB. The open-source utility **dfu-util** (http://dfu-util.gnumonks.org/) is used to write firmware to this board. This driver should be used instead of default driver, installed by Windows.

Windows:

1. Download **Zadig** from the page http://zadig.akeo.ie/ (example)
2. Run Zadig. In the "Device" menu select "Load preset device.." (example)
3. Select file "SimpleBGC_GUI/conf/SimpleBGC 32bit board.cfg"
4. Install driver **WinUSB** (example)

To check that driver is installed properly:

1. Close (set) "FLASH" jumper on the board and connect it by USB to PC (preserving this order exactly!)
2. Windows will find a new device "SimpleBGC 32bit board"
3. Open (remove) jumper, re-connect USB and run GUI to upgrade firmware

Linux:

Most Linux distributions ship dfu-util in binary packages for those who do not want to compile dfu-util from source. On Debian, Ubuntu, Fedora and Gentoo you can install it through the normal software package tools. For other distributions (namely OpenSuSe, Mandriva, and CentOS) Holger Freyther was kind enough to provide binary packages through the Open Build Service.

- Copy dfu-util to "SimpleBGC_GUI/bin/dfu-util-linux" to enable the GUI to find and execute it

MAC OS:

Mac OS X users can also get dfu-util from Homebrew with "brew install dfu-util" or from MacPorts.

- Install MacPorts from http://www.macports.org/install.php
- Find and install **dfu-util** from there
- Copy dfu-util to "SimpleBGC_GUI/bin/dfu-util-mac" to enable the GUI to find and execute it

FAQ and Troubleshooting

Q: Firmware uploading process was interrupted and board is not working now, not responding to GUI. Is it fatal?

A: No, its not (permanently) fatal for your board (its impossible to damage the board in such way). You just need to upload special "recovery" firmware. You can find it in the "firmware" folder, named *'simplebgc_recovery_32bit'*, or download it from our site. Refer to instructions on how to upload firmware in the manual mode (above). Then, you can connect to the board and upgrade to any version, as usual.

Q: I know from somebody that there is new firmware version, but I don't see it when checking for updates. Why?

A: There may be beta versions that are available for beta-testers only, or maybe different versions for different boards. You will receive only stable versions issued for your board by observing the specified version for automatic update.

Q: Can I upgrade firmware from Mac or Linux?

A: Yes, starting from GUI 2.42b7. But check the note above.

Q: My board has no USB connector, but has bluetooth. Can I upgrade firmware?

A: Yes, you can upgrade via Bluetooth the same way as USB. If your board has an integrated Bluetooth module it is already configured properly to work for upgrade. External Bluetooth modules need to be configured to 115200 baud, even parity. If you have problems with re-connection to bluetooth under Mac OS, you can try to upgrade in the manual mode using “FLASH” jumper, as described above.

Q: I am using an external bluetooth module and it works fine with the GUI. Can I upgrade firmware through it?

A: Yes, if you configure module to "Even" parity. To work with GUI, it may be either "Even" or "No" parity, but to upgrade firmware it needs to be configured with "Even" parity only. Look for instruction for your module how to configure it.

Q: Is it required to disconnect battery when upgrading firmware?

A: No, it does not matter if the board is powered from battery, or from USB only.

Q: After upgrade, my GUI can't connect to the board. What to do?

A: **Its important that firmware and GUI both have matched versions. Changes in the firmware usually require changes in the GUI,** so old GUI will not work with the new firmware. You can download the matched GUI from our website. A (hyper) link to a version matched GUI is generally provided in the description of the firmware.

Q: I got an error during uploading: "CreateProcess error=14001"

A: Some required libraries are missing on your system. You need to install Microsoft Visual C++ 2008 x86 redistributable: http://www.microsoft.com/en-us/download/details.aspx?id=5582

14. System Analysis Tool

This tool lets you grab information about system response and displays it in a form of “Bode plot” - amplitude and phase response versus frequency. “System” for analysis purposes may be considered any system that has input and output and unknown transfer function between them.

From Bode plot we can make an assumption about system stability, find problematic areas in frequency domain, and with the help of advanced tools like Matlab find a way to increase the performance of the controller.

