SimpleBGC 32bit 3-Axis Software User Manual

Board all 32bit versions
Firmware v. 2.63 and above
GUI v. 2.63 and above
Last edit: 6. Sep. 2024

© Basecamelectronics® 2013-2024

CONTENTS

2. Overview....3
3. Step-by-step setup sequence....9
4. The Basecam GUI overview....13
5. Hardware settings....17
6. Stabilization settings....30
7. PID auto-tuning....35
8. RC Settings....44
9. Follow Mode Settings....52
10. Service Settings....55
11. System Monitoring....67
12. Adjustable Variables....68
13. Firmware update....74
14. System Analysis Tool....78
15. User-written scripts....83
16. Encoders....84
17. Magnetometer sensor....86
18. Bluetooth module configuration....90
19. Using an external IMU sensor to improve the precision of stabilization....92
20. Support of the MavLink protocol for FC connection....100
21. Correction of the motor's "cogging" effect....106
22. Using 4th motor for a gimbal frame positioning....109
23. Tips on designing mechanics of a gimbal....113
24. Encoders non-linearity calibration....116
25. Possible problems and solutions....119
26. Appendix A: versions comparison chart....121
27. Appendix B: running GUI on Mac OS PC....123

© Basecam Electronics® 2013-2024

2. 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 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 downloaded from the Basecam website. Note that the GUI software version should match (or be greater than) the firmware version deployed on the board.

Also needed ultimately is a gimbal with 1, 2 or 3 brushless direct-driven 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 mass-centered at that point. Gimbal construction should be very stiff – the only motion that is acceptable, is a rotation of motor's shaft. Flexible arms or shaft bearing end play may seriously impact the quality of stabilization.

Additional optional components such as switches, joystick operation and interfaces to remote control devices (PWM, Sum PPM, or S.Bus from a standard RC receiver) are described in detail farther on.

SimpleBGC32 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. Stabilizing is accomplished by driving gimbal motors in response to reception of a signal from the gyroscopic sensor(s). The primary gyroscopic sensor is mounted on the camera to register precisely any rotation (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, a data from both is used by the controller board simultaneously for more precise system stabilization. To improve system performance, optional rotary position sensor (encoder) may be installed on each motor. More info about advantages and requirements of using encoders, you can find on the page http://www.basecamelectronics.com/encoders/

SimpleBGC32 implements a simple Serial API protocol, that allows any external device to communicate with the gimbal controller and introduce a great possibility to apply stabilization solutions in many areas. Serial API specification and examples can be found at the page http://www.basecamelectronics.com/serialapi/

© Basecamelectronics® 2013-2024

2 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 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 pre-installed 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. It may be done from an external software, or using our GUI ("Board" – "Configure bluetooth" utility). See below for more details.

Basic connections

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

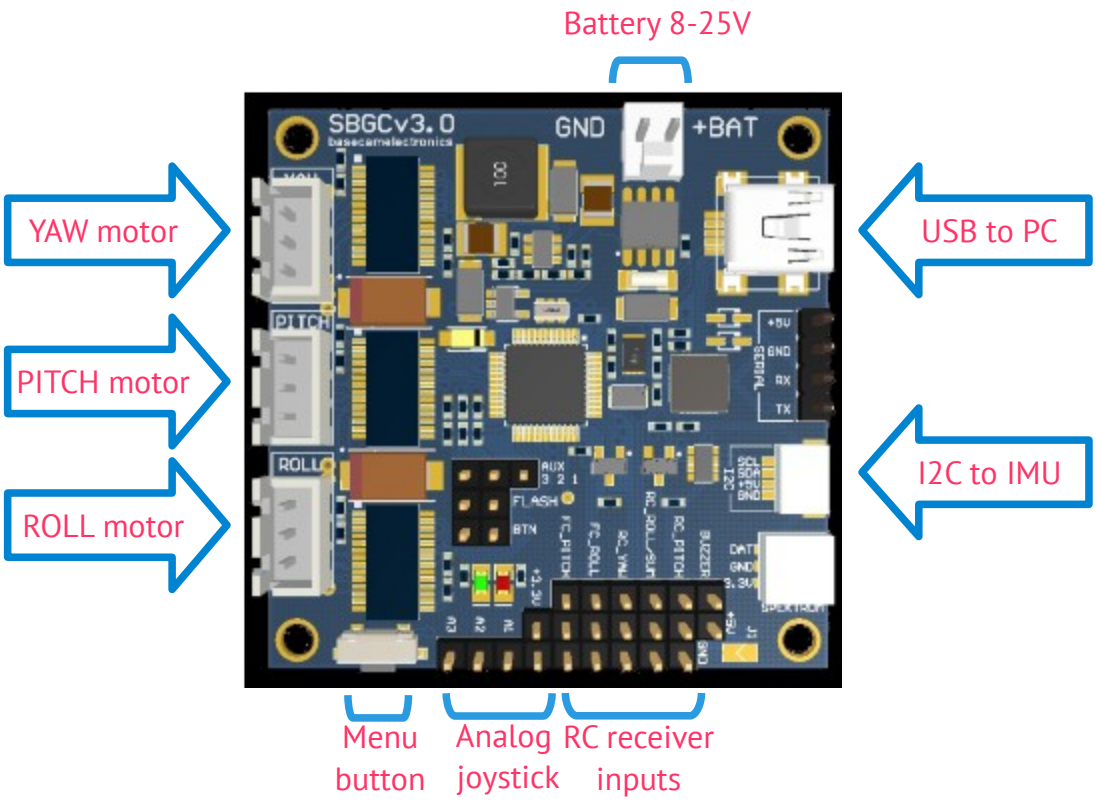


Fig.1 Basic connections

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 an Y-cable and in either case a single connection is made to the port as shown.
3. Each **motor** is connected to the corresponding motor output. If any output is not used, disable it in the GUI. Motors are configured in the "Hardware" tab.
   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.** Observe polarity at all times, do not make an incorrect connection! Even a brief (instantaneous) incorrect connection may damage (or destroy) the board.

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: Maximum allowed battery voltage is defined for each controller individually in its specifications. If you use a lithium-polymer battery (LiPo), and it's marked 3S, that stands for the quantity of (standard 3.6v nominal) cells in a given battery. The maximum voltage of a cell is 4.2V when fully charged. It means that a fully charged 3S LiPo is equal to 12.6V and 5S LiPo is equal to 21V. Observe 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:

Windows:
Execute *SimpleBGC_GUI.exe*

MAC OS:
See Appendix B: running GUI on Mac OS PC

LINUX:
Run script *run.sh* If there is 'no permission' or 'path not found' error, create directory '/var/lock' and give permissions to write into it.

Running GUI on high-DPI displays

If GUI window appears abnormally small – you have a high-DPI display or font settings > 100% is not applied – you may need to perform additional actions to make it looking better.

Option 1: Download and install **Java 9** JRE from Oracle. Allow installer to remove old versions.
At the moment of writing this article, Java 9 download was located here:
http://www.oracle.com/technetwork/java/javase/downloads/jre9-downloads-3848532.html

NOTE: If you have other Java applications in your system that are not compatible with the Java 9, you can keep old JRE, but you have to manually specify the path to the JRE 9 in the "run_console.bat" file and start GUI using it.

Option 2: In Windows 10, you can change the "compatibility" property of the Java executable. Note that it affects all Java-based applications.

- Find **java.exe** in the JRE or JDK installation folder
- Right click -> "Properties"
- Go to the "Compatibility" tab
- Check "**Override high DPI scaling behavior**"
- Set "**Scaling performed by**" to "System"
- Repeat the same for the **javaw.exe**

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:
https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers

NOTE: For the "Tiny Rev. A" version the driver for Windows can be downloaded here http://www.st.com/web/en/catalog/tools/PF257938. This is latest official driver from ST company. But it was

reported, that it does not work under Windows 8. In this case, try previous version, that should work: http://www.basecamelectronics.com/files/drivers/VCP_Setup.zip. Normally, new COM port should appear in the Device Manager, under "COM and LPT Ports". If it's not true and new device is located under "Universal serial Bus devices" as "Unknown device", you need to upgrade driver manually, selecting "STMicroelectronics Virtual COM port" driver.

WARNING: the version 6.7.4 of the CP210x driver may work improperly with the older versions of GUI (all versions prior to 2.63b0), causing a big delays in serial communication. You can downgrade to previous version of a driver (can be downloaded by the link https://www.basecamelectronics.com/files/drivers/CP210x_Windows_Drivers_6_7_2.zip), or update GUI to the latest version where this problem is solved.

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 dropbox. 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 controller to the GUI, 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 other Bluetooth 2.1-compatible modules (BLE modules like HM-10, HM-11 are not supported!). 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 shown in the 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**. Working with different baud rates is possible (just change parameter Hardware->Serial Port Speed to match module's baud rate) but some operations like realtime data monitoring, will be slowed down, so better to configure bluetooth module. To change Bluetooth module settings, see its manual. Starting from firmware version 2.55, there is a tool under "Board" → "Configure bluetooth..." that can configure most popular bluetooth adapters (see Bluetooth module configuration).

Serial-over-Network (UDP) and TCP/IP connection options

These types of connection allows to configure SimpleBGC controller remotely, when it is physically connected by UART to another device, that can communicate with the GUI over network (Wi-Fi, Ethernet, Internet).

- **UDP:** Before connecting, you have to configure local port where GUI listens for incoming UDP messages, and remote host and port to send outgoing messages (optional). You can do it in the "File → Settings.." menu. If remote port and host is not specified, it will be obtained from the first incoming UDP message.
- **TCP/IP**: Use this type of connection to work with the WiFi-to-UART transparent bridge. In the settings you have to configure remote host and port. This information comes in a specification of

particular device. For the "Basecam WiFi UART adapter", host is **192.168.4.1** and port is **23**. Note that first you have to connect to WiFi spot manually using a wireless network manager of your PC.

Running the application

1. Attach USB cable (or, if connection over Bluetooth, pair the devices. Default password is 1234 or 0000, generally).
2. Run GUI, select COM port from the list in the left corner dropbox 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 currently selected profile will be saved.
6. In order to erase the settings of ALL profiles, general settings and calibration data, go to menu **"Board" — "Erase EEPROM"**.
7. To switch over to the settings of another profile, choose the desired profile from the drop-down list labeled "Profile" (you can find it in the upper right corner of application's window). 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, by RC command, or by operating the menu button on the board. Note that some settings are shared among all profiles. These settings concern hardware component configuration in particular, as well as sensor orientation and configuration, and some others. Once you press **WRITE** button, currently edited profile will be written and becomes active in the board.
8. 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.
9. When finished tuning, you can back-up your configuration several ways:
    - **Saving profile to a file:** all parameters that are visible in the GUI (including hidden extended views), will be saved. But most of calibrations and scripts that are not loaded from the board on connection, do not go there.
    - **Saving EEPROM backup to a file:** it contains all profiles and all calibrations, scripts and so on - the most complete configuration of the system.
    - **Saving EEPROM backup to a cloud:** The same as saving EEPROM backup to a file, but you can access it over Internet. Also some market-available gimbals may have password-protected factory backup, that is impossible to overwrite and corrupt by the user.

3. 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 aligned in parallel with the motors axes. Pay a great attention to mechanical links: they must be rigid and backlash-free! Otherwise it may cause unstable work in real conditions (frame vibrations, wind, etc), make system sensitive to impacts like steps.

2. Calibrating the sensor

Make sure IMU sensor is connected and recognized by the system: arrows on the gauge panel reflects a rotation of a sensor.

Configure sensor orientation by setting **Axis TOP** and **Axis RIGHT** parameters. The fastest way is to use the auto-detection utility: press **AUTO** button and follow the instructions (at first step, level sensor, then tilt it on the right side). More details you can find in the Main IMU sensor section.

Calibrating Gyroscope

The Gyro sensor 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 1 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 Calibration).

> **IMPORTANT:** Before processing any kind of calibration, you need to reset old values by pressing "RESET" button in the "IMU Calibration helper" window!

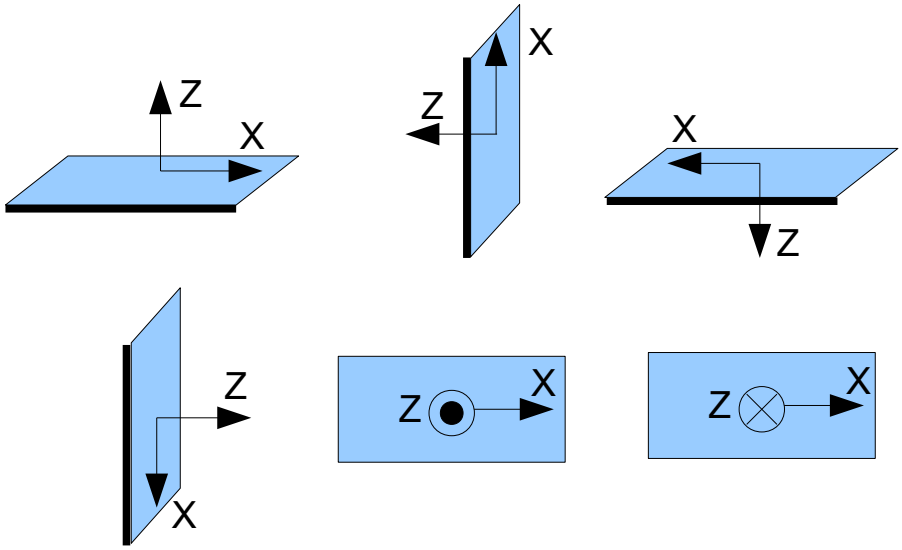
- **Simple calibration mode:** set the sensor horizontally, and press **CALIBRATE.ACC** button in the GUI (or the menu button, if it's assigned to "Calibrate ACC" action). The LED will blink for 2 seconds. Be sure not to allow the sensor to move during calibration.
- **Advanced 6-point mode (recommended):** turn sensor 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 2-3 seconds (until the LED is stops flashing). *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).*

To calibrate second sensor placed on the frame (if present), select it by the toggle buttons "Camera IMU/Frame IMU". All raw sensor data, IMU angles and all calibration commands now relates to selected sensor.

To simplify the 6-point calibration, use the "IMU Calibration helper" tool. It will show a currently selected position and positions that are already calibrated.

NOTE: Precise accelerometer and gyro calibration is a very important for horizon holding during dynamic flying or YAW rotation. it's advised to use a temperature compensation to keep precise operation in a wide range of environmental temperatures (see Temperature Sensor Calibrating).



3. Setting up basic parameters

- For 1- or 2-axis system, disable unused motor outputs in the “Hardware” tab - “Motor outputs” group. It is important to tell system how many active motors are connected, to select proper stabilization algorithm.
- Set **POWER** according to the motor configuration (see recommendations below)
- Connect the main power supply. Motors should start to spin and if all is okay at this moment – will stabilize camera. If motors spin randomly – check that sensor orientation is correct and motor outputs are assigned to proper motors.
- Auto-detect number of poles and motors direction in the "Hardware" – "Motor configuration" – "AUTO". Do not proceed to next step until proper direction is detected! It is normal if number of poles is detected with small error – in such case enter the correct value manually.
- Set parameters "Stabilization settings" – '**Gain multiplier**' to 1.0, '**Outer P**' to 100 for all axes (default values).

- 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, setup and calibrate second (frame) IMU (see Second IMU sensor).

4. Connecting and configuring RC

- Connect any free receiver's channel to the input labeled as "RC_PITCH ", observing the correct polarity

In the RC Settings tab:

- Assign "RC_PITCH - PWM" input to PITCH axis.
- Leave all other axes and CMD channel as "no input".
- For PITCH axis, set **MIN.ANGLE**=-90, **MAX.ANGLE**=90, **ANGLE MODE** is selected, **LPF**=5, **SPEED**=50.
- 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 **SPEED** setting. If stick need to be inverted, select the **INVERSE** checkbox.

If your RC stick have neutral position, you can select the **SPEED MODE** that is commonly used for gimbal control.

Connect and tune remaining axes the same way, as required. You have 5 PWM inputs to assign to all axes

and to the “command” channel.

Analog joystick connection as well as other RC receiver protocols like S-bus, sum-PPM, Spektrum, are discussed below in the "RC settings" section.

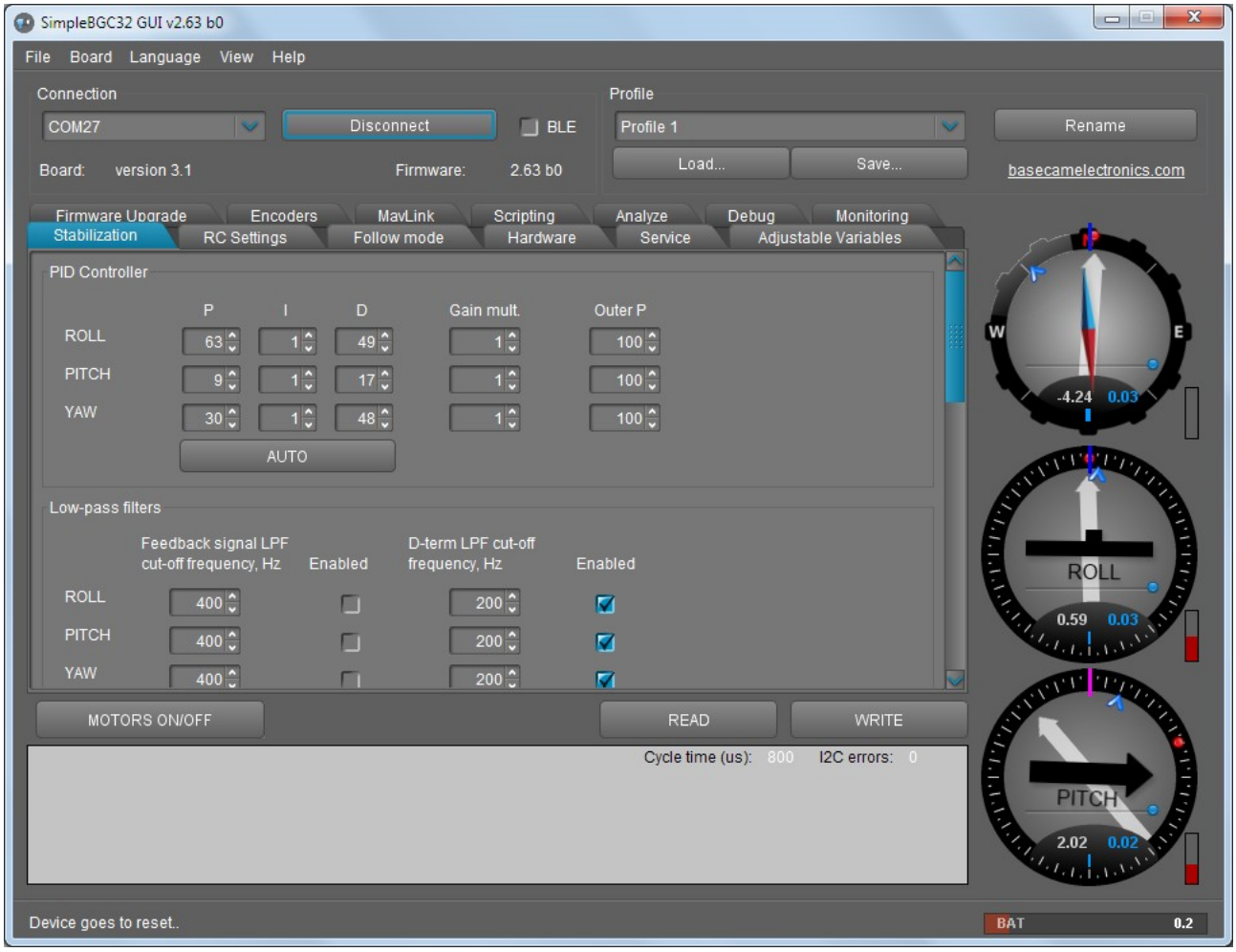
5. Testing gimbal in real conditions

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 on gimbal's mount, balancing propellers, and so on.

> **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.

4. 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':
   - Stabilization – settings related to stabilization algorithms, like PID gains, filters.
   - Hardware — Parameters related to hardware configuration of a system: motor configuration, motor outputs, power level, IMU sensor position and calibrations and so on.
   - RC Settings – settings to control the gimbal roll/pitch/yaw orientation with RC inputs.
   - Service – service functions like menu button assignment, working positions and behavior at startup, battery monitoring, sound.
   - Follow mode – 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.
   - Upgrade – lets you to check the version of firmware and upgrade if necessary.

- Adjustable variables – you can change many system parameters on-the-fly by remote controller or joystick
- Analyze – tool that helps to fine tune system performance by scanning its frequency response.
- Scripting – you can write user scenarios, load to EEPROM and execute by remote command.
- Encoders – if gimbal is equipped by encoders, all parameters and calibrations are set there.

2. Connection — COM-port selection and connection status.
3. Profile — Profile selection, loading, re-naming, and saving.
4. Gauge Panel — graphic visualization of gimbal orientation angles in three axes.
   - *Black arrows display the angles, blue arrows are a 10x time magnification to provide higher precision. Angles in degrees are displayed below by white digits.*
   - *Red marks show target angles that gimbal should keep, in 10x resolution. Normally, red and blue marks should match.*
   - *Grey horisonatl bar below shows a deflection from the setpoint (an error of a position hold), with the blue lines showing the maximum (peak) values, accompanied with the blue digits showng peak amplitude in degrees. Using these numbers, a stabilization quality can be evaluated.*
   - *Vertical red bars on the right side show the actual power level from 0 to 100% of motor's capabilities.*
   - *White arrows shows an angle of each joint (a relative angle of the motor measured by encoder). All arows point vertically in a normal gimbal position (all angles have zero values).*
   - *Blue ball on the horisontal line shows the balance of the payload, I,e, how much force is needed to hold the payload. It may be used to evaluate if payload is balanced well.*
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.

HINT: If not all tabs or settings that are described in this manual, are displayed, check that the view level is set to maximum: **Menu – View – View level**. Also note that the presence of some parameters depends on the hardware and firmware versions.

File menu

- **Save/Load profiles to a file** – duplicates LOAD and SAVE buttons. Operates for currently selected profile.
- **Save profile using template...** – sometimes it's necessary to prevent some parameters to be included in the profile file. For example, you want to share some settings like RC or Follow parameters between different systems, but do not want to overwrite other system-specific settings like PIDs or hardware configuration. You can edit XML file (that profile actually is) manually, removing unwanted parameters, but bad thing is that you have to do it each time you save profile to a file. Alternative is to make it only once, than you can use this edited file as a template. Templates are managed in the "File" – "Settings" dialog, where you can specify a path to a template,

then it will appear in the drop-down menu.

- **Save all profiles to a file** – the same as "Save profile", but saves all profiles to a single file. When it is loaded back, you have to execute "Write all profiles to the board" under the "Board" menu.
- **Settings..** - change settings related to the GUI application functionality.

Board menu

This menu encapsulates options related to controller.

- Read/Write settings - duplicates READ, WRITE buttons
- **Execute action** – execute command in the controller. The list and description of available actions you can find in the "Service" section of this manual.
- **Sensor** – commands related to the IMU sensor calibration
- **Configure bluetooth** - makes an initial setup of the connected bluetooth module. See corresponding section below.
- **Erase EEPROM** - completely reset controller by erasing EEPROM. All settings and calibrations will be lost.
- **Backup manager**

  It allows to make a backup of the EEPROM image, which holds the complete configuration of a system (including all calibrations, scripts, adjustable variables and so on). It is safe to restore EEPROM image made in older versions of firmware, in more recent versions – we maintain back-compatibility. But restoring configuration made in recent version, from the older version, is not possible.

  There are several options available:

  - **Save/Restore to a file.**
  - **Save/Restore to the server**. Requires the Internet connection. You can save up to 3 user backups.
  - **Save/Restore as factory backup.** It's password-protected, that makes it impossible to overwrite or corrupt by the user.
- **Backup/Restore IMU calibration** – backup only calibration of the IMU sensor. You can use this option to transfer calibration between two systems, when IMU was calibrated on one system but is used on another.

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

- **View level** – you can select the level of complexity, that defines which settings will be allowed for editing, and which will be hidden.
- **Choose one of the themes** - 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.
- **Full screen** – go to full screen mode. But in this mode controls will not be resized.

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.

5. Hardware settings

Motor configuration

- **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" and "D" value of PID settings. If you raise the POWER value, you should re-tune your PID values as well.
- **BOOST POWER** - additional power that will be add to the main power in case of error caused by the lost synchronization of electrical and magnetic fields in a motor. It helps to restore sync and return camera to the normal position.
  Note: this parameter is ignored in the encoder firmware.
- **Overall current limit, mA** *(frw. ver. 2.66+ with encoders)*– if this value is set to non-zero, controller tries to estimate an overall current consumed by the motors, and limits it if exceeded. The given limit is automatically distributed between all motors according to their demand. Use this option to protect the battery in a case when it can provide enough current for a single motor, but all motors being fully loaded, cause an overload. A resistance of each motor should be configured to let this function to work. Limitations:
  - The current is estimated according to the resistance of the motors and the sensed voltage of the battery, and may differ from the actual current.
  - Only the current through motors is limited. Current, consumed by the MCU (~100mA) and all connected periphery, is not estimated and is not taken into account.
  - If voltage of the motor driver circuit is not equal to the battery voltage (for example, the "Tiny" controller has an internal DC/DC regulator), the estimated current will differ from actual current.
- **Motor outputs** — you can randomly assign hardware motor outputs for any stabilization axis, or disable output that is not used, for 2- or 1-axis stabilizers. Options are:
  - **ROLL out, PITCH out, YAW out** – motor ports labeled by corresponding names on the main controller
  - **SBGC32_I2C_Drv#1..4 – external motor drivers, integrated directly into motors and controlled by the I2C bus. More info: http://www.basecamelectronics.com/sbgc32_i2c_drv/**
- **INVERT** – reverse motor rotation direction. *It's extremely important to choose the correct motor rotation direction before tuning other parameters!* To determine the correct direction, set the POWER value big enough to rotate the camera without loosing synchronization. Level the camera tray horizontally and click the AUTO button in the "Motor configuration" settings. The gimbal will make small movement to determine its direction. *Gimbal should not have any obstacles that prevent free rotation at this range!* Wait for the calibration procedure to complete for each motor. Finally, the detected number of poles and inversion will be displayed in the GUI.

© Basecamelectronics® 2013-2024

- **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. Count your motor magnets and enter this value if the value is not correct in the GUI.
- **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 **Ultra-high** (~30kHz) that may be selected (but not recommended).
- **Magnetic linkage of a motor** (ver. 2.60+) – this value depends only on **Back-emf constant** of a motor, and does not depends on any other parameters of a system. Configuring it properly helps to achieve better precision of stabilization and better control at high speeds of rotation of a frame or a camera. Note that this compensation is applied only if a speed of a motor is known exactly (from the 2nd IMU or encoders), and POWER is set below its maximum (255) to have a room for compensation.

  AUTOMATIC CALIBRATION:
  - Power ON motors and disable "Follow" mode. Gimbal should be configured and working, PIDs should be tuned. 2nd IMU or encoders should be enabled and configured.
  - Press **AUTO** button and tilt the frame with the moderate speed and amplitude by all axes during 15 seconds (1-2 tilts per second with 20-40° amplitude). LED will flash and "calibration" sound will be emitted.
  - On complete, new values will be loaded into fields. Check that system operates without problems. If precision of stabilization decreases or system becomes unstable, most probably estimated values are too high: set values to 0 and repeat calibration.

  MANUAL CALIBRATION:
  - Set PID gains low temporarily. When you rotate the frame by hands quickly - camera does not have enough reaction to compensate it and rotates a little in the same direction. But when you find the correct magnetic link value - it stops to rotate in that direction and just wobbles around the target position. Start from value 10 and increase it until you get the lowest affection of the frame movements to the camera movement.

- **R, Ohm** *(frw. ver. 2.66+)* – resistance of one phase of motor's winding (a resistance measured between any two terminals divided by 2). Is used by the current limiter function.
- **Synchronize motors** *(frw.ver. 2.70b9)* – this utility will be useful to synchronize electrical field and direction of two motors connected in parallel and working on a single axis. The controller will apply constant power to the selected output at an arbitrary electrical angle, so both motors will align to this angle and should be mechanically connected there. There is an optional "rotate" checkbox which causes electrical field to be slowly rotated, to ensure that the direction in both motor matches as well. During the synchronization motors should rotate freely; ensure that the level of applied power is enougth to overcome friction and other forces, and is safe to not overheat the motors. There is a "timeout" parameter which defines how long to apply power.

Gimbal mechanics configuration

- **Order of hardware axes** – for non-standard gimbals, you have to define the order of motors counting from the camera. Default is "Camera – PITCH – ROLL -YAW". Currently supported additional configurations:
  - Camera-YAW-ROLL-PITCH

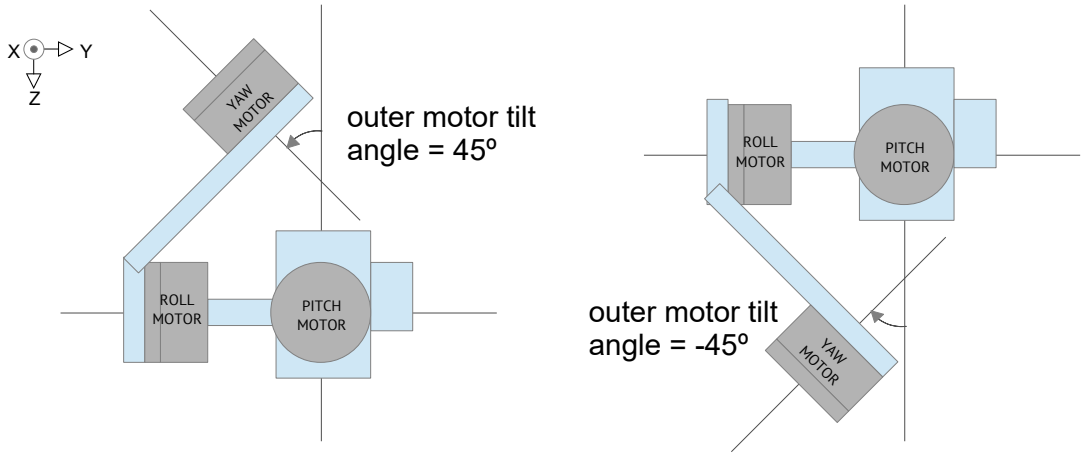
- Camera-ROLL-YAW-PITCH (limited series of controllers)
- Camera-ROLL-PITCH-YAW

- **Outer motor tilt angle** *(2.63b0+)* - If the shaft of an outer motor (YAW in regular configuration) is not orthogonal to the shaft of a middle motor (ROLL in regular configuration), set the angle of deflection by this parameter.

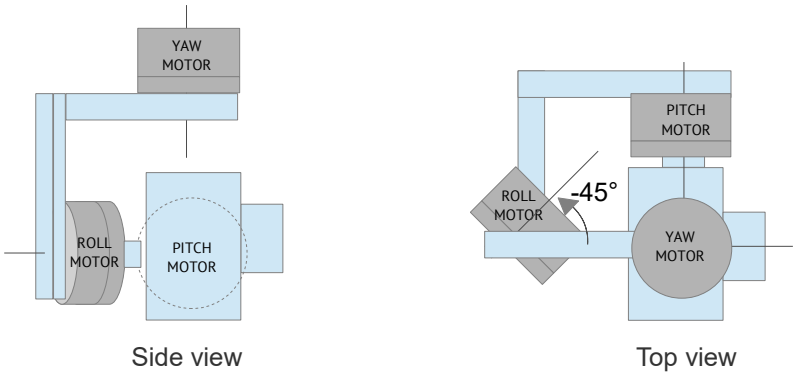
  Direction: when observing from the inner motors' "+" axis, counter-clockwise is positive. The positive direction of axes: X – right (or East), Y – forward (or North), Z – down, forming END coordinate system.

  NOTE: The proper "home position" where encoders offset should be calibrated, is shown on the picture below. General rule: having camera as an origin, firstly align the axis of the inner motor to one of X,Y,Z, then align middle motor, then outer motor.

- **Middle motor tilt angle** *(2.67b1+)* - similar to the "outer motor tilt angle", defines the deflection from a normal middle motors' axis. May be combined with the "Outer motor tilt angle".
  Direction: when observing from the outer motors' "+" axis, counter-clockwise is positive.



Outer motor tilt angle and home position for Cam-Pitch-Roll-Yaw configuration.
In this example, the inner motor is PITCH with the corresponding X axis "+" pointed towards the observer.



Middle motor tilt angle and home position for Cam-Pitch-Roll-Yaw configuration.
In this example, the outer motor is YAW with the corresponding Z axis "+" pointed away from the observer.

IMU sensor settings

- **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).

- **Gyro dead band** - helps to cut off gyro noise around zero (that may be audible as 'white noise' in heavy setups), and to make system more immune to self-excitation.
- **ACC low-pass filter, Hz** - defines the cut-off frequency of 2nd order low-pass filter, applied to the accelerometer data before it used as an attitude reference. It removes the disturbances caused by the short movements with the lateral accelerations. Without such filter, these accelerations affects the attitude vector, causing unwanted errors in the IMU angles, especially with the low "Gyro trust" values. Setting the cut-off frequency a bit lower than the possible rate of accelerations during normal use of a gimbal, can help to minimize their negative affection. The trade-off is a delay, introduced to the accelerometer data, that increases the time of transient processes in the sensor fusion algorithm.
  Recommended value is 0.1 – 0.5 Hz. To disable the filter completely, set this parameter to 0.
- **Misalignment correction –** angles of deflection of actual axes of a sensor from ideal axes (that match motor shafts in neutral position). You can detect misalignment in the placement of the main IMU sensor automatically. To do that:
  1. Turn motors OFF;
  2. Firmly fix the joint between the inner motor (that connects to a camera) and the middle motor (next in order). Now the camera platform has only one freedom of rotation – over the inner motor's shaft, that is fixed in space;
  3. Press **AUTO** button. You have 10 seconds to rotate the camera by hands in both directions several times (make 5-10 rotations to ±45 degrees roughly). On complete, new values will be applied and displayed in the GUI.
- **I2C high speed** – use 800 kHz communication speed over I2C bus. It may give positive effect by reducing delay between gyro signal reading and correction applied, or in case if there are many devices are connected to I2C bus. But it will increase a chance to get the I2C errors, so use this option with care. *Warning: AS5048B encoders do not support high speed I2C.*

IMU Calibration

Basic calibrations are described in the section Calibrating the sensor. Advanced calibrations are described in section Temperature Sensor Calibration. Also controller can calibrate gyroscope on each power-on sequence, according to the parameter **"Automatic gyro calibration":**

- **Calibrate at system start** – always do a gyroscope calibration at power on. Gimbal should be placed on rigid surface and leaved without motion (even vibrations are not allowed!) for several seconds.
- **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).
- **Try to calibrate or use previous values** – with this option, IMU sensor will be calibrated at the system startup, if no motion is detected. If motion is above threshold (you hold gimbal in hands at startup), calibration will be interrupted and previously saved values will be used.

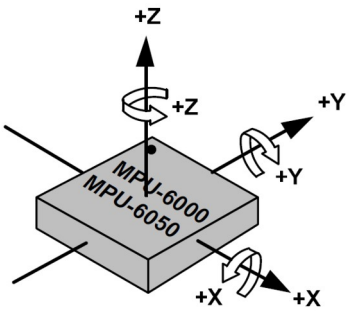
Frame IMU – configure 2nd IMU sensor

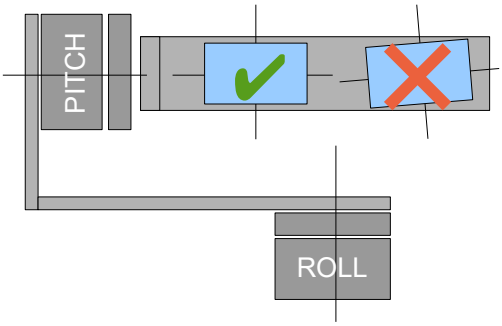
- **Mounting position** – set the location of the frame IMU. See Second IMU sensor section of this manual.
- **Swap frame and main sensors** – swap the roles of IMU sensors.
- **Estimate frame angles from motors** - this uses the motor's magnetic field for rough estimation of frame tilting, and helps 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 **Home position Offset** parameter in the "Follow mode" tab. Like with the Follow mode, it's 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, or use encoders, because the data from these sources is more precise than from motors.
- **Use gyro signal as feed-forward** – if enabled, signal from 2nd IMU gyroscope is passed to motors as feed-forward signal to improve the precision of stabilization. It's enabled by default and gives noticeable advantage in hand-held usage, but can make things worse, when the high level of vibrations impacts 2nd IMU sensor, if it's located on the frame of UAVs (multirotors, planes).
- **LPF frequency, Hz** – low-pass filter that is applied to filter out unwanted noise and vibrations before applying feed-forward signal from the frame IMU. Default value is 10Hz. Setting it too low is not recommended because phase delay will be bigger and the result of such correction will be not precise.

Main IMU sensor

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, by specifying axes direction in the "Top" and "Right" dropboxes, or using **AUTO** button to find proper direction automatically in 3 simple steps. 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.



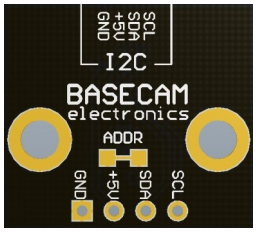


Supported sensor models:

- *Invesense MPU6050* - not expensive MEMS sensor with precise 3-axis gyroscope and 3-axis accelerometer. Has poor temperature stability.