This tool is quite complicated to use and is intended for use only by qualified personnel with an engineering degree in systems analysis (control theory).

Collecting data

The main concept is to provide a “stimulus” signal to the input of the system, and then observe a signal on the output. Input and output data is measured with a fixed sampling rate and stored in the CSV file. Then signals are converted to the frequency domain, and a transfer function in the form of the cross-power spectral density (CPSD) is computed. For all frequencies that are present in the input signal, we can build amplitude and phase response plots. When displayed in logarithmic scales, its called a “Bode plot”.

Choosing stimulus signal

The most important things to say about a stimulus signal:

- It should contain a wide spectrum of frequencies. White noise and sine sweep for example, meets this condition.
- System should operate inside its most “linear” range. If stimulus is too low – non-linear effects like static friction and noise in the sensor used for measurement output will significantly impact test result. If stimulus is too high, there is a chance to over-saturate the signal inside the system: actuators may reach their limits, integrators may be clipped by wind-up thresholds, and so on. Proper selection of the stimulus amplitude is very important to get test result close to reality. Maybe several trials will be required to find clear-looking (and therefore useful) Bode plots.
- Generally, the gain of a system decreases at the higher end of the frequency range due to mechanical inertia. We can raise stimulus amplitude at high frequencies to compensate for this drop in gain.

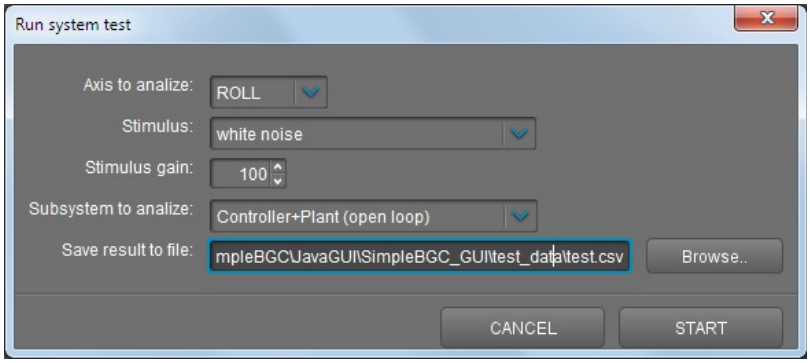
Open-loop vs closed-loop test

In most cases, we are interested in the “open-loop” system response. But if we have an integrator inside the controller, we find that it has big gain for low frequencies that can lead to over-saturation of an output and makes the system non-linear. For example, after integrating gyroscope data, we can get non-zero DC offset of rotation speed. As a result, without negative feedback, camera will go to infinite rotation. Its not a problem for analysis, because DC gain in the output data can be effectively removed. But in real word, camera can't make infinite rotation because of physical limits.

The solution is to run system in the closed-loop mode and to mix the closed-loop feedback signal with the stimulus signal. But near low frequencies, closed-loop system generally operates very well, that means that the output of the system closely matches its input, and the stimulus signal is effectively removed. That is the reason why the closed-loop mode is not perfect for analysis near low-frequency.

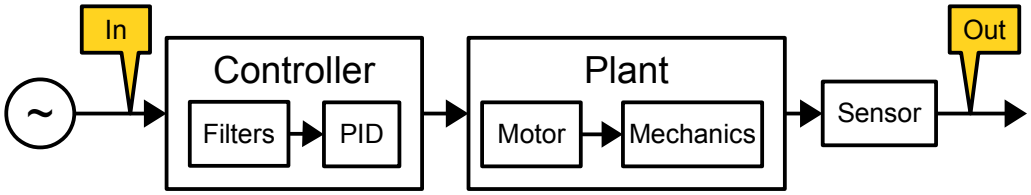
Starting test

In the “Analyze” tab press “Run test..” button and configure the test:

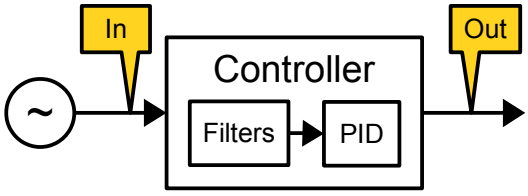


1. Select the axis to test
2. Select stimulus to feed to the input of a system
    - **White noise**: this signal contains the full set of frequencies distributed uniformly.
    - **Sine sweep:** signal with constant amplitude and frequency that goes from 1Hz to 500Hz.
    - **Sine sweep (exponential gain):** the same as above but gain exponentially grows from value set in “Gain” field at the lowest frequency to the maximum at the highest frequency. This type of signal may help to increase the quality of analysis in the high-frequency area, because system gain significantly drops there.
3. Select the **gain** of the test signal. Chose this value experimentally, to keep the system inside its linear range during the whole test, and at the same time have non-zero output.
4. If “white noise” is selected as stimulus, select **cut-off frequency**. Frequencies above this will be removed before passing to the test system. Note that the bode plot for the high-frequency area in this case will be useless.

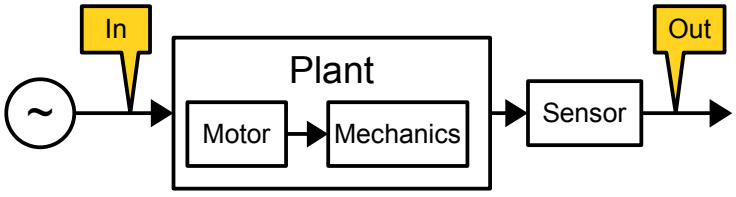
5. Select system to test:
    - **Controller + plant:** input is passed to the PID controller, output is read from the gyro sensor.



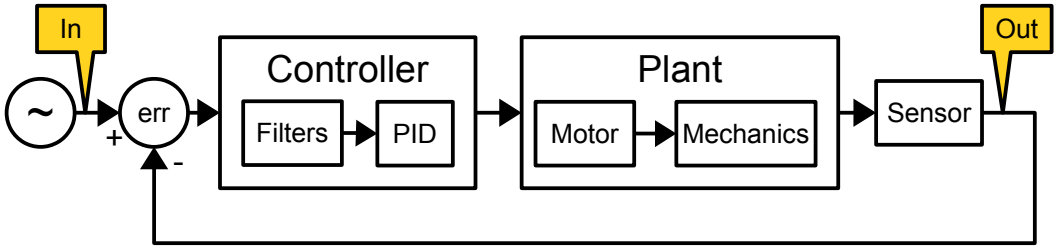
    - **Controller only**: input and output obtained from PID controller. In this test, motors are disabled and test is not visible. Don't set a big gain (to prevent clipping inside controller).



    - **Plant only**: input is passed to motor, output is read from gyroscope sensor. Again, be careful with the gain parameter.



    - **Overall system response:** input is passed as RC input and system tracks it as in normal operation mode. Output should track input signal (gain is close to 0 dB, phase is close to 0 in well-tuned system).



6. Place gimbal on a steady support, power motors ON and begin test. Its important to not disturb the gimbal during the test, especially for open-loop modes. Full test will take about 40 seconds. This time is enough to collect data for good averaging. But you can finish test at any time by pressing "CANCEL".

If something goes wrong during a test (for example, stimulus is too low and you see that the system's response is too weak, or on the contrary stimulus is too big and the system goes outside limits (looses sync) you can stop the test, correct start conditions and repeat the test again.

When the test is finished go to processing of the data collected. In the time-domain graphs, check that the

output of the system is not too low, otherwise test result will be overly noisy and unreliable.

Processing test results

When test is finished it is displayed in the GUI in a form of Bode plot:

But you can analyze grabbed data by more powerful tools like Matlab or similar programs. Therein are wide sets of utilities from system identification to system tuning but high engineering skills are required to make clear use of them.

Reading and understanding the test results

You need to understand the basics of system analysis before reading a Bode plot. There are many tutorials and papers related to this area, for example:

http://support.motioneng.com/utilities/bode/bode_16.html

In few words, on the "*Plant only*" response graph you can see the response of motors and mechanics, and check for potential mechanical resonances.

On the "*Controller+Plant*" response graph you can find the gain margin (0 minus amplitude at frequency, where phase crosses -180 degree) and the phase margin (180 minus phase at frequency, where amplitude crosses 0dB). The basic principle of stability: phase margin should be greater than 30 degrees. Gain margin should be kept in the range -3..-6 dB. The bigger negative values means a more stable system but less accurate tracking of errors. If the gain or phase margin is close to zero, the system is unstable and tends toward self-excitation at those frequencies where zero margins are detected.