- *Invesense ICM20608* – slightly better signal/noise ratio compared to MPU6050; wider bandwidth (you may need to adjust LPF filter to match it to MPU6050 to keep the same PID settings); separate LPF filter for accelerometer makes it more immune to vibrations; better temperature stability.
- *Basecam CAN_IMU (Based on ICM20608)* – this device encapsulates separate MCU that reads IMU at high rate, apply additional filtering and send data via CAN bus protocol to the main controller.

Second IMU sensor



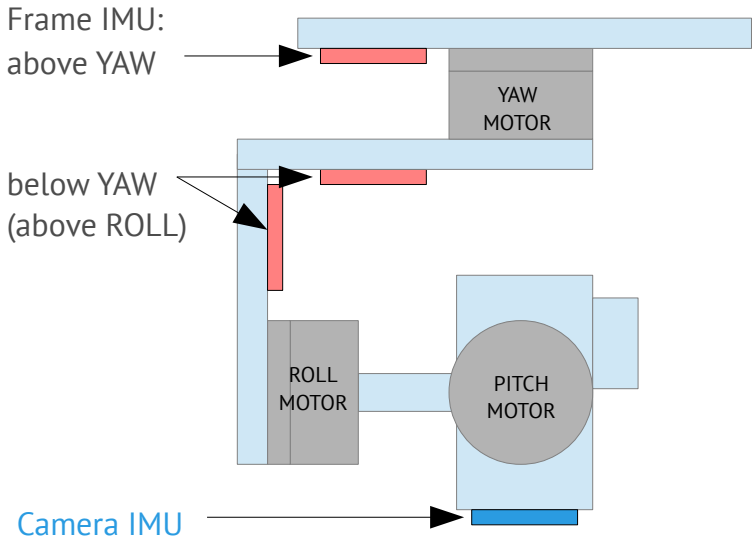
There is an option to install the second IMU sensor on the gimbal's frame. The advantage is more precise stabilization (you may use lower PID's to get the same quality) and knowing frame tilting greatly helps 3-axis systems to extend the range of working angles.

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 original Basecam IMU sensor, address 0x69 may be set by **cutting the ADDR bridge**, located on the back side of the sensor.

CAN_IMU can be used as 2nd IMU as well – it has soldered jumper to choose "Frame" or "Main" role.

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/outer motor**, it helps to stabilize ROLL, PITCH and YAW motors in non-encoder system*. 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).

*In the encoder-enabled firmware, the 2nd IMU does not improve the stabilization quality, but still can be

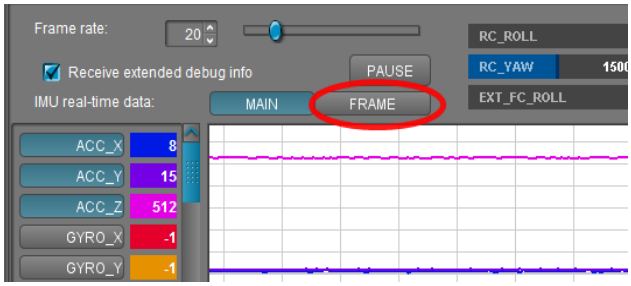
helpful in some cases if mounted in the position **below YAW/outer motor**. In this position it's used to properly handle a "gimbal lock" condition, when the outer motor becomes parallel to the inner motor. Also, for systems built with geared motors, it helps to detect the startup period of the encoder (more information about this use case is provided in the "SimpleBGC32 Encoders manual").

There is a particular option "**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 for the YAW motor. In some cases this can give better result than the main IMU, because mechanically "IMU-to-Motor" link is shorter and more stiff, which is good for a closed-loop operation.

Like the main (camera) IMU, the frame IMU may be mounted in one of six orientations, with the axes parallel to the motor's axes in a normal position.

Configuring the frame IMU

To configure the frame IMU, first of all set its mounting position in the "**Hardware**" tab, "Frame IMU sensor" group. Write settings to the board and go to the "Monitoring" tab. Press the button "**Frame IMU**":



If the second IMU is connected properly, this button becomes active. After pressing on it, all IMU-related realtime data corresponds to the frame IMU. You may notice the right panels with arrows are displaying now angles not for the main, but rather for 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. But for the second IMU, precise calibration is not so crucial, as for the main IMU.

Precision of angle measurement

A MEMS gyroscope-based IMU gives very good precision, especially compared to single accelerometer. But it still can be affected by environment, that can reduce the precision and give negative effects like lost horizon, slowly drifting angles, cross-axis interference (rotation by one axis lead to declination by other axis). Below are the most common reasons and our recommendations how to solve them:

- **Vibrations**: try to isolate gimbal from vibrating platform by dampeners.
- **Lateral or centrifugal accelerations** (fast accelerated slides or movement by a curved trajectory): consider "Gyro trust" setting.
- **Wrong calibration** of accelerometer or gyroscope: carefully follow our instructions and check the validity of calibration from time-to-time.
- **Misalignment** of sensor's axes and gimalbal motor's axes: pay attention to sensor orientation when mounting sensor on the gimbal, or apply a correction by "Misalignment correction" parameters later.
- **Changes in temperature** than affect calibrations: do the temperature calibration
- **Drift of heading angle** without good reference: install and configure a magnetometer sensor.
- **Over-saturation of gyro sensor**: prevent rotations faster than 2000 degree/second.

The problem of mutual azimuth drifts of two IMU sensors

Gradual drift of angles taken from Gyro is a normal situation, and you need to take into account it in any AHRS (attitude and heading reference systems). Additional sensors can be used to correct gyro drift: an accelerometer and magnetometer.
An accelerometer corrects 2 axes of a gyro by gravitation vector.
A magnetometer corrects 3rd axis by Earth's magnetic field vector.

Complete IMU generally includes 3 sensors (called 9-axis system). Using a magnetometer in gimbals is not very common since the precision of a magnetometer highly depends on the environment and it is difficult to calibrate it properly. Fortunately, in the most cases of gimbals usage, the absolute precision of the azimuth detection is not required. But using two IMUs (first installed on the camera tray, and second installed on the frame of a gimbal), the azimuth of one sensor has to match the azimuth of another sensor. In the SimpleBGC32 controller, special algorithms are used to correct mutual azimuth drift. It allows the system to work stably in almost any conditions.

The following are methods which are automatically applied by the controller to correct absolute drift and mutual azimuth drift of both sensors:

- **The limits caused by a gimbal's design**. For example, if the second sensor is installed below YAW, its azimuth in normal position always match the azimuth of the first sensor. But when the frame inclined forward at 90 degrees, this condition is wrong and other methods should be used.
- **Detecting rotation of motors by the electric field**. If the second sensor is installed above YAW, its azimuth may not match the azimuth of the first sensor. But if the rotation angle of YAW motor is known, it is possible to match their azimuths. In the different orders of hardware axes, for example, Cam-YAW-ROLL-PITCH, this situation appears in any position of the second sensor. Note that this correction works if motors are switched on, and system was started in "normal position" when azimuths of both sensors were matched (though additional algorithms are used to synchronize azimuths, it's better to always take care about proper start position).
- **Detecting rotation of motors by encoders.** Using encoders (at least one installed on YAW axis) significantly improves the precision of correction.
- **Using magnetometer.** If a magnetometer is connected to the IMU sensor (frame or camera) then its azimuth will match True North. The second sensor will be automatically corrected by the magnetometer by one of the above methods.
- **Using precise orientation data from an external AHRS system**. Using Serial API, you can provide the precise orientation of the camera tray or a frame measured by an external system with high-grade IMU using command "CMD_AHRS_HELPER". In this case, an appropriate sensor will be corrected using this data, and the second sensor will be corrected by one of the above methods.
- **Using AHRS data from flight controller** - you can connect UAV autopilot (for example, Ardupilot or Naza) to the SimpleBGC32 controller by MavLink protocol, to synchronize their attitudes.
- **Using external high-quality IMU sensor** – this is the most preferred option. SimpleBGC32 supports several models of high-quality IMU sensors, described in details in the section Using an external IMU sensor to improve the precision of stabilization

There are also a number of methods to manually correct gyro drift:

- **Providing the heading angle from an external source.** Via adjustable variable "FRAME_HEADING_ANGLE" you can provide the heading angle to the controller. It will be translated to the main IMU, if possible, and used as heading reference. The possible case where it may be used: gimbal is mount statically, so the frame angle does not change. Or, gimbal is mounted on a crane, and its azimuth is known from it's controller.
  ** This method is similar to CMD_AHRS_HELPER, but the reference vector is computed automatically from single variable taking into account frame's attitude.*

- **Correcting gyro offset manually**. If operator can observe a picture from a camera, it can detect a gyro drift direction and apply correction by adjusting a knob on a remote controller. It can be linked to the "GYRO_HEADING_CORRECTION" adjustable variable.

Advanced IMU calibrations

The "Extended" and "Pro" versions of the SBGC32 controller support advanced calibration of the main IMU sensor. You can calibrate scale, bias, non-orthogonality and alignment for the accelerometer and gyroscope, 21 parameter in total. These values can be set in the "Calibration helper" tool, "Advanced" tab. You can enter values manually, load from the sensor's EEPROM (if present) or calibrate it using the GUI tool. There are several methods available:

Method 1: Using the high-quality external IMU sensor as a reference.

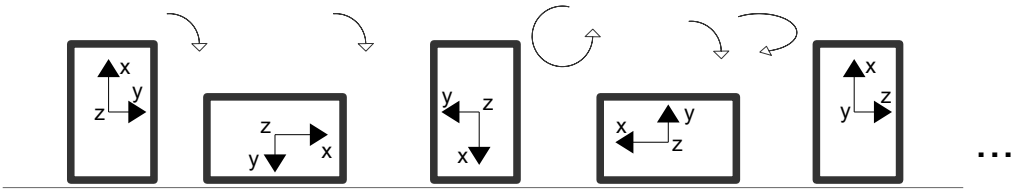
It calibrates the gyroscope only.

1) Install the external IMU sensor on the camera platform and configure it as described in the section 18. For big gimbals, it might be better detaching camera platform or using a special temporarily plate allowing to mount and align both IMUs precisely.
2) Turn motors OFF – it's better to make this calibration manually.
3) Press the **"Calibrate by Ext.IMU reference"** button, then randomly rotate the camera platform along all axes. Do the rotations varying speed and direction, but not too sharp.
4) When calibration is finished, new values will be displayed. Write them by pressing the **"WRITE"** button and restart system to apply changes.

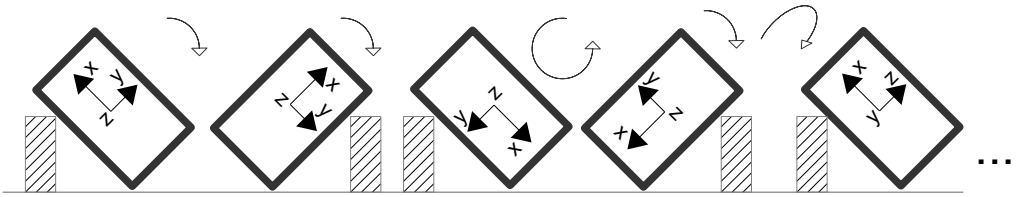
Method2: Multi-point calibration

It calibrates all 21 parameters of the gyroscope and accelerometer. It does not require any special equipment rather then a heavy bar where you should firmly mount the IMU sensor, and a flat surface where the bar should be placed and stay steady on its faces. This method collects data in multiple static positions to calibrate accelerometer, and analyzes rotations between them to calibrate gyroscope.

1) Place the bar with the sensor attached to it, to the first static position.
2) Press the **"Multipoint calibration"** button. The progress dialog with the red circle appears.
3) Wait about 3 seconds to let system initialize and collect data in 1st position.
4) When the circle becomes green and "click" sound is emmited, quickly rotate the bar to the next static position (put it on different face) and leave steady there for ~2 seconds, while the circle is red. Try to make transition in a form of direct rotation by the shortest path; change the direction of rotations and the axis of rotations;
5) Repeat step 3) and 4) 10-15 times, to collect static positions for all six faces of the bar and rotations around all three axes in both directions.
6) Collect another 10-15 points, but now keep the bar inclined, as shown in the picture:



Series 1 (10-15 static points)



Series 2 (10-15 static points)

7) You can finish calibration at any moment by pressing the **"FINISH"** button, or wait until the progress indicator goes to 100%.

8) The new values will be displayed in the fields. Write them by pressing the **"WRITE"** button and restart system to apply changes.

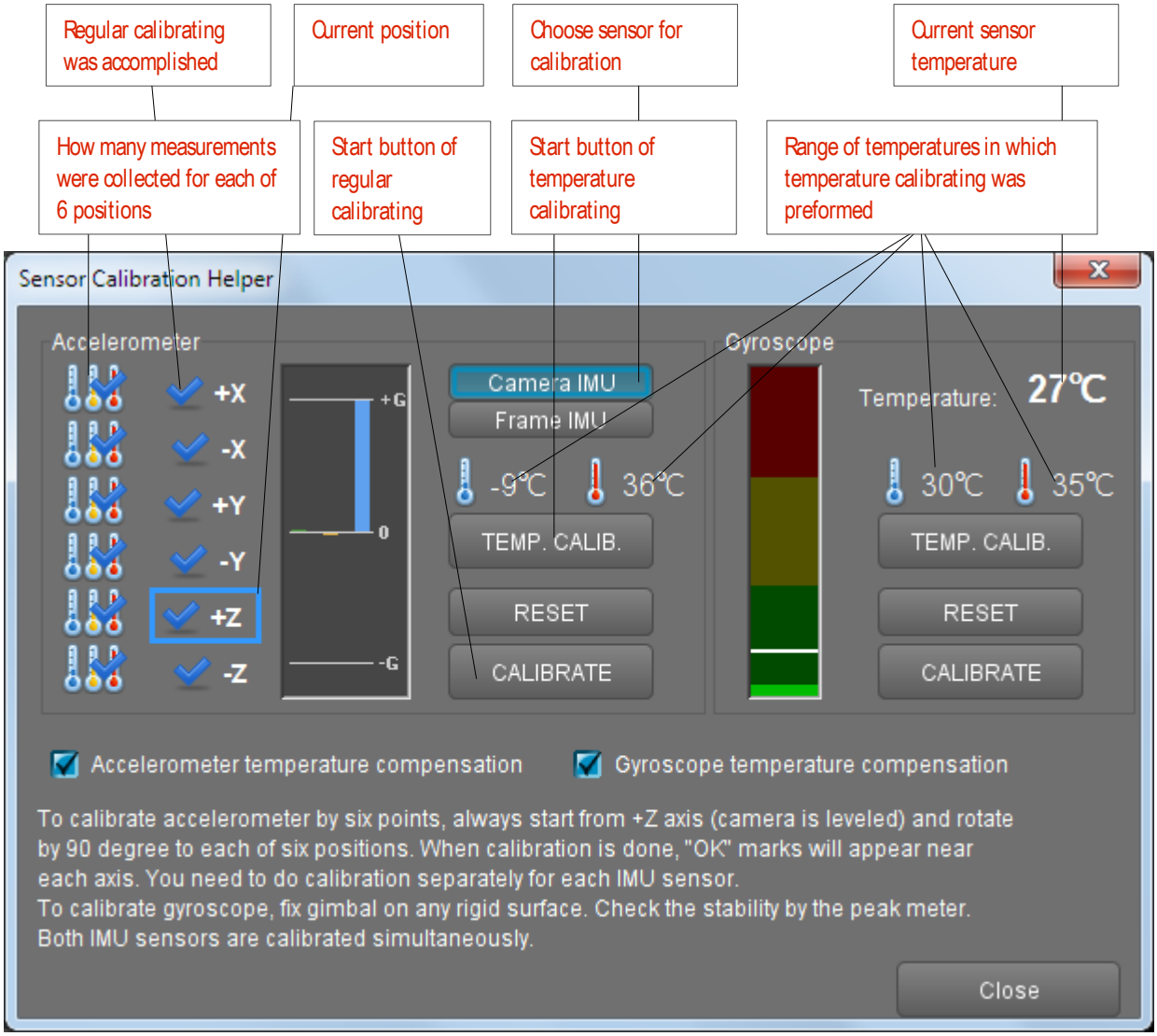
There are no requirements to make faces perfectly leveled or 90 degree -aligned each other. The only requirement is to keep sensor in static phase absolutely steady, w/out any motion, vibration or bounce! Second important requirement is to collect data that uniformaly covers all axes of accelerometer and gyroscope.

Temperature Sensor Calibration

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 the whole PCB have the same temperature. In order to ensure this condition the sensor should be encapsulated in an enclosure having significant mass (to provide thermal inertia), placed into an insulating shell cut out of a piece of plastic foam (EPP foam used for packaging is okay). 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.

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 "IMU calibration helper" tool.

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, 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: Starting from firmware version 2.56, regular calibration by 6 points does not disable the temperature calibration, but updates it to match the actual values at the current temperature. So, while the temperature compensation is being always active, you can from time to time make a regular calibration to improve its precision.

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 the 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 slowly. Calibration continues as long as temperature increases. **Ensure total immobility of the sensors during whole calibration process!**
5. As soon as the temperature stops rising, 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 raw data in the "Monitoring" tab when totally immobile equals

to zero within the whole temperature range applied during calibration; drifting of axis arrows is absent or very low.

NOTE: Starting from firmware version 2.56, the regular gyro calibration does not disable the temperature calibration, but updates it to match the actual values at the current temperature. So, while the temperature compensation is being always active, you can sometimes make a regular calibration to improve its precision.

If gyroscope calibration at system start is enabled, it will refine the temperature compensation, but not save it to the EEPROM memory.

Serial port settings

- **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.
  Note: this setting is applied only for UART1. Other ports are hardly configured to 115200 8N1.
- **Enable UART2** (ver. 2.60+) - this option allows to send SBGC Serial API commands to the UART2 port, or use it for MavLink connection. Port settings are 115200, 8N1 (8bit, 1 stop bit, no parity). The pin-out depends on the board version and may be shared with the other functions:
  - *"Regular", "Tiny"*: Rx is shared with AUX3, Tx is not connected;
  - *"Extended"*: Rx is shared with SPI MISO, Tx is shared with SPI SCK;
  - *"BaseCamBGC Pro"*: dedicated UART2 port
- **Swap RC_SERIAL <-> UART2 ports** (ver. 2.60+) - this option swaps RC_SERIAL and UART2 ports. It may be used to assign RC_SERIAL port functionality (the only port where Spektrum and S-bus protocols are supported) to different pins, if the original pins are occupied by the other functions. RC_SERIAL pin-out depends on the board version:
  - *"Regular", "Tiny", "Extended"*: Rx is shared with RC_ROLL, Tx is shared with RC_YAW;
  - *"BaseCamBGC Pro"*: Rx is marked as "S-bus", Tx is shared with AUX3;

6. Stabilization settings

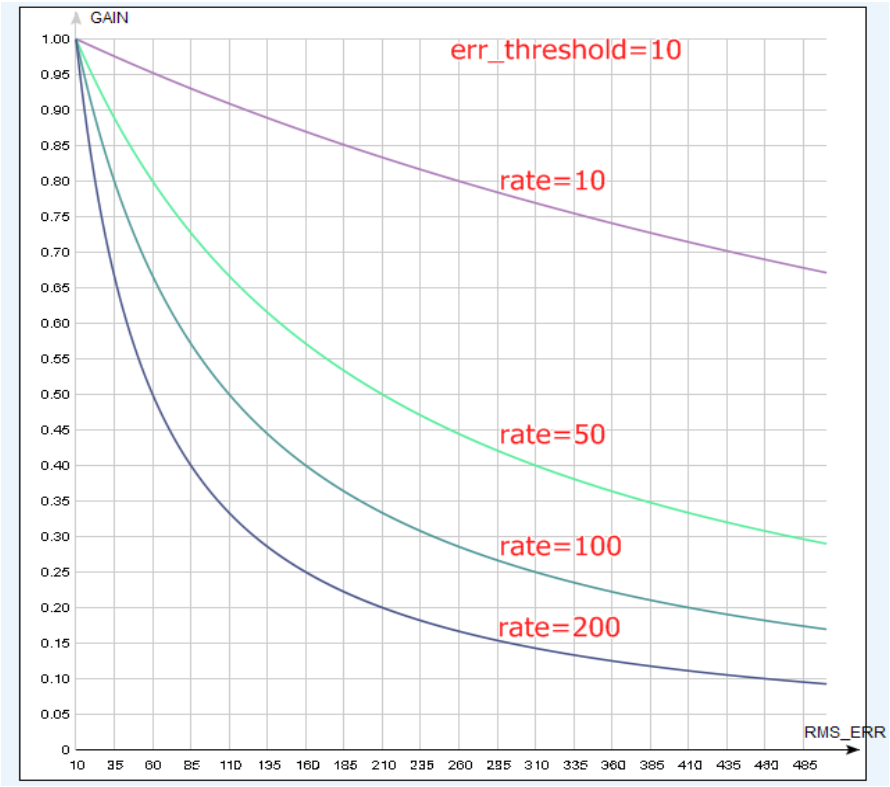
PID settings

- **P,I,D – PID gain 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.*
- **Outer P** – proportional coefficient for outer controller, that defines how fast angle error is corrected. The bigger value, the faster camera returns to normal position after big declination. Default value is 100. Normally, there is no reason to change it.
- **Gain multiplier** – additional multiplier for P,I,D coefficients. Changing this value allows to extend a range of PID settings for setups where normal range is not enough. Also it allows to adjust PID gains on-line during gimbal operation, by mean of RC knob or analog potentiometer linked to the **PID_GAIN_X** adjustable variables. It gives to user a simple way to find a balance between precision of stabilization and stability of PID loop without connecting to the GUI and adjust PID setting separately.
  Note: Before 2.60 firmware, it was called 'Gyro high sensitivity' and if enabled, acted as 2.0 gain multiplier
- **AUTO** - show dialog window to set-up and start automatic PID tuning. Detailed description is given below in section PID auto-tuning.

Adaptive 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:

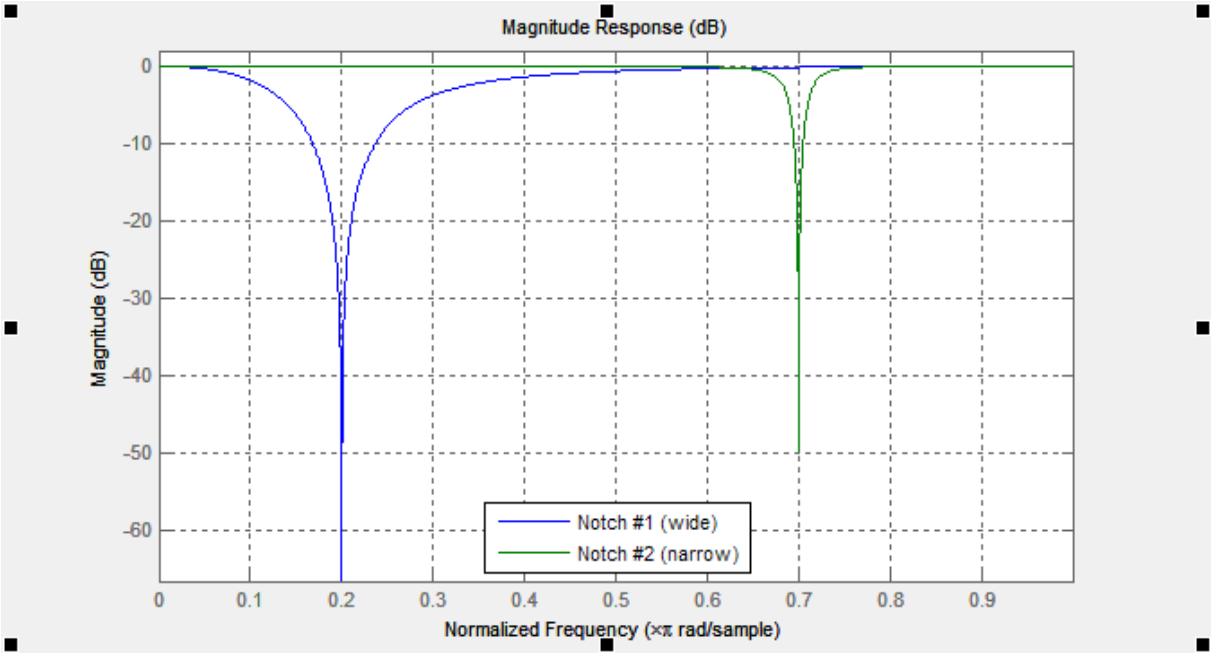


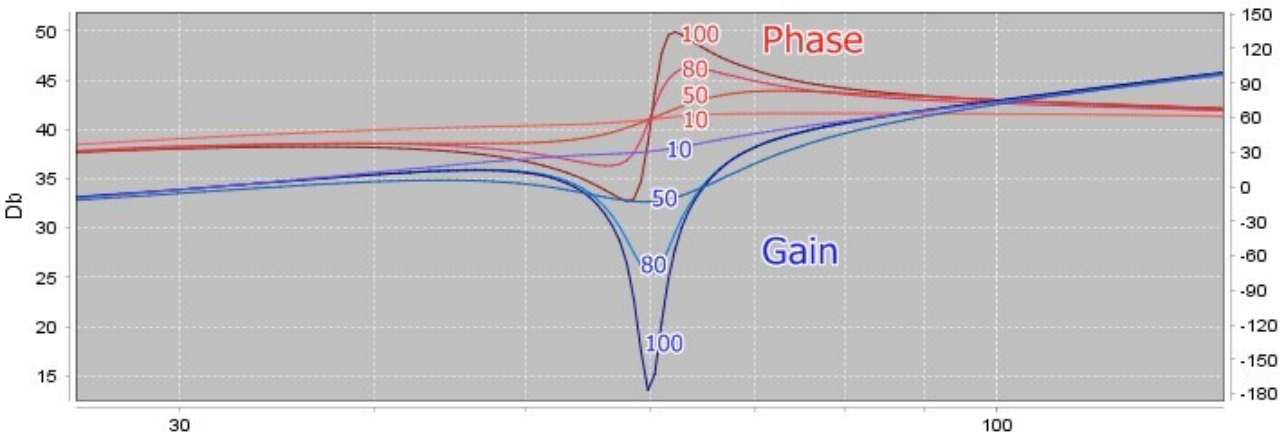
- ***Recovery factor***, *0..10 - defines, how fast PID gains are recovered back when the system becomes stable. Too low of a value may increase a chance that vibration comes back in a short time. Too high of a value may cause worsen of operation (because lowered PID values are kept longer). Recommended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

Digital Filters

Filters can greatly improve the quality of PID controller operation in some cases.

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 100dB to get maximum effect, set it <20dB to compensate only light resonances.

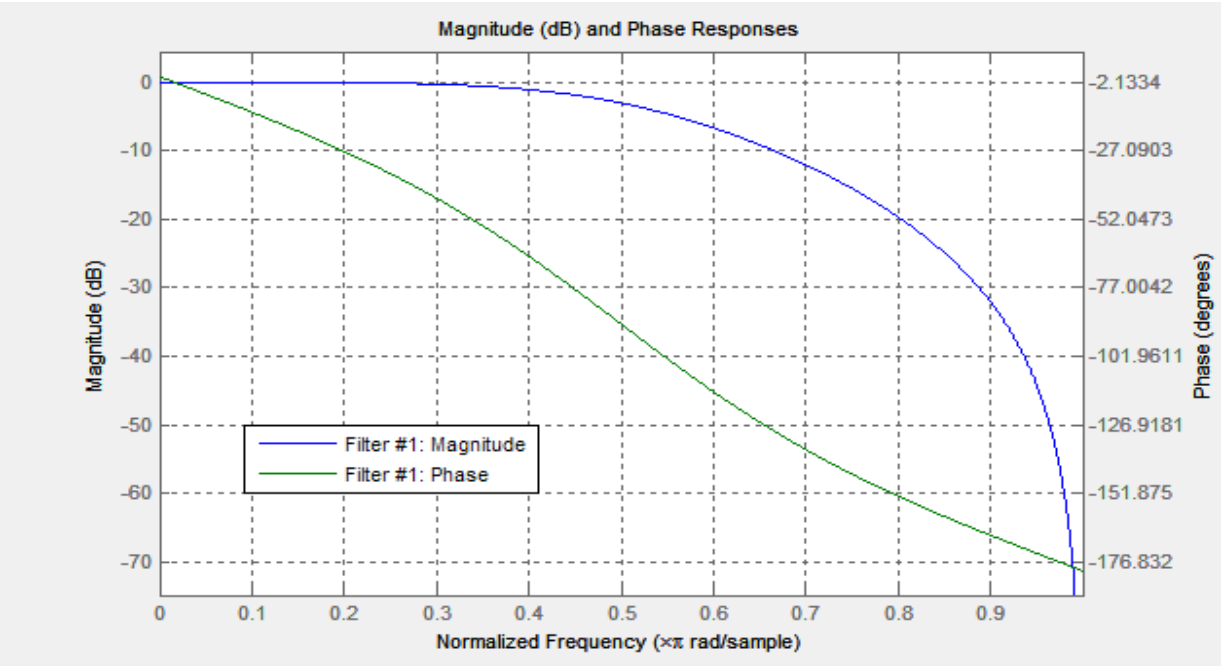
Example: gimbal behavior is stable, but when camera tilts downward 60 degrees a strong vibration occurs and this effectively prevents an increase gain of PID (which might otherwise be desired to improve traction).

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.
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 check-box "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).

Another very effective way to detect amplitude and frequency of resonances in a system, is the System Analysis Tool described below in this manual. Testing mechanical system response (called "plant" in this tool), you will receive clear information about all resonances: center frequency, width and amplitude in dB. Making test "plant+filters", you can see, how effectively notch filters compensate resonances.

Starting from version 2.60 firmware, you can turn a notch filter into a **peak** filter, that in opposite, amplifies



©

the narrow band of frequencies. To configure filter such way, set negative value to the "gain" parameter.

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.*

Axis to stabilize for 1- or 2-axes systems

This group of settings allows to assign a stabilization axis (that corresponds to Euler angle rotation) to a particular motor. By default in 1- or 2- axes systems, each motor stabilizes the matching Euler axis in normal position. But you can re-assign motor to stabilize different axis.

Example: 1-axis gimbal is used to stabilize the ROLL angle of a camera as its main application. But you may want also to use it to rotate camera by the YAW axis, to shoot 360° panorama or time-lapse. Controller can be configured to switch stabilized axis automatically, or switch by profile basis – just assign different axes to a motor in different profiles.

> **NOTE:** The term "axis" can have different meaning: physical motors or Euler axes that controller is aimed to stabilize. In our system the labels ROLL, PITCH, YAW in the sections like "Stabilization settings", "Hardware settings" relate to motors (that matches to stabilized axes only in normal position), and in the sections like "RC settings", "Follow mode" relate to corresponding Euler axes.

7. PID auto-tuning

Auto tuning aims to tune P, I, D parameters the most efficient way. Through the use of powerful algorithms from control theory, the result of auto-tuning overcomes the result of manual tuning in most cases. It's a recommended way to get maximum performance from your system.

Before you start PID auto-tuning, make sure that the following parameters of your system are configured correctly: a camera is balanced, IMU sensor is set up correctly (position is configured, and accelerometer and gyroscope are calibrated), motor outputs, "Power", "Inverse" and "Number of poles" are set up (last 2 settings can be configured automatically, as described in the user manual). Any further changes of these parameters can affect the PID controller functionality, and you will have to configure it again.
Check that **PID gain multiplier** is set correctly (default is 1.0). Set the parameters **P**, **I**, **D**, so that a gimbal will remain steady and vibration free. For example, P=10, I=0.1, D=10..20. Currently, it does not matter how good the gimbal holds its position. Set a camera horizontally and make sure that it can rotate through 20-30 degrees on all its axes.

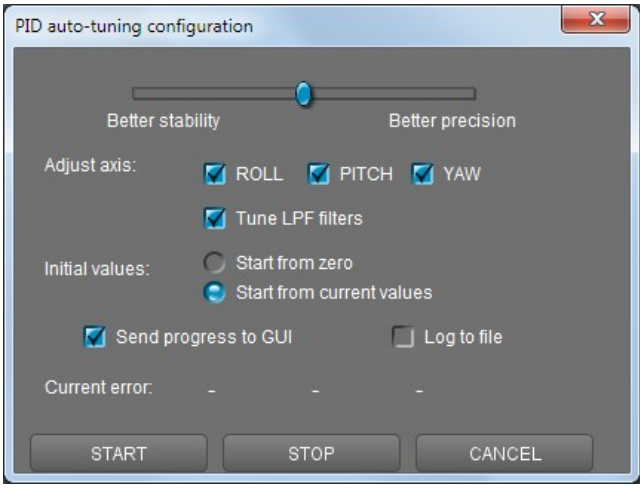
There are several options to automatically tune PID and relative parameters: running algorithm in the controller, or tuning parameter using GUI tool. The second option is introduced in the GUI version 2.68b7 and is described in the section **"Automatic PID controller parameters tuning based on the system response in a frequency domain"**.

Running auto-tune algorithm in the controller

Plug-in a battery, connect board to the GUI and press the **"Auto"** button in the PID parameters section. A dialogue window will be displayed, where you can modify some parameters affecting the algorithm.

Version1 of PID auto-tune algorithm

For the firmware versions before 2.73



The slider at the top defines the target of tuning. If it is closer to **"Better precision",** the auto-tuning program will try to achieve maximum gain and keep it. If the slider is closer to **"Better stability"**, the auto-tuning will try to find moderate gains where system is more stable.

You can configure all axes together or each axis separately. Tuning axes separately can give better result in some cases.

If you want to use your current settings as a start point, select **"Start from current values".** Otherwise the auto-tuning process will start from zero values. In this case the auto-tuning program will perform an additional test for each axis to detect initial parameters.

Starting from the 2.60 firmware, the auto-tuning program can automatically set up **Low-pass filter** frequency. This filter can significantly improve stabilization quality in systems which have high-frequency resonances.
Usually, the bigger gimbal and the heavier camera are, the more efficient *Low-pass filter* works. If a system

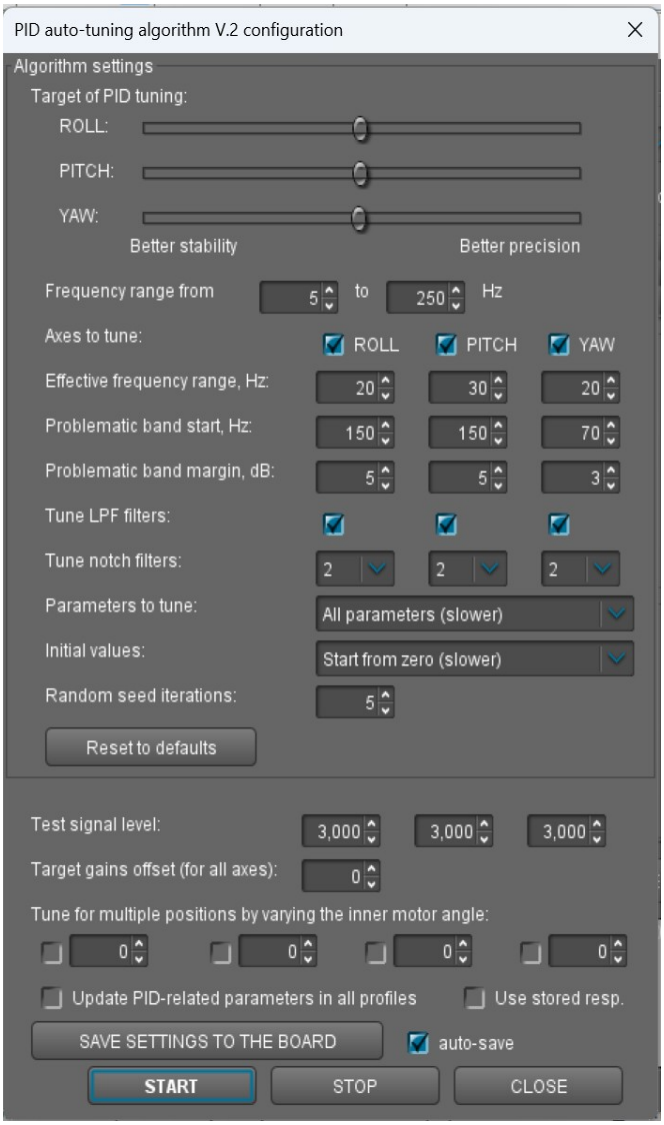
has significant resonance, *Notch filters* can be used as an alternative to Low-pass filter. (see the manual, chapter "Digital filters").

Select **"Send progress to GUI"** checkbox to see how PID values change in real-time during the auto-tuning process. Select **"Log to file"** to write PID values together with some debug variables to the file "auto_pid_log.csv." It can be analysed later to understand system behaviour better. There are many tools to plot data from log files, for example http://kst-plot.kde.org.

Version2 of PID auto-tune algorithm

For the firmware version 2.73 and later, "Extended" and "Plus" family of the controllers

This algorithm replicates the "Optimizer" tool in the GUI's "Analyze" tab, with some simplifications as it runs on limited resources. All parameters of the algorithm are described in the corresponding section Algorithm parameters. Below are parameters that are unique for the on-board version:



Test signal level: the strength of a stimulus signal emitted to capture the system response. See Capturing the system response section, describing how to chose the proper stimulus strength.

Target gain offset: it will be added to the value specified in "target of PID tuning" parameter. It appears in the system as an adjustable variable, which allows to adjust the algorithm's target in runtime.

Tune for multiple positions by varying the inner motor angle: normally, the system dynamics may change seriously when the gimbal configuration is changed because of rotation of the inner and middle motors. This algorithm can collect system responses in multiple positions and take them all into account, providing a result that will be stable across the whole working range. You can specify up to four positions by providing angles for the inner motor and enabling them. If only one position is enabled, gimbal will move to this position. If none is enabled, it will start tuning in the current position.