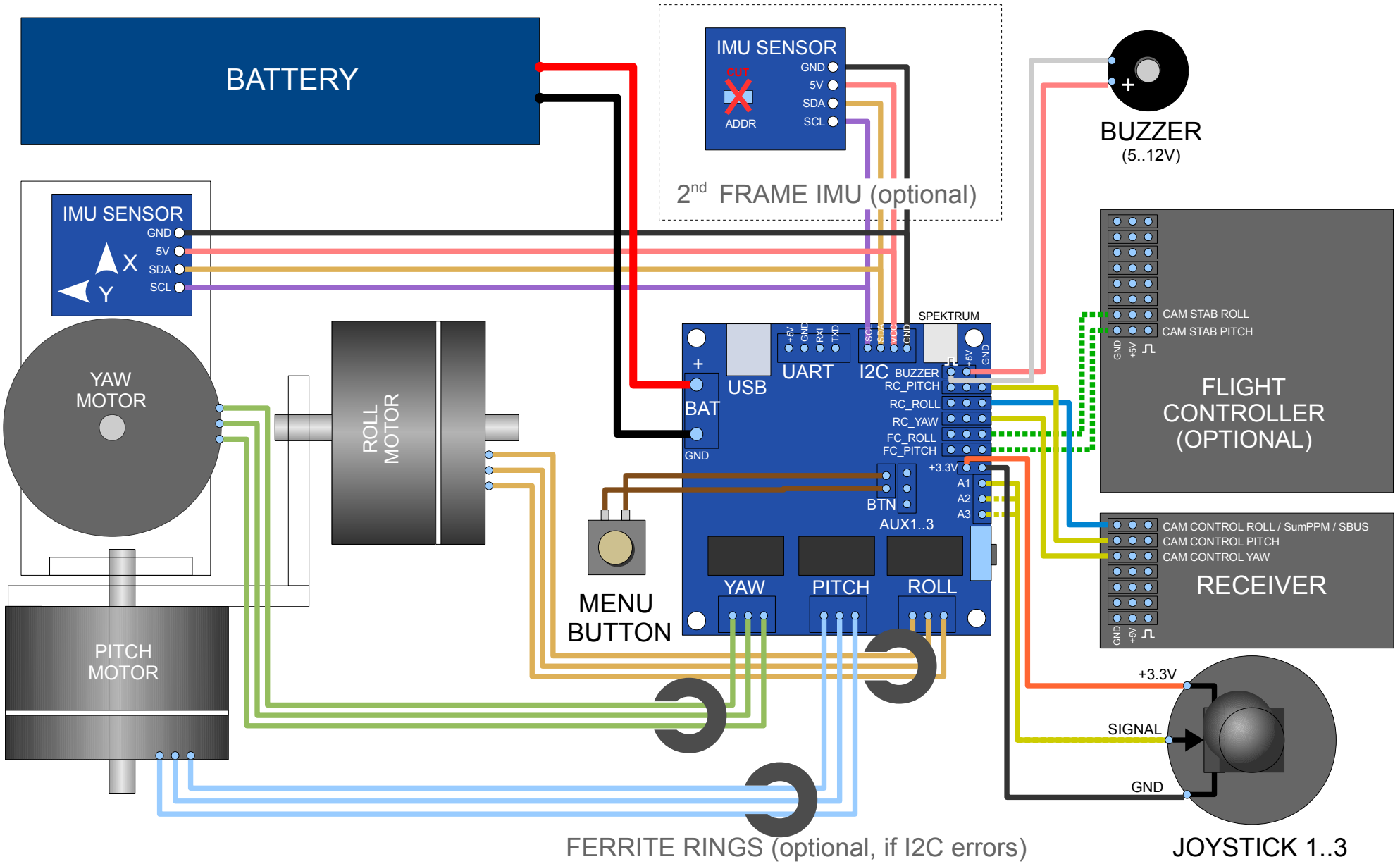
On the "*Overall system*" response graph you can find how effectively a system tracks its input signal on different frequencies. You can estimate working frequencies where gain is close to 0dB and phase is close to zero. Bumps on the gain plot can show potential resonances.