Update PID-related parameters in all profiles: works exactly as it sounds. All 5 profiles will be updated.

SAVE SETTINGS TO THE BOARD: this button doesn't start PID tuning but saves the configuration to the EEPROM.

Auto-save: every time "START" is pressed, the configuration is saved to the EEPROM.

START: starts auto tuning. It can be also started by the menu commands "**PID all parameters auto tune**" and

"**PID gain auto tune**".

Start without the GUI connection

With both algorithms, it's possible to start the auto-tuning process without using GUI. You need to assign the appropriate command to a menu button or RC "CMD" channel. This command can be used for fine tuning a system when changing a camera or a lens. The auto-tuning process will use parameters that were configured and saved to EEPROM.

Hints on tuning:

- During the auto-tuning process hold the gimbal so as it will be used. It's okay to hold it in hands for a hand-held gimbal
- If the tuned system is stable in a horizontal position but is not stable when the camera is tilted, try to repeat the auto-tuning in the position of maximum instability. Also, you can manually decrease the "Gain multiplier" parameter to keep the system stable in a whole working range.
    - For the Version2 of algorithm, you can specify multiple positions and it will find PID values suitable for all that positions.

Automatic tuning using "Analyze" tool in the GUI

The "Analyze" tool allows to evaluate the performance of stabilization. Also starting from firmware ver. 2.68b7, we introduce a powerful auto-tuning algorithm that allows choosing optimal parameters of PID controller by the minimization of a *sensitivity function* which is obtained by analysing system response in a frequency domain.

This algorithm allows the developer to do a primary gimbal setup. Also, it can be launched directly from the controller (it will be implemented in firmware 2.70 and above), which allows to quickly optimize parameters of PID controller during the gimbal normal usage, that's very convenient for non-advanced users. As well it can be launched automatically each time system starts up, to adapt to the payload (installed camera).

Tuning parameters for all possible positions and different payload

Since the dynamic characteristics of the system are significantly dependent on the relative orientation of the axes of the gimbal and the position of the camera relative to the frame, parameters that are optimal for one position may not be acceptable for another: the system may self-excite (oscillate). Similarly, when installing a different payload (cameras of different weights and sizes), the dynamic characteristics of the system are also changed. The auto-tuning algorithm can pick up such parameters to ensure system stability in all positions in which the testing was performed. At the same time, the highest possible PID gains will be selected to ensure the highest possible quality of stabilization.

Thus, the following tuning algorithm is recommended:

1) Capture the frequency response in several operating positions of the gimbal (for example, in the normal position, tilted up by 45°, by 90°, down by 45°, by 90°). Capture response for the several cameras from the supported range (for example for the lightest and the heaviest camera).
2) Run automatic tuning as described below.
3) Evaluate the parameters found (see the subsection "**Stabilization quality evaluation**"). If the solution found is better than the previous one, apply the new parameters and write them to the controller. Check that system normally operates with new parameters in all positions and with different loads.

If after automatic tuning the system becomes unstable, then you can choose one of the options listed below:

Option 1: You can test the system in this position / with this load, and run the algorithm from step 2 again, adding a new test to the set of previous tests.

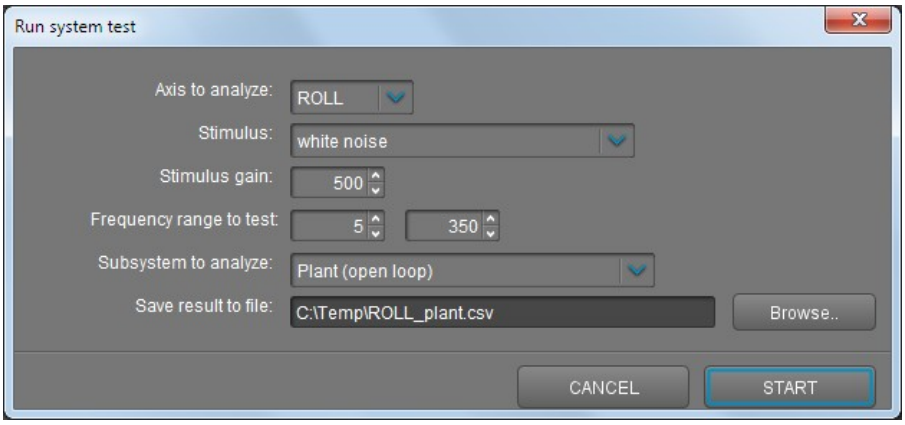
Option 2: adjust the parameters of the algorithm, setting more strict robustness conditions. Robustness is the ability of a system to resist from small changes of dynamic configuration without the need of adjusting parameters. The Algorithm can reduce the gain (slightly degrading the quality of stabilization), but at the same time increasing the system stability margin so that potential small variations in the system's mechanical properties do not lead to self-excitation. For this purpose, the algorithm has several settings that are described below.

Capturing the system response

To perform tuning, you have to capture the frequency response of the system, as described in the Section "System Analysis Tool". Prior this, the gimbal must be fully configured, calibrated, and operational, except for the PID controller parameters. It is necessary to set a small gain for PID, enough to perform a minimal stabilization. Since testing takes place separately for each axis, the remaining axes (that are not being tested at the moment) should be stable - there should be no self-excitation and no oscillations. It is also necessary to balance the camera before the testing.

Steps to perform to capture system response:

- Get connected to the controller. Motors must be switched on.
- On the "Analyze" tab, click the "Run test .." button. Set the recommended parameters:



  - Select the axis that needs to be tested.
  - "Subsystem to analyze" set to "Plant (open loop)" (or "Plant + notch filters (open loop)", if notch filters are already involved and you do not want to re-configure them)
  - Specify the frequencies to consider when tuning system. We recommend specifying a range of 5..350 Hz, as most of the artifacts and system dynamics are contained in this range.
    NOTE: Setting the lower border below 5Hz is not recommended for open-loop systems, as it can cause too big rotations.
  - The stimulus type set to "white noise". No other types are supported.
  - The stimulus gain should be sufficient to ensure that the amplitude response exceeds the sensor's internal noise and possible non-linearities such as friction in the bearings, but not causing the system to be saturated or overloaded (going out of acceptable range).
    Starting from firmware 2.72.x the stimulus gain is automatically attenuated to prevent a saturation.
  - Select the file where you want to save the result.
- After testing, the result is displayed in the "Analyze" tab in the form of amplitude-over-frequency and phase-over-frequency characteristics, in a logarithmic scale of frequencies and amplitudes.

Notes:

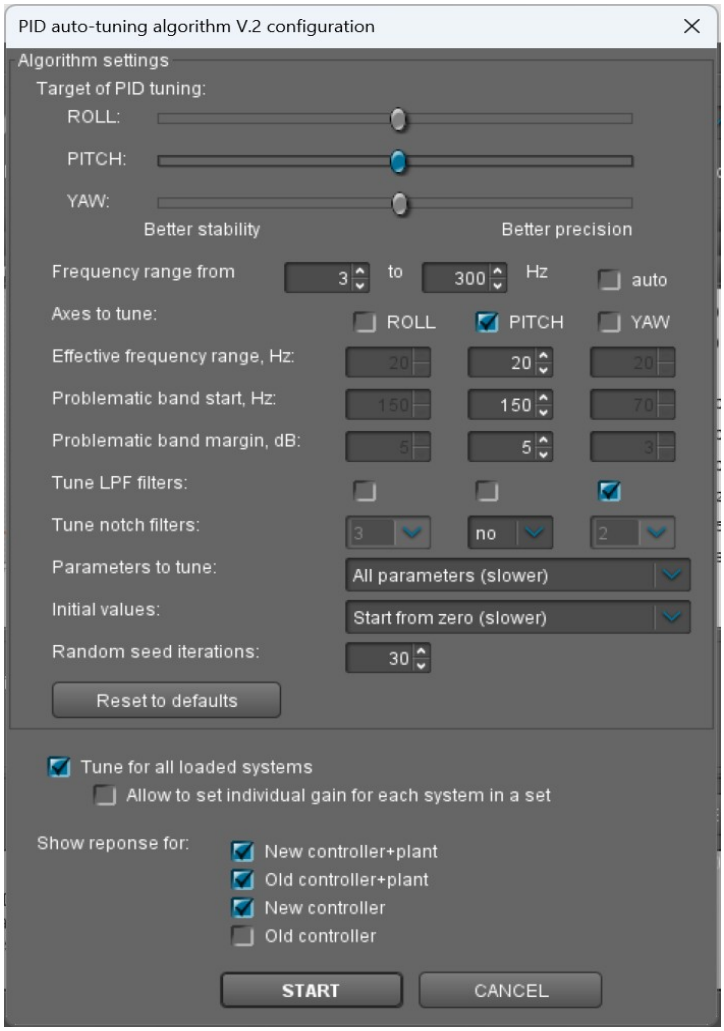
- This test is performed with the open-loop control causing that the motor can rotate at high angles. If the test is interrupted by an error - rerun it, reducing the gain of the test signal.
- If a bandwidth error occurs, increase the COM port baud rate to 256000 bps in the "Hardware" - "Serial connection" settings.

Running the optimization algorithm

1. Get connected to the controller and read parameters from it. It is also possible to run an algorithm without connection to the controller by loading profile from the file.
2. On the "Analyze" tab, add all the test data of the "Plant" type related to the system that you want to optimize. Add as many files, as you need by pressing the button "Add from file..". Do not mix data from different axes or different gimbals. If you plan to configure Notch-filters either, test the system in "Plant" mode (without filters). If Notch filters are configured and used, then test the system in "Plant + notch filters" mode - in this case, their effect includes in the system response and algorithm

does not take care about them.

3. When you click the "Optimize .." button, a pop-up window appears with the parameters of the algorithm. It contains the values that were specified during the last session. Some parameters are common, some are axes-dependant. Only a single axis for which test data is loaded is active.



4. Enable the "Tune for all loaded systems" checkbox. If it is not enabled, only the last file loaded will be taken into account. The meaning of remaining parameters of the algorithm is described below.

5. Click "START" button and wait for the algorithm to finish. If a solution is found, the new values and charts will be displayed (selected by the "Show response for").

6. Click "Apply new values" button and confirm the action. The new values appear in the corresponding fields in the GUI. Click the "WRITE" button to write and apply new parameters to the controller (or "Save to file" to write to the file if the controller is not connected). If the option "Update all profiles and write to the board" is selected, the new values are applied to all 5 profiles, and then all profiles are written to the controller.

   NOTE: You can interrupt the operation of the algorithm without waiting it to finish. In this case, the intermediate solution is still displayed, but it may be not optimal.

Starting from firmware 2.73.x ("Extended" and "Plus" family of the controllers), this algorithm can be run by the controller. See Version2 of PID auto-tune algorithm

Algorithm parameters

- **Target of PID tuning** – the slider sets the target for the optimizer. Choose a balance between high accuracy of stabilization and stability margin (robustness). If the gimbal has a tendency to self-excitation in certain positions, move the slider towards "Better stability" and repeat the automatic tuning.
- **Frequency range, from .., to** – used in terms of which frequencies to analyze. It ignores frequencies outside this range. Normally, a range of 1 – 250 Hz contains all useful information.
- **Effective frequency range, Hz** - limits the frequency at which stabilization is necessary (i,e, where S-curve is negative)
- **Problematic band start, Hz**
  Problematic band margin, dB – frequencies where resonances are possible or may appear when changing the position. Starting from this frequency, the algorithm adds a gain margin so that the gain peaks caused by potential resonances in some positions, do not exceed 0 dB limit. If you optimize the system using responses captured in multiple positions, you can skip setting this margin or can set it small.
- **Tune LPF filters** – if this option is enabled, the algorithm adjusts the parameters of the LPF filters. If disabled, uses the LPF filters with their current settings.
- **Tune notch filters** – if this option is enabled, the algorithm adjusts parameters of notch filters. The number of filters for automatic tuning can be set. The higher it is, the longer the algorithm works.

NOTE: To be able to configure notch filters, you need to test the response of the system without filters ("Plant only" mode)!

- **Parameters to tune** – chose between all parameters and single parameter - only the PID gain. Use the last option if the system is already well tuned, and you only need a slight adjustment of the gain to meet the robustness conditions for all loaded test data. This option also usable to run algorithm inside the controller: as it makes tuning very fast, it can be configured to run at the startup of a system.
- **Initial values** – you can select "from zero" or from the current values. Sometimes the optimal solution is difficult to find if the starting values are unsuccessful. In this case, try the "from scratch" option. If the system is already well tuned and you need only to adjust it slightly, use "from current values".
- **Random seed iterations** – if algorithm failed to find good values in a single run, specify a random seed to cover a bigger range of possible combinations. WARNING: It can significantly increase the required time, especially if algorithm runs on the controller.
- **Tune for all loaded systems** – if this option is selected, optimization is performed for all loaded files. If it is not selected, only for the last file loaded.
- **Allow to set individual gain for each system in a set** — if this option is enabled, the algorithm selects the optimal parameters that are common to all loaded test data, but at the same time allows varying the value of the "gain" parameter (PID controller gain), individual for each loaded system. This mode can be used to quickly adapt to cameras of different sizes: the gimbal manufacturer collects test data for different cameras from a wide range of different sizes and weights, the algorithm selects a universal set of parameters suitable for all loaded systems but considering that the optimum gain will be adjusted later. Afterwards, the controller selects only the gain parameter when installing a new camera — this process is simpler, more reliable, and faster than going through a multitude of PID controller parameters.

Evaluation of the quality of stabilization

The frequency response of the system provides very valuable information about the quality of the PID controller. When the automatic tuning algorithm finishes its work, it displays the expected (theoretical*) sensitivity curve* (S-curve) of the controller. Also at any time, you can capture the real sensitivity curve by running the "Controller + plant" analysis. The S-curve is displayed in green, having the horizontal axis with the frequency and the vertical axis with the gain in dB. It clearly shows how the PID controller performs its main task: rejecting of external disturbances.

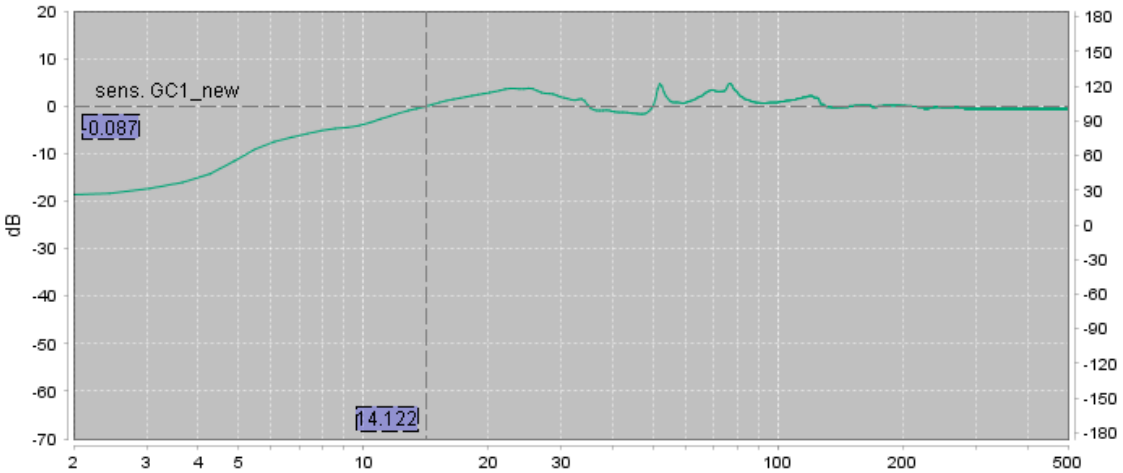
Effective stabilization frequency

Any PID controller has a limitation on its working frequency band, beyond which the controller does not eliminate external perturbations and even begins to amplify them.

Any external perturbation can be represented as a combination of oscillations at different frequencies that forms its spectrum. For example, the spectrum of a perturbation like a soft pressure on a camera with a hand is located mainly at low frequencies and is quite well compensated by the PID controller. Impacts during the walk are transmitted through the operator's body and affect both low and high frequencies, that is much harder to compensate. The similar case is vibration coming from the UAV frame, that can also have a wide range of frequencies. In both cases, good vibration dampers are required: they isolate high frequencies, leaving only low frequencies which PID controller can compensate.

Evaluation of the sensitivity curve

The S-curve gives a clear understanding which frequencies can be effectively compensated and which are out of reach for the particular PID controller. The smaller its negative value (in dB) and the higher is the frequency, where it becomes positive, the better. 0dB is when the external disturbances pass unchanged, negative if they are weakened, and positive if amplified. Try to avoid positive values above 3-5 dB - with high probability, the system self-excites at that frequency where the S-curve has a peak above 3-5 dB.



S-curve example of a real system

In this example, at the lowest frequencies, the system attenuates the external disturbances by -20 dB, its effective frequency of compensation is from 0 to 10 Hz (but at 10 Hz the attenuation is only -3.8 dB). Peak gain does not exceed 3.7 dB, which guarantees the stability of the system. The controller amplifies frequencies from 20 to 30 Hz, therefore it is necessary to isolate external disturbances at these frequencies as much as possible. There are also potential problems at frequencies of 50 and 75 Hz - the main mechanical resonances are located there, but the controller handles them correctly and it is not a problem.

By loading the data of several tests into the "Analyze" tool, you can compare the S-curves for different sets of parameters and select the best ones. It is also possible to monitor how the quality of stabilization changes in different positions and for different cameras by performing tests at each position / camera and comparing their S-curves (remember, you need to test "Controller + Plant" system to get the S-curve

displayed).

HINT: you can expand chart window to bigger size by pressing the "Maximize" button

PID controller parameters that algorithm can adjust and their meaning

- **P, D** - these are the main parameters that determine the feedback gain. The higher those are, the better, however too high values lead to self-excitation. Also, the actual gain of the controller depends on the parameters "Gain multiplier" and "Power".
  NOTE: "I" parameter is ignored and is always set to 0, as it affects only the extremely-low frequency range that is out of reach of this method.
- **Low-pass filter** — allows reducing the gain at high frequencies. Usually, at frequencies above 10-20 Hz, it is impossible to achieve good stabilization, so there are no reasons having high gains there. The low-pass filter allows getting rid of the negative effects caused by resonances and internal noise of a sensor, attenuating signal after a specified frequency.
  NOTE: the higher the rigidity of the system, the better, since the resonances (even if they are) are shifted to a high-frequency region that can be effectively filtered out. A too flexible and poorly-damped system will never provide a good stabilization quality because its resonances are shifted to the low-frequency region.
- **Notch filters** — sometimes strong resonances in the mechanical system can be filtered by the narrow-band notch filters. However, the property of the 3-axis gimbal is that the resonance frequencies can drift when the relative position of the gimbal parts changes or when different cameras are installed. Therefore, notch filters are difficult to configure and use. Our auto-tuning algorithm can decide, is it worth to use notch filter, and can adjust its bandwidth to cover possible variations of the resonant frequency along all loaded test data.

8. 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** – Additional 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.
- **Calibrate RC inputs** button (*frw. ver 2.65+)* allows to calibrate up to 5 inputs by 3 points: min, max and neutral. The procedure is the following:
  1. In the RC input calibration dialog, chose inputs you wish to calibrate;
  2. Press the "RESET" button to clear the existing calibrations. Let system few seconds to apply changes;
  3. Move sticks to their extreme positions. Min and max values will be remembered and displayed on the gauge indicators;
  4. Move sticks in a neutral position and press the "CALIBRATE" button;
  5. On the gauge indicators you will see that now the RC signal on each input covers the full range when the stick is moved, and returns precisely to 0 when the stick is in a neutral position.
  6. You can fine adjust calibration manually by changing values in the input fields. Press the "SAVE AND CLOSE" button to apply changes.
- **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%.
- **RC control modes**
  - **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 is constant and defined by the "SPEED" parameter, though near the target angle speed gradually decreases.
  - **ANGLE MODE [TRACKING]** *(frw. ver. 2.68b7+)* - the same as ANGLE_MODE, the only difference is that the rate of turn does not depend on the "SPEED" parameter, but matches with the rate of turn of the RC stick, filtered by the LPF and acceleration limiter.
  - **SPEED MODE** — RC stick will control the rotation speed of the camera. If stick is centered- camera stands still, if stick is deflected, camera starts to rotate, but does not exceed min-max range. Speed of rotation is proportional to the stick deflection (taking into account "Expo curve"

© Basecamelectronics® 2013-2024

parameter), and to the "SPEED" parameter.

  - **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 only 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. System will start with these angles in SPEED mode.
  - **Do not update initial angle** – set this option to not update the initial angles in the EEPROM after executing "Set tilt angles by hands" menu command or "Swap RC PITCH - ROLL", "Swap RC YAW-ROLL" commands. If not set, system will start with the new initial angles next time.
- **RC Sub-Trim** – correction for transmitter inaccuracy. At the neutral point the transmitter should produce a PWM signal equal to 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 neutral position, and press AUTO button. Actual data becomes new center point. Press WRITE button to apply settings. *Starting from the 2.65 version, it is possible to calibrate RC input by 3 points: neutral, min and max. It's a preferred way of RC calibration.*
- **Dead band** — adjusts dead band around the neutral point. There's no control while RC signal is inside this range. It helps to achieve better control by eliminating jitters from unintended movement of the stick around neutral point. This feature works differently in SPEED and ANGLE modes: in the SPEED mode, dead band is created around neutral point, in the ANGLE mode, dead band tracks stick position, and small jitter in this position is eliminated.
- **Expo curve** – adjusts the curvature of an exponential function applied in the SPEED control mode. 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.
- **Limit Accelerations -** this option limits angular accelerations in case of hard RC or "Follow" control, forming S-curve in a angular distance profile. Use it 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.
- **Limit jerks** *(frw.ver.2.65+)* - similar to the acceleration limiter option, but limits the "jerks" – the rate of change of an acceleration. It allows to form the S-curve in a speed profile. This parameter defines the time in milliseconds, required to rise the acceleration from zero to a nominal value. Use it careful because it adds a noticeable delay in control and may cause instability in the combination with the other parameters.
- **PWM bypass** – a mapping that allows to redirect a signal, captured on the RC serial input, or Sum-PPM input, or ADC1..3 inputs, to the special 'servo out' pins in the PWM format. This signal can be used to drive a hobby servo or IR remote camera trigger, for example. On the SimpleBGC32 boards,

these pins share PWM output function with other functions:

	"Regular", "Tiny", "Extended"	"Pro"	"CAN_MCU"
Servo1	FC_ROLL	SERVO_PWM1*	FC_ROLL
Servo2	FC_PITCH	SERVO_PWM2*	FC_PITCH
Servo3	RC_PITCH	SERVO_PWM3*	AUX3
Servo4	AUX1	FC_ROLL	AUX1

* *Separate interface located on the logic board*

To enable servo output on any of these pins, make sure that it is 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 to connect other RC-controlled devices like IR camera shutter trigger.

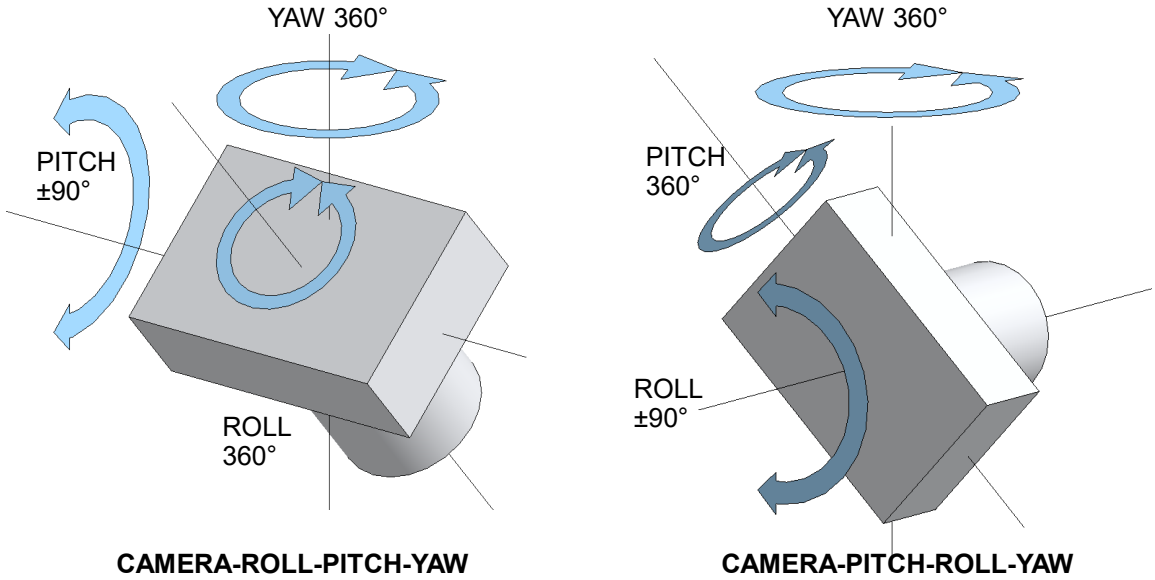
When connecting regular hobby servo to these ports, there are two options to get +5V to supply them:

- 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).

Duty mode flag changes the output waveform. When enabled, the duty cycle (time, in percent, when the signal is "on" during the single PWM period) is proportional to the RC signal input. This mode is suitable to drive loads like LED, DC motors, bulb lights, etc. - anything that accept a power regulated from 0 to 100%. Note that the controller's servo output pins can't provide high currents, so an external FET is required to drive a high-power load.

Order of Euler angles

The control of the gimbal from RC transmitter or joystick is made by 3 separate angles (called "Euler angles"): ROLL (to control horizon), PITCH (to tilt up-down), and YAW (to turn left-right). To rotate camera from one direction to another, we can make three separate rotations by three axes. But the order of rotations plays a role. You can change the order of rotations in the parameter "**Order of Euler angles**". The difference is displayed in the picture below:



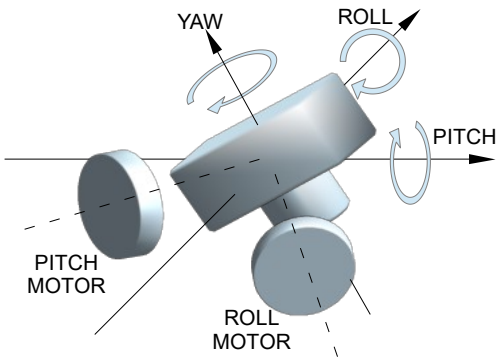
Order of Euler angles

The default order is “Camera → PITCH → ROLL → YAW → Frame”. As you can see, it does not allow to roll camera to angle greater that ±90 degrees, because in this case PITCH axis becomes equal to YAW axis and the PITCH angle can not be distinguished from the YAW angle. it's recommended to set Min.angle=-85, Max.angle=85 for ROLL axis in the “RC” tab, to not allow this case.

In opposite, if you select order “Camera → ROLL → PITCH → YAW → Frame”, you can roll camera to any angle (including infinite 360 degree rotation), but can not pitch it more than ±90 degrees. The same, setting Min.angle=-85 and Max.angle=85 for PITCH will help to prevent entering forbidden area.

> **NOTE:** This setting is profile-based. It means that you can assign different Euler order to different profiles and switch between them on-the-fly by menu button. Stabilization will be not interrupted.

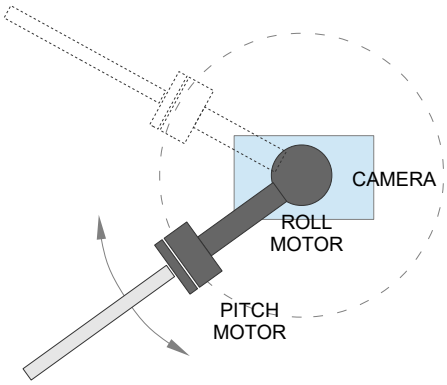
"Cam – YAW – ROLL – PITCH" - This order of the axes is needed for special cases when you need to point the camera to a certain point on a ground and hold it at all pan and tilt angles of a frame. In a typical case, the stabilization axes PITCH and ROLL are rotated together with the rotation of the camera (or a frame) by YAW axis. This means that if the camera is looking down at a certain ROLL and PITCH angle, and you pan the camera (or a frame in 2-axis system) - its optical axis draws an arc, ie not locked to a point. If you choose the order of the Euler angles "YAW-ROLL-PITCH", the optical axis of the camera is always locked to a point on the ground regardless of the evolutions of a frame and YAW rotations of a camera. This mode is suitable for 2-axes gimbals mounted on a gliders with the camera pointed down.



> **NOTE:** the RC control in space of Euler angles by ROLL and PITCH axes is no longer tied to the orientation of the frame, but is linked to the Earth. It means that when you turn the camera by 90 degrees left or right, ROLL becomes PITCH and visa versa. Also, you must use a magnetometer to prevent a drift of the internal coordinate

system relative to the Earth, caused by the drift of the gyroscope.

"Cam – PITCH(M) – ROLL – YAW(M)",
"Cam – ROLL – PITCH(M) – YAW(M)"



In this mode the ROLL axis is always linked to the horizon (locked), but the PITCH and YAW axes always match the corresponding motor's axes. Camera orientation is not linked to the ground any more (and isn't described by the Euler angles), but still can be controlled from RC in SPEED or ANGLE mode.
As an example, this mode can be used to build a 2-axis head where the ROLL motor is linked to the camera and stabilizes relative to the ground, whilst the PITCH motor is linked to the boom and can work in any position on 360-degrees circle regardless its alignment to the Euler PITCH or YAW angles.
This "M" mode has some differences from a normal mode:

- The "motor-related" axis is stabilized in some extent but the motor tries to maintain its angle. The strength is defined by "OuterP" parameter in PID settings. It can be set to 0 to avoid this behaviour (mostly to be used in conjunction with the RC SPEED mode with endless rotation).
- MIN. and MAX.ANGLE parameters are considered motor-related.

Screen-related X,Y,Z *(frw.ver. 2.72.x)*

A very special mode where the axes of control are always linked to the viewport of the camera: YAW for panning (X axis pointing left-to-right), PITCH for tilting (Y axis pointing up-to-down), and ROLL for rolling (Z axis pointing forward as a camera's FOV). Works only in conjunction with the RC SPEED mode.

Control in motor coordinates ("Servo mode") *(frw.ver. 2.72.x)*

If enabled, gimbal is controlled in motor coordinates rather then global Euler angles. Motors hold their position; RC command for SPEED / ANGLE relates to the motor's axes and angles rather then Euler axes/angles. This mode is similar to a conventional PTZ head (but the gyroscope sensor is still used for the stabilization).

> **HINT:** To disable the motors holding their position when the frame is rotated, set OuterP = 0 in the "Stabilization settings".

Selecting different axes for stabilization in 1- or 2-axis gimbals

Starting from the firmware version 2.62b7, it is possible to assign different *stabilization axes* to a motor in different profiles, or use automatic selection of best matching axis for a particular motor. This group of parameters is located in the "Stabilization settings" tab, available for the "Expert" level of view.

Example: 2-axis gimbal is build in a configuration "Camera- PITCH motor - ROLL motor". In the profile 1 all stabilization axes are set to their defaults – i,e, ROLL motor stabilizes ROLL angle, and PITCH motor stabilizes PITCH angle. In the profile 2, ROLL motor is configured to stabilize YAW axis. When gimbal is switched from profile 1 to profile 2 and the frame is tilted 90 degrees forward, configuration becomes "CAM - PITCH - YAW".

Second option is to enable automatic selection of the YAW axis for a ROLL motor in profile 1. In this case, gimbal can be switched to the "CAM – PITCH – YAW" configuration without need of switching profile, just being powered ON in new position.

> **NOTE:** It is not required to change parameters related to a motor, like output port, PID gains, Encoder, POWER and so on, because motor is not changed. But parameters related to Euler axes, like RC setting, Follow mode settings – need to be configured for all axes that can be stabilized by this motor.

For 3-axis gimbals, do not change these settings from their default values. Also, for some arrangements of motors in 2-axis gimbals (like ROLL-YAW or PITCH-YAW), when frame is tilted – motors are assigned to Euler axes automatically on-the-fly, so no need to use these settings.

Step signal source *(frw.ver. 2.65+)*

This feature translates any signal source to a new virtual signal source, that changes its value by the fixed steps. The main purpose is to allow to use 2-way or 3-way switches to adjust parameters of a system. For example, you can link the "Follow speed" parameter to a switch: each click "Up" increases value, click "Down" decreases it. You can even use the existing joystick for this, extending its functionality: when the mode button is not pressed, joystick controls gimbal movements as before; when the mode button is pressed, it acts as 3-way switch and can adjust any parameter of a system: up to 4 parameters could be adjusted by the regular 2-axis joystick.

- **Slot to edit** - select one of the six slots to edit its properties. It allows to create up to six new sources.
- **Signal source** - select the source of a signal that will be used to detect "clicks" and to change the value of this channel. "Click" is detected when the signal goes below or above a threshold in 1/3 of the full RC range. In terms of PWM signal, it is below 1330 us and above 1630 us.
- **Hold menu button to switch signal source to this functionality** – if this option is set, input is active only when menu button is held. Any other function of this input is disabled. It allows to use a single joystick for a control of gimbal's orientation, and as a source for a step signal to adjust its parameters, for example.
- **Number of steps** - The full RC range is divided by this number and produce the value of a single step of adjustment.
- **Mode of operation** - If the "Count clicks" option is selected, the value is set according to the number of clicks in sequence. For example, to set the value of 5 steps, click switch 5 times quickly in sequence. If the "Up-down" option is selected, the value is incremented or decremented on each click. Short sound is emitted to confirm each click. When you "click and hold", auto-repeat function is started.
- **Initial value** - On system start, the initial value of a channel may be set to 0 or to 50% of a full range.

When you have configured the step signal source, you can use it as any other RC signal source. It is denoted as "STEP_SIGNAL_1..6" in option lists. Particularly, you can use it with the Adjustable Variables to adjust the value of any parameter in the "analog" type, or to trigger actions in the "trigger" type. The later allows to duplicate the function of the "Menu" button, adding extra switches and linking actions to numbers of clicks, if single button with their five actions is not enough.

Gamepad controller (Joystick) usage in the GUI

Starting from GUI version 2.70b3, it allows connecting a gamepad controller (joystick) to the PC and sending its values to the gimbal. It may be helpful for testing gimbal during the setup, emulating a regular joystick or a remote controller.

Values are passed on via the “API_VIRT_CHxx” signal source. They may be assigned for controlling gimbal's main axes in the "RC" tab or for adjusting run-time parametes or triggering actions in the "Adjustable variables” tab.

The mapping between controls and API_VIRT channels is given for the Xbox One controller as an example:

Control	Identifier	source channel
Left stick X axis	x	API_VIRT_CH1
Left stick Y axis	y	API_VIRT_CH2
Left trigger (positive), Right trigger (negative)	z	API_VIRT_CH3
Right stick X axis	rx	API_VIRT_CH4
Right stick Y axis	ry	API_VIRT_CH5
-	rz	API_VIRT_CH6
4-position navigation pad (D-pad)	pov	API_VIRT_CH7
Button “A”	0	API_VIRT_CH8
Button “B”	1	API_VIRT_CH9
Button “X”	2	API_VIRT_CH10
Button “Y”	3	API_VIRT_CH11
Left bumper	4	API_VIRT_CH12
Right bumper	5	API_VIRT_CH13
“View” button	6	API_VIRT_CH14
“Menu” button	7	API_VIRT_CH15
Left stick press	8	API_VIRT_CH16
Right stick press	9	API_VIRT_CH17

For other gamepad controllers, you can find out the mapping by observing the displayed values in the “PC Gamepad” tab.

To start data transmission, select controller from the drop-down list and press the “ENABLE” button. If controller is connected after GUI is started, it may be required to restart GUI.

You can specify the rate at wich new values are readed from the gamepad and sent to the gimbal, by the “Interval” parameter. The less is interval, the more responsive gimbal control is. The optimal value is 20-50 ms (which corresponds to 50-20 Hz of the update rate) .

> **NOTE:** Be careful to not exceed the serial bandwidth if setting interval too low! Each message takes about 40 bytes; at the default 115200 baud rate and 50Hz update rate, it takes about 17% of the bandwidth, which is not critical and leave a lot of space for other communication.

The gamepad thread is paused when the "Monitoring" thread is paused; also, GUI pauses this thread automatically when needs to free up serial traffic for service tasks (writing/reading parameters, etc.).

Warning: Gamepad connection was tested on Windows only; other OS are expected to work, but not tested.

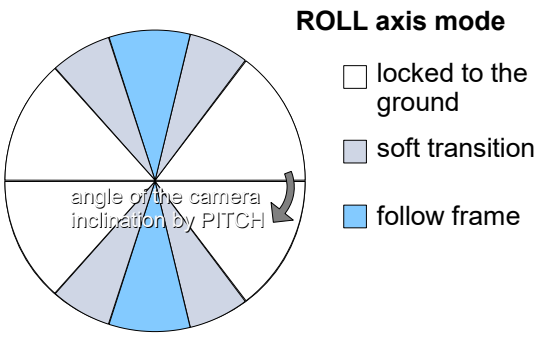
9. Follow Mode Settings

Follow Mode is a special control mode that makes the camera "follow" a movements 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 by RC or joystick
- **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 aircraft's angles to these outputs in the PWM format (that servos use). SimpleBGC can get this information and use it to control a camera such way that it tracks the tilting of aircraft. 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 in which 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. But if encoders are used, this is not a problem.

  - **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**: 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. The value is expressed in degrees for the standard 60-degrees follow range, and is proportionally stretched when range changes.
- **Expo curve:** when the expo curvature parameter is greater than zero, 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 (up to ±45 degrees). This feature gives considerable freedom in camera operation, from fine and smooth control to very fast movements.
- **SPEED -** adjust the speed of the camera rotation. Don't set big values that motors can not handle (if

motor does not produce enough torque to move the camera, it will skip steps and synchronization will be broken). In this case an acceleration limiter may help to have high speed but not to miss steps.