On the "*Controller only*" response graph you can see how the PID controller affects amplitude and phase of the input signal and see the contribution of digital filters.

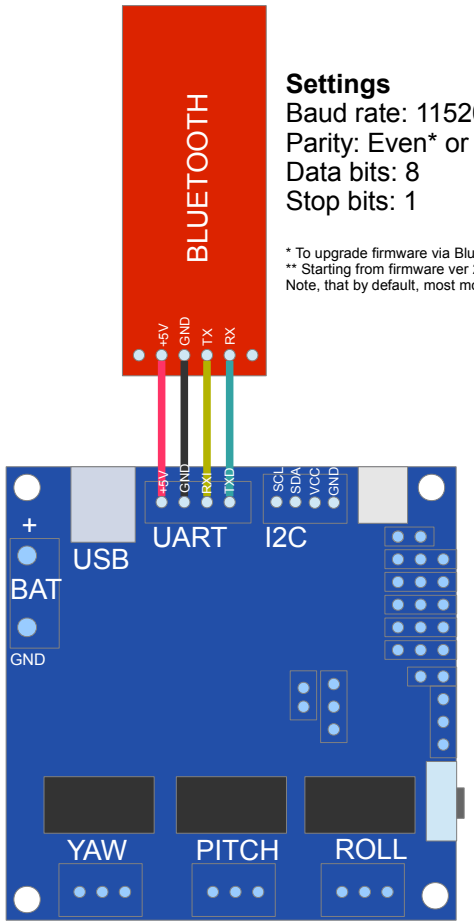
15. Possible problems and solutions

Problem	Possible causes	Solutions
Motors don't spin	-Power supply is not connected -Supply polarity inverted -POWER set to 0	-Check all connections -Set POWER between 50..200
Camera is trying to align, but falls back	-Camera is not balanced -It's an error in motor windings, or one phase is broken - POWER is not high enough	-Balance camera -Check motor winding - Increase POWER parameter
During fast YAW rotating, camera deflects by ROLL, and then slowly gets to horizon.	-Bad accelerometer calibration -Sensor is not in parallel with motor axes	-Make advanced ACC calibration by 6 positions -Align sensor with motor axes
During fast motion with acceleration, camera deflects, and then slowly gets to horizon	-This is normal effect of accelerations	-Try to increase Gyro Trust in Advanced tab
YAW arrow slowly spins in the GUI	-Slow drift is normal (less than 1 degree/minute). It's because of gyro drift over time.	-Note sensor Immobility during gyro calibration -Re-calibrate gyro
Camera slowly drifts by any or all axes just after power on	- Bad gyro calibration	-Re-calibrate gyro
Clicks and crunch are heard during work. LED is synchronously blinking.	-I2C errors present. Errors are possible if sensor wires are too long, or motors outputs affect sensor by capacitive linkage (signal and power wires are run close to one another and there is capacitive linking).	-Shorten sensor wires; -Lower pullup resistors values on the sensor board; -Install a spike LC-filter on motor outs (make 2-3 turns of motor cable through ferrite coil); - Install spike LC-filter on sensor wires (same as motor filter);
High-frequency oscillations.	-Feedback self-excitation as a result of high D parameter	-Check the graphs to understand on what axis the problem is and lower D value.
Low-frequency oscillations.	-Feedback self-excitation as a result of high D parameter or low P parameter.	Lower P, increase D
GUI cannot connect to the board.	-Wrong COM-port selected -GUI and firmware versions dont match.	-Try different COM-ports -Upload the latest firmware, and download matching GUI version.

SimpleBGC 3.0 (32bit) connection diagram



SimpleBGC 3.0 (32bit) bluetooth connection



Settings
Baud rate: 115200
Parity: Even* or None**
Data bits: 8
Stop bits: 1

* To upgrade firmware via Bluetooth, only 'Even' parity will work.
** Starting from firmware ver 2.41, 'None' parity is supported, too. Note, that by default, most modules configured with 'None' parity.

SimpleBGC 32bit RC signal routing diagram

firmware ver. 2.43