IMPORTANT NOTE: For high SPEED values (above 50-100) it's strongly recommended to set "LPF" parameter greater than 2-3, "Expo curve" parameter greater than 50, and "Dead-band" parameter greater than 3-5 degrees. 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 speed control in the "Follow" mode. If this value is set high, fast movements of the handle will be smoothed. But it requires careful operation and a little training to prevent unwanted oscillations of the camera. it's recommended to not set it below 2.
- **RANGE** *(frw. ver. 2.62b6+)*: this parameter defines the angle (in degrees) of the deflection of an outer frame that generates a control signal of full strength. Outside this area control signal is clipped. Default value is 60 degrees.
- **Follow rate inside dead-band** – very soft control to always keep camera in the center of the dead band. Set it to 0 to disable this feature.
- **Home position offset**
  - with encoders: it allows to adjust home position. Home position normally is set at the zero motor angle, but may be shifted by this parameter.
  - without encoders: it allows to fine adjust the initial position of the camera if it's "snapped" to motor's poles in the "follow" mode. For PITCH and ROLL motors there is an option to calibrate offset automatically. To do this, power on the system, hold frame leveled, move camera to desired position relative to a frame by joystick or RC, and press **AUTO** button.
- **Use Frame IMU, if possible** – if 2nd IMU is connected, system can use it to detect the motor angles instead of method based on electrical field estimation. IMU-based method is more reliable, because it will not loose synchronization like in case of electrical field.
- **Apply offset correction when axis is not following** – when any axis is not following, corresponding motor should not produce control signal for it and for other axes. But when the axis enters follow mode (for example, ROLL may be switched from "Lock" to "Follow" mode depending on PITCH angle), or when frame is rotated such way that motor starts to stabilize another axis – motor should produce zero control signal, even if it's not in the "normal" position. it's recommended to have this option enabled.
- **Disable follow mode by holding menu button pressed** *(ver. 2.62+)* - use this function to temporarily disable the follow mode by holding the menu button, without need of profile switching. It's recommended to disable action assigned to the 'Long press' menu command, if button will be held more than 3 seconds.
- **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 the encoder-less single IMU configuration, there is no such information. But most of FC's have servo outs for connecting gimbals, that may be used to obtain such information. These outputs should be connected to SimpleBGC controller through EXT_ROLL and EXT_PITCH inputs and the following steps to be performed:
  - In the **RC** tab make sure that inputs EXT_ROLL and EXT_PITCH are not assigned to any function.
  - In the **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).

- Turn gimbal ON. It should be properly tuned and stabilization should work to this step.
- Push **AUTO** button in **External FC Gain** group and smoothly incline aircraft's frame to different directions by all axes for 10-30 seconds. Controller will match signal from the aircraft and from the IMU sensor and find a relation between them.

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 ±45º 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 it's the wrong direction of rotation of the motors and you need to change the **INVERT** flag in the "Hardware" tab.

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

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.

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

10. 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, Calibrate ACC (temp. compensation)** – the accelerometer calibration, works the same way as the button selection in the GUI. Runs regular calibration or temperature calibration.
- **Calibrate Gyro, Calibrate Gyro (temp. compensation)** – gyroscope calibration. Runs regular calibration or temperature 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.
- **Motors OFF safely** *(frw.ver. 2.66+)* – if camera is not balanced well, it does not falls freely, but slowly goes to an equilibrium position, then motors are turned OFF.
- **Reset controller –** completely restart system, like doing a power cycle.
- **Frame upside-down (inverse over the middle motor)**– configures system to work in inverted position when the middle motor rotates by 180 degrees. New configuration is stored to EEPROM and applied 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. YAW axis returns to position where gimbal was started.
- **Level ROLL to horizon,**
  Level PITCH to horizon,
  Level ROLL, PITCH to horizon – moves camera to the "leveled" position.

  NOTE: prior to 2.70b4 it saves new INIT.ANGLE=0 to EEPROM, if "Protect initial angle" flag is not set. Starting from 2.70b4, it never changes the INIT.ANGLE value.
- **Center YAW axis** – moves camera to the home position by YAW axis (YAW encoder should be installed).
- **Bind RC receiver** – start bind procedure. This is applicable only for Spektrum satellite receiver, as described in the "RC" section.
- **PID auto-tuning** – start auto-tuning of PID parameters with the options saved from the latest auto-

tuning session.

- **Menu button press** – this is emulation of menu button single press. Example of use case: assign toggle switch without fixation on your RC receiver to CMD channel; assign this action to High or Low state of CMD channel, depending on toggle switch's active state. Now you can execute up to 5 actions by pressing switch remotely, as you do it be pressing the menu button.
- **Run script from slot 1..4** – execute your scenario from any slot. See User-written scripts section.
- **Untwist cables** - if any motor made a revolution greater than 180 degrees, the system will rotate the camera along this axis by 360 degrees in the opposite direction to minimize cable twisting. Camera after the maneuver will remain in the same position where it was. The following conditions are required for proper operation:
  - It must be known angle of the motor (via the encoder on the axis or second IMU-sensor)
  - Twisting can be tracked only in the process. If the system has already started with a twisted cable, it is impossible to figure out.
  - The axis of the motor should be not inclined too much (no more than 60 degrees from its normal position)
- **Rotate YAW ±180 from current position** – turns camera 180 degrees from the current absolute position by the YAW axis. The direction of rotation is chosen automatically: for the encoder-enabled gimbal, it will find the most optimal path to avoid hardware limits of all motors. In non-encoder gimbals, it will reverse direction on every call of this function.
- **Rotate YAW 180 from home position** – Rotates camera 180 degrees from the home position by the YAW axis. Unlike previous command, angle is calculated relatively to a frame, so YAW encoder should be installed. To return back, use "Center YAW axis" command.
- **Switch YAW 0/180 from home position** – use this action to switch between forward and backward positions (0 and 180 degrees, respectively) for the YAW axis relatively to a frame. YAW encoder should be installed.
- **Switch portrait mode (ROLL 90/0)** *(frw.ver. 2.61+)* – rotate camera clockwise to a portrait mode. In this mode you can switch profiles – camera remains tilted.
- **Setup and start time-lapse motion** *(firmware ver. 2.62+)* – with the hand-held gimbals, it allows to program a motion sequence to move from one position to another, and process it within a given time, defined by the "Time-lapse time, sec." parameter. Detailed description is in the section "Time-lapse shooting".
- **Repeat last time-lapse motion** *(frw.ver. 2.63+)* - repeats the last time-lapse motion sequence precisely. Makes a 3-second pause before start.
- **Load profile set from slot 1..5,**
  Restore settings from backup slot *(frw. ver. 2.63+)* - loads all settings (general and all 5 profiles) from the given slot.
- **Invert RC ROLL, [PITCH, YAW]** *(frw. ver. 2.66+)* – invert RC control for a given Euler axis.
- **Snap to fixed angle** *(frw. ver. 2.66+)* – force all axis to snap to a fixed angle, if the snapping option is enabled in the section "Service" - "Force to new position by hands".
- **Camera Rec/Photo event,**
  Camera Rec/Photo event *(frw.ver. 2.66+)* – generates an event and send it via the Serial API to the subscribers. It could be used by external applications or external devices, connected to a gimbal by the Serial API protocol.

Additionally, there are special actions if you press 'menu' button more than 5 times:

- **10 times –** full erase of all settings.
  ver. 2.55+: make 9 short clicks than 10th long press

ver. 2.66+: does not erase settings, but restores them from the backup slot, if present. **WARNING:** Use this option for recovery only, if board is not accessible from the GUI or no other ways to make it working.

- **8 times –** all COM-ports function will be reset to their defaults: parsing of the SimpleBGC serial protocol only.
- **12 times** *(frw. ver. 2.50+)* – serial speed will be reset to default value 115200 and all COM-ports function will be reset to their defaults. Use it if you changed the speed or functionality of COM-port and the board is not accessible anymore.

Working positions

- **Frame inversion auto detection** – if enabled, the controller detects if system is started in the "inverted frame" position, when frame is turned by 180 degrees over the middle motor (commonly it's ROLL motor). This mode equals the menu command "Frame upside-down (Inverse over the middle motor)".
- **Brief-case mode auto-detection** – it's useful in the "Follow" mode, when you turn frame by 90 degree over any axes, and do not want for camera to follow the frame. Converting to briefcase position is very simple – just hold camera and rotate frame by 90 degree. Also it will be automatically detected at startup, if you have the "Follow" mode enabled by default.
- **Upside-down PITCH auto-rotate** – when the frame is turned upside-down over PITCH motor, camera will be rotated by 180 degrees to track new position of the frame. Be sure that PITCH angle is not limited in the RC tab, to allow camera to make this movement.
- **Return to home position on profile switch** – normally, camera does not move when you switch to a different profile, but keeps it position. You can enable this option in any profile, to move camera to a home position defined by the "INIT. ANGLE" parameter in the "RC" tab, when switching to this profile.

Profile sets *(frw.ver.2.65+)*

The profile-based management of system parameters may be not enough to cover all possible use cases. For example, if we use profiles to switch between the modes of operation (like "Lock", "Follow" modes, various speeds/sensitivity settings, etc), we can not use them to switch between different mechanical configurations at the same time (different camera weights, different sets of equipment and so on). To solve this problem, we can use a "profile sets". Each set holds all general parameters (including IMU sensor calibrations) and all parameters in five profiles.

> **NOTE:** Some extra parameters like adjustable variables and scripts, IMU temperature calibrations, are not included in the profile set.

To create a new set, just configure all parameters in all profiles as usual, select a slot, and press the "SAVE" button. To load parameters back, press the "LOAD" button for a selected slot. **Note, that all existing parameters will be lost (replaced by the parameters from the set)!**

To load a profile set without PC connection, you can assign the menu action "Load profile set #1..5" to the menu button (as well as RC CMD channel or a trigger-type adjustable variable), or configure the startup action, as described in the section Execute action at system start.

There is a special slot called "BACKUP". It is intended for a (very rare) case when the settings in the EEPROM becomes corrupted – in this case this slot is loaded automatically. Also you can load it by the special combination - "9 clicks-and-hold" of the menu button (see the Menu Button section).

Startup behavior

- **Center YAW axis at startup** – after system power-on, move camera to the neutral position by the YAW axis. This function requires encoder on the YAW axis to be installed and configured.
- **Remember last used profile** – when profile is switched by the menu button, it becomes default: next time system starts, it will be activated by default.
- **Search and move motors to home position at startup** *(encoder firmware only)*- If enabled, motor will move to home position at startup before system initializes. If motor has a freedom of rotation greater than 360º (but not infinite), system will search one of the hardware limits, and home position will be counted from it according to **Min.,Max. angle** software limits set in the "Encoders" tab. They should be set 5-10 degrees ahead of hardware limits.

Battery Monitoring

On all 32-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.

Needless to say, it is important to have a good calibration of voltage sensor to properly detect battery charge level. Also take into account that under big load, measured voltage may be lower than actual, if wires are too thin.

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 20mA (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.

In the "Buzzer" settings group, you can choose which modes should be sounded.
In case of gimbal with encoders, you can adjust sound volume by slider.

Custom melody *(frw.ver. 2.66b8+)*

You can assign custom melodies to be played by the motors for some actions. Leave the field empty to use default melody.

Melody is encoded as a text string in format: "**a, b, c c c c c ..**", where:

a - duration of each note, 1..255 (it's a number of 8ms samples, so duration is a*0.008 seconds)

b - envelope decay factor, 0..15. Value 0 for no decay, 1..3 for a fast decay, 4 and above for a slow decay

c - space-separated list of notes from **C#5** to **B8**. Special symbols are supported:

; (semicolon) - to restarts an envelope

Pxx - to make a pause xx samples and restart envelope

For example, this is how the default intro melody is written:

```
6, 3, C#5 C#5 C#5 C#5 P10 F5 F5 P4 A5 A5 P4 C6 C6 P4 D6 D6 P4 E6 E6 P4 F6 A6 B6 B6
```

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.

Additionally, main LED may be used for other indications, available under "Service" – "LED indicator" parameter:

- **Blink profile number** – the number of short blinks (1 - 5) indicates a currently selected profile
- **Blink battery voltage** – the number of short blinks indicates the charge level of the battery, from full to empty:
  - 100% - 60%: LED is constantly on
  - 60% - 40%: blink once
  - 40% - 20%: blink twice
  - 20% - 0%: blink 3 times
  - 0% and below: constantly blinks

Battery charge level is defined by the comparing of actual voltage with the range defined by the parameters

"**Full battery**" and "**Low voltage - stop motors".** You need to set these parameters properly for the model of battery being used, even if corresponding functions are not enabled.

Misc. settings

- **Emergency stop** – enable this option to immediately stop motors if problems are detected, like:
  - High rate of I2C errors;
  - Over-temperature, short-circuit, under-voltage, over-current protection in motor drivers (some boards only);
  - IMU data compared to encoder data inconsistency;

    GUI displays the reason of error, if connected. To restore system after emergency stop is triggered, press a menu button once to make full restart of a system.
- **Restart system after a delay, ms** – if this parameter is set to any non-zero value, when the emergency stop error is triggered, system will be restarted automatically after given time.
- **Max. time to work under full load, sec.** *(frw.ver. 2.65+)* — when any motor is fully loaded (by applying external force or if system is badly unbalanced) during the given time, system generates the Emergency stop error and turn OFF motors. Applicable only for the encoder firmware. This option could be used to prevent a damage when working gimbal is put in the bag, for example.

Automated motion tasks

These parameters are used when gimbal executes a motion command ("Home position", "Rotate ±180", "Switch portrait mode" , etc.)

- **Speed** – the speed of movement. Units equals to the "SPEED" in RC tab.
- **Acceleration** – the acceleration of movement, degrees/sec^2
- **Auto-pass motor's hardware limits** - if particular motor has a limited range of rotation (configured as min., max. angles in the "Encoders" tab), and the commanded camera rotation approaches the limit, system will automatically do a quick rotation in opposite direction to pass over the restricted area and to keep rotation from its other side. This flags are per motor.

Execute action at system start

This group of parameters is intended to configure the actions (up to 4) that can be executed at system start when power is turning ON and special combination of RC inputs is applied. For example, someone can configure the combination "Hold sticks Up-Left when pressing menu button" to load profile set #1, and "Stick Down-Right when holding menu button" to load profile set #2, and any other combination to calibrate gyroscope, and so on.

- **Signal source** – which input will be monitored for a signal. If two inputs are selected, a condition should be satisfied on both.
- **Threshold** – the value of the signal on a given input should exceed the given threshold. If the threshold is negative – the value should be less then the threshold. The nominal range of a signal is -500...500. For example, if the signal source is an analog joystick, being properly calibrated it will produce value -500 declined fully left, 0 in the neutral position, and +500 declined fully right. Set a threshold to -400 or 400 depending on a direction you want.
- **Menu button** – if enabled, requires a menu button to be pressed additionally to other conditions.
- **Action** – which action to run when all conditions are satisfied. Not all of actions are supported in this mode, though they all are listed.

Time-lapse settings

The SimpleBGC gimbal controller can be configured to rotate the camera with very low speed and high precision. Combined with the time-lapse shooting mode in a camera, it allows to achieve various visual effects. There are several ways to implement this function that are described below.
It is important to have precise gyroscope calibration: even slow gyroscope drift will cause visible unwanted motion in the time-lapse video after speed-up. The result is better if gimbal has encoders.

Menu command "Setup and start time-lapse motion" *(firmware ver. 2.62+)*

Prepare:

1. In the GUI, set the time-lapse parameters in the "Service" – "Automated motion tasks" group (You can set different time in different profiles):
   - **Time-lapse time, sec.** - time required for the time-lapse motion. There is also an option to adjust it by the adjustable variable "TIMELAPSE_TIME", linked to a potentiometer, for example.
   - **Acceleration in and out time, %** - camera will softly accelerate at the start and de-accelerate at the end of a trajectory during the given amount of time, in percents of the total time.
   - **Frame angles are fixed** - Normally, gimbal's frame is not moving during the time-lapse, so we can use this information for a gyro drift correction. It works for the encoder gimbals or when the 2nd IMU is mounted in the "Above YAW" position.
2. Assign a command "Setup and start time-lapse motion" to the menu button.

Make a time-lapse:

1. Configure your camera for a time-lapse shooting
2. Start gimbal in normal mode and move the camera to the final position, where motion should be finished. **Fix the gimbal's frame very firmly and do not rotate it until the time-lapse is finished!**
3. Activate this function by the menu command "*Time-lapse: set in and out angles and start motion*". The "calibration" sound is emitted. You have 10 seconds to move the camera to the initial position where time-lapse motion should begin (you can use a joystick, RC remote or move the camera by hands and fix it there). Don't forget to start time-lapse sequence in the camera.
4. After 10 seconds the "confirmation" sound is emitted and time-lapse motion starts. The camera will move to the final position within specified time. You have not to touch the camera or gimbal until time-lapse is finished. On complete, the "completion" sound is emitted and gimbal returns to normal operation.

To interrupt time-lapse motion at any time, activate this function again or turn gimbal OFF and ON.

Alternative method: in firmware 2.69b1+ it's possible to set the final angle by the command "*Time-lapse: set out (final) angle*". Once it's set, move gimbal manually to the initial angle (you are not limited in time now), and start timelapse by the command "*Time-lapse: set in and out angles and start motion*". It will skip setting the target angle and start motion immediately.

Repeating the last time-lapse motion

It is possible to repeat the last programmed time-lapse motion multiple times with high precision. Just activate menu function "Repeat last timelapse": gimbal will return to the start point and after 3 seconds will start new motion sequence. To get maximum precision, do not move gimbal's frame between shots. A little difference of framing between two shots may present, because of the IMU sensor drifting or a change in environment conditions.

Creating a time-lapse sequence using scripting language

You can create a script that makes a required motion, and start it by the menu command. Refer to the section "User-written scripts" for information how to write scripts. Below we provide an example of a script that can help in time-lapse shooting:

```
# EXAMPLE: TIME-LAPSE SHOOTING
# Let system to know that the frame is still, to compensate a drift of gyroscope;
SET_ADJ_VAR NAME(FRAME_HEADING_ANGLE) VALUE(0)
# Set the 'gyro trust' parameter low enough to better compensate drift of gyroscope
SET_ADJ_VAR NAME(GYRO_TRUST) VALUE(60)
# (Optional) move camera to the desired initial position. Skip this command to start from the current position
#ANGLE PA(0) RA(0)
# Pan left with the speed 0.1 degrees/sec and tilt up with the speed half slower.
SPEED YS(0.1) PS(-0.05)
# Wait 10 minutes
DELAY TIMEOUT(600)
```

Force to new position by hands *(frw.ver.2.65+)*

This group of settings allows to re-position a camera by hands. This option can be enabled/disabled for selected axis. Camera can be either set to any arbitrary angle, or snapped to a predefined set of angles.

- **Time to apply force, sec** – if camera is rotated by hands and fixed in a new position more then specified time, the new position is remembered as a target position. To disable this feature for any axis, set it to 0.
- **Hold menu button to adjust position** – if this option is selected, a new target position is applied only if user holds the menu button when positioning camera by hands.
- **When axis is forced to new position** – select one of the options
  - **Do not snap** – set the current position as a new target position
  - **Snap to 45/90/180º** - find the nearest angle that is divisible by the selected angle and set it as a new target.
- **Fine adjust position near snap angle** – if the option "snap to angle" is selected, you still can fine adjust angle in a small area of the "snap" angle. To snap any axis again, just tilt it to a bigger relative angle.
- **Ignore RC limits and Euler order** – for example, if you want to tilt ROLL axis to 90º, it is not possible if Euler order is set to "PITCH-ROLL-YAW", because ROLL angle is limited there. Enabling this option you let system to automatically switch to a new Euler order, where this position is allowed. The same for the RC limits: if ROLL limits are set to ±45º, this option cancels limits.
- **Snap at system start-up** – combined with the option "When axis is forced to new position", it allows to snap gimbal to predefined angles if it is held near these angles when system is powered ON. Note that it does not relate to the option "Time to apply force, sec" – i,e, it may be disabled or enabled, no matter.

Extra buttons *(frw.ver. 2.68b4+)*

Up to five extra buttons can be connected to the digital inputs of the controller: AUX1..3 and all RC inputs. Button is connected between the selected input and the GND; an external pull-up resistor is not required – pin is pulled up internally. Button type may may be either a momentary push button, or a latching button, depending on a required behavior.

Extra buttons control the values of virtual RC channels named "**EXTRA_BUTTON_1..5**" from the low level

(button is released) to the high level (button is pressed).

Configuration

- **Pin** – select the input where button is connected. It should be not occupied by other functions!
- **Latching** – for a momentary push button, a latching mode can be enabled: each click toggles the state between min and max.
- **Inverted** – if enabled, the signal is inverted (i,e, on press it outputs low signal, on release - high).

How to use extra buttons

Once the extra button's state is mapped to the RC virtual channels, it can be used in every place where RC channels are assigned to functions.

Extra buttons can be combined with the "Step signal source" mechanism or the "Adjustable variables" mechanism, to control the parameters of a gimbal or to trigger actions. For example, it is possible to assign the button to the "Step signal source" mechanism (SSS), then assign SSS to the "Adjustable variable(s)". Each click on the button will change the value of any parameter (or multiple parameters) in steps from min. to max., then repeat in a cycle.

SSS modification allows to count a number of clicks and set the value correspondingly, i,e, 1 click for the starting value and 5 clicks for the ending value in a range. In a combination with the trigger-type adjustable variables, it can be used to reproduce a behavior of the regular "mode" button by assigning actions to each of five levels of the signal.

Linking the *EXTRA_BUTTON_x* channel directly to the adjustable variable that controls the mode of operation (like "Follow" mode), it can be used to switch between modes w/out need of switching profiles.

Serial link over CAN *(frw.ver. 2.71b6+)*

For the controllers equiped with the CAN bus, it allows to pass additional serial data via gimbal's CAN bus between modules that has hardware serial port and support this functionality[1]: main controller, GPS_IMU, CAN_IMU[2].

[1] Ensure that modules are loaded with the latest firmware in order to support this functionality.
[2] CAN_IMU serial port UART1 and UART2 Rx, Tx are shared with other pins, check a pin-out on the product page.

The possible usage scenarious for this function:

- GPS_IMU used as a main IMU and located on the camera platform, communicates with the remote GNSS-receiver, located somewhere on the frame;
- External host controller connects to the main controller's UART port and communicates with the GPS_IMU Serial API;
- External host controller communicates with the camera's serial interface (like Sony LANC), connecting to the main controller's serial port and routing serial data transparently via CAN bus to the CAN_IMU's serial port, where the camera interface is connected.
- Arbitrary host controller, located on the camera platform, connects to the CAN_IMU's serial port and gets access to the SBGC32 Serial API, allowing recording realtime data or controlling gimbal.

Serial Link configuration is done by the *configuration slots*. You can set up up to 5 routes.

- **Serial port** - select a hardware serial port on the device to be the 1st end of the route. For the SBGC32 local ports they should be enabled in the "Hardware" tab or "RC" tab (for the "RC_Serial" port).
- **Baud rate** - baud rate parameter of the selected serial port. For the SBGC32 local ports, this parameter overrides the value specified in the "Hardware" tab.
- **Parity** – No, Even or Odd.
- **Function** - Which function should be assigned to the 2nd end of the route.
  - SBGC32 Serial API - get access to the main controller's Serial API
  - GPS_IMU Serial API – get access to the GPS_IMU's Serial API
  - Transparent serial - for sending arbitrary data. In this function, the target port should be specified and the 2nd end should be configured in the additional slot
  - GPS_IMU GNSS receiver (primary / secondary / RTK rover / RTK moving base) – allows GPS_IMU to use external receiver (check the GPS_IMU reference manuals for details)
  - *Virtual ports (to be implemented later)* – as a glue to assign other functions available in the controller. Virtual ports can be selected later in serial port assignment drop-downs in other tabs.
- **Target port** – for the function "Transparent serial", specifies the 2nd end of the route. It should be also configured in the additional slot as the 1st end, having this port as 2nd end, forming two (A → B) and (B → A) pairs.
- **Timeout, ms** - How long to wait serial data on the hardware serial port to fill up 9-byte packets before sending them to the bus. Value 0 - send immediately on receiving each single byte. Recommended value is 10ms to have a good balance between delays and throughout efficiency.

Retracted position *(frw.ver. 2.72.0)*

Allows to setup a retracted position and move camera there by the corresponding menu command. Gimbal uses "servo" mode for that, so it can firmly hold this position. Another option will be turning motors OFF after reaching it.

The following parameters are available:

- **Motor angles** – angle of motors (relative to the calibrated "Encoder offset") defining the retracted position
- **Move by the shortest path if allowed** – if the software limits are not defined or defined for multiple turns, gimbal choses the shortest path if the distance to the target exceeds 180 degrees. If this option is disabled, the motor always goes by the long path, making multiple turns, if needed.
- **Disable motor after parking** – if enabled, turn motors OFF after several seconds passed since it reached the target position.
- **Allow RC control in retracted position** – by default, a remote control is disabled during this operation. But it can be re-enabled after reaching the target. Note that the gimbal still stays in servo mode, where the RC control is motor-related.

In order to return to normal operation, the "Move to home position" or "Move to startup position" menu command should be issued.

Options for monitoring the process from an external controller

SBGC32 controller can trigger an interrupt "On entering retracted position" on the selected pin, if enabled in "Service" – "Misc. settings". It can be used by an external controller to mechanically lock the motors. Also, the corresponding event is triggered, if the host controller subscribed on it using the Serial API's command CMD_DATA_STREAM_INTERVAL (see Serial API documentation for details). It triggers ON event when reaching the position, and OFF event when motors are disabled (after ~3 seconds).

Shake generator *(frw.ver. 2.73.1)*

It emulates a camera shake with a very flexible configuration of motion patterns. Axes PAN, TILT, ROLL are screen-related, unlike the Euler axes that are ground-related.

Shake generator can be started by a button in GUI, when switching to the particular profile, or by menu commands "Shake generator ON", "Shake generator OFF". Also, setting the "master gain" parameter to 0 disables it.

Parameters are profile-based, so you can have different shake patterns in different profiles. Also, all parameters are exposed as "adjustable variables" and can be linked to RC channels in order to adjust them in runtime.

Parameters available:

- **Amplitude** - defines the amplitude of a generated shake signal. Set to 0 to disable generation for this axis. WARNING: High amplitude and frequency may overload the motor; check its temperature after 5-10 minutes of work to prevent overheating.
- **Master gain** – a common multiplier which is applied to per-axis amplitudes.
- **Resonance effect, dB** - emulates the resonance effect (as if the camera were mounted on a spring suspension). Value 0 dB disables this effect.
- **Frequency shift, %** - shifts the frequency parameters for the selected axis (acts as a multiplier). It allows to have different speeds on different axes, for example smooth PAN shaking and sharp TILT shaking.
- **Dynamics** – defines the speed of movement in terms of frequency
  - **Base frequency, Hz** - defines the generated base frequency to start from
  - **Frequency range** – random variation of the base frequency: F = (F_base ... F_base * F_range)
  - **Resonance frequency, Hz** – works in conjunction with "resonance effect, dB"
- **Pause insertion** – helps to make the generated shake more natural, when there are periods where the camera is steady
  - **Period, ms** - shake signal is paused after the given intervals
  - **Balance, %** - how much the pause cycle relates to the generation cycle. The more value is, the shorter periods of generation.
  - **Attenuation, %** - how much the signal is attenuated during the 'pause' cycle. Value 0 - no attenuation, 100 - full attenuation (no signal at all)
  - **Randomness, %** - random variation of some pause parameters, 0-100%, helps the effect not look "synthetic"
  - **Phase variance, %** - shifts the pause insertion time for different axes. 0% - all axes are paused synchronously; 100% - different axes are paused at different time (selected randomly)

- **Enable in this profile** – generation is started at system start or when the profile is switched.

11. 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 check-box "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.

12. Adjustable Variables

SimpleBGC firmware supports not only the 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.

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.

Starting from the firmware version 2.62, the extended version of mapping of the signal source's values to the system parameter's values is available: by meaning of a look-up table (LUT) with the interpolation between nodes.

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

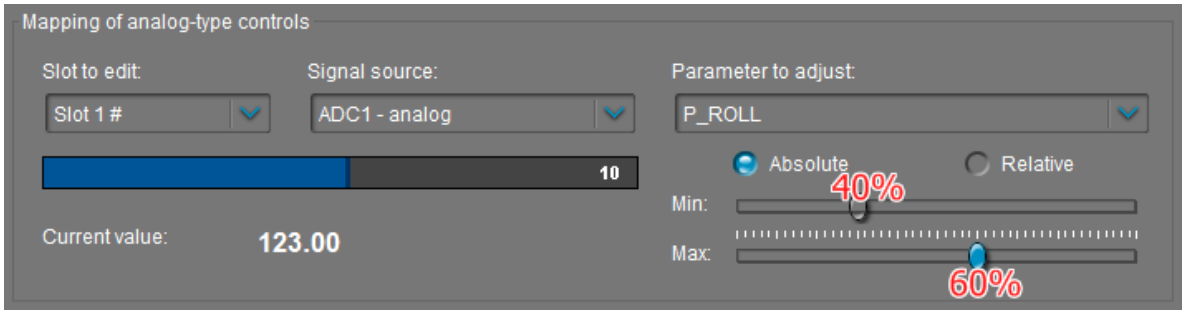
Setting control of the Trigger type

- 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.
- Select the **type of variable** *(firmware ver. 2.62+)*.
  - **Absolute:** the resulting value of the variable is set according to the RC signal, mapped to a range between the configured Min. and Max. points. If variable is linked to the GUI control, it's value is ignored.
    This is default type for firmware ver. before 2.62x
  - **Relative:** the resulting value of the variable is based on it's original value (configured in the GUI) and the multiplier, that depends on the Min., Max. sliders and the RC signal, mapped between them using logarithmic scale (which gives 1.0 rate in the middle and 0.1x and 10x at the borders).
- 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.

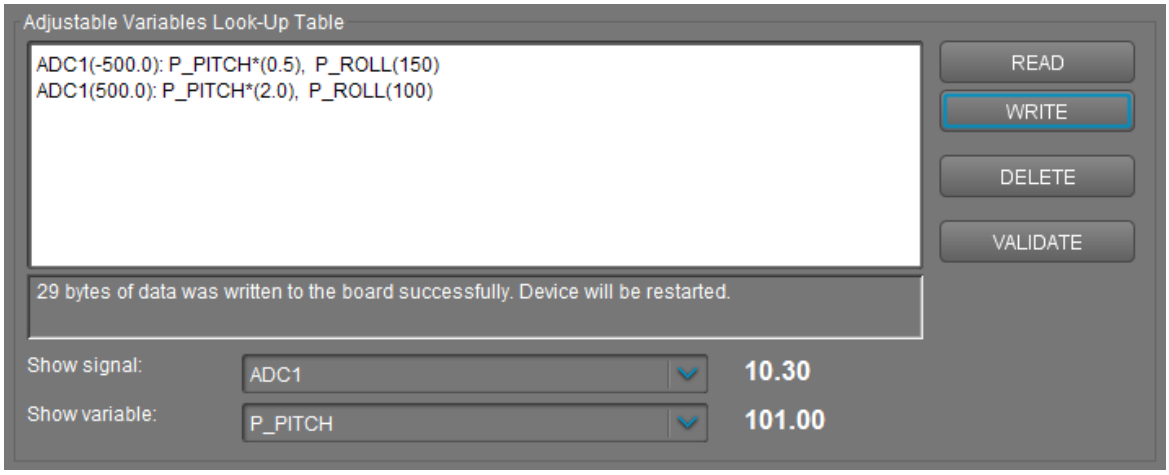
There is a possibility to invert a control, so that when a signal goes up, a variable goes down. To achieve this set Min. slider greater than Max slider.

You may notice that Min. and Max. sliders extend the range of a variable to ±10%. it's 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.

Setting up a dependency using look-up table

Starting from the firmware version 2.62, there is a second option to control system parameters: by using a look-up table (LUT), that defines a relationship between the signal source and the parameter. The advantages of this approach compared to the "Analog" type of control, are better precision – values are set by numbers instead of sliders, - and a possibility to set more complicated curves by using multiple points.



The LUT definition has the following format:

SOURCE_1(VALUE_1): PARAMETER_1(VALUE_1), ... , PARAMETER_K(VALUE_1)
...
SOURCE_1(VALUE_N): PARAMETER_1(VALUE_N), ... , PARAMETER_K(VALUE_N)
SOURCE_2(VALUE_1): ...PARAMETERS...
...
SOURCE_2(VALUE_M): ...PARAMETERS...

Each row defines a single point on the curve. It should be defined at least two points for each signal source, to let to interpolate between them. The set of parameters and their order should exactly match in all points of the particular source. If the name of the parameter is accompanied by the asterisk character "*", it acts as a multiplier: original value of the parameter, set in the GUI, is multiplied by the value, estimated from the LUT. Multipliers are interpolated between points by the exponential law, that better reflects their nature.

Example:

ADC1(-500): P_ROLL(100), P_PITCH(100)
ADC1(500): P_ROLL(150), P_PITCH(150)
ADC2(-500): FOLLOW_SPEED_PITCH*(0.5), FOLLOW_SPEED_YAW*(0.5)
ADC2(500): FOLLOW_SPEED_PITCH*(2.0), FOLLOW_SPEED_YAW*(2.0)

In this example, when the signal on the ADC1 input changes from 0 to 3.3V (that corresponds to a full range of potentiometer), the value of parameters P_ROLL and P_PITCH changes from 100 to 150. When the signal on the ADC2 input changes in the same range, the value of parameters FOLLOW_SPEED_PITCH and FOLLOW_SPEED_YAW starts with the half of its original value, set in the GUI, and ends up with the double of original value. Note that in a neutral point, the value of the parameter exactly matches the original value, due to the exponential low of interpolation for multipliers.

The number of sources and points is not limited, but it should be taken into account the size of LUT in memory (both RAM and EEPROM), that is shared between other extra functions of firmware, that consume memory, like scripts, Rx/Tx UART buffers and so on.

To write table to a device and read it back, use dedicated "READ" and "WRITE" buttons. The controller should be restarted to apply changes.

To check how LUT is processed in real-time, you can monitor any signal source and any variable, by selecting them from the drop-down lists 'Show signal' and 'Show variable'. Additionally, you can use them as a reference for the names of source and variables.

Signal sources and their ranges

- All RC signal sources, ranges from -500 to 500:
  `RC_ROLL_PWM, RC_PITCH_PWM, RC_YAW_PWM, FC_ROLL_PWM, FC_PITCH_PWM, ADC1...3, RC_VIRT_CH1...19, API_VIRT_CH_1...31.`
- Relative angles of all motors, expressed by SIN and COS. Ranges from -1.0 to 1.0:
  `SIN_INNER_MOTOR, COS_INNER_MOTOR,`
  `SIN_MIDDLE_MOTOR, COS_MIDDLE_MOTOR,`
  `SIN_OUTER_MOTOR, COS_OUTER_MOTOR.`
- Absolute values of SIN, COS of motor's relative angles. Ranges from 0.0 to 1.0:
  `ABS_SIN_INNER_MOTOR, ABS_COS_INNER_MOTOR,`
  `ABS_SIN_MIDDLE_MOTOR, ABS_COS_MIDDLE_MOTOR,`
  `ABS_SIN_OUTER_MOTOR, ABS_COS_OUTER_MOTOR.`
- Momentum of inertia of mechanical system linked to each motor*. Ranges from 0 to 32767.
  `MOMENTUM_ROLL, MOMENTUM_PITCH, MOMENTUM_YAW.`

* Supported in the “Extended” version of controllers only.

In future, the list of sources may be expanded. You can find all supported sources in the drop-down list “Adjustable variables Look-Up Table” - “Show signal”.

Table 1. Decoding of names of controlled variables

Parameter's name	Frw. ver	Min	Max	Description
P_ROLL, P_PITCH, P_YAW		0	255	Parameter of 'P' PID-controller
I_ROLL, I_PITCH, I_YAW		0	255	Parameter of 'I' PID-controller multiplied by 100
D_ROLL, D_PITCH, D_YAW		0	255	Parameter of 'D' PID-controller
POWER_ROLL, POWER_PITCH, POWER_YAW		0	255	Parameter 'POWER'
ACC_LIMITER		0	1275	Acceleration limiter- unit of measurement: $1°/s^2$
FOLLOW_SPEED_ROLL, FOLLOW_SPEED_PITCH, FOLLOW_SPEED_YAW		0	255	The speed of movement in the mode "Follow"

FOLLOW_LPF_ROLL, FOLLOW_LPF_PITCH, FOLLOW_LPF_YAW		0	15	Smoothing of operation in the mode "Follow"
RC_SPEED_ROLL, RC_SPEED_PITCH, RC_SPEED_YAW		0	255	Speed of movement when operating from the RC transmitter
RC_LPF_ROLL, RC_LPF_PITCH, RC_LPF_YAW		0	15 (255)*	Smoothing of operation from the RC transmitter
RC_TRIM_ROLL, RC_TRIM_PITCH, RC_TRIM_YAW		-127	127	Neutral point trimming for channels controlling the camera by ROLL, PITCH, YAW in the speed mode
RC_DEADBAND		0	255	The dead-band of the RC signal for the camera control channels in the speed mode
RC_EXPO_RATE		0	100	Degree of exponential curve depth for the RC signal
FOLLOW_PITCH		0	1	Follow mode by the PITCH, ROLL axes: 0 - off, 1* - follow PITCH and optionally ROLL **frw. ver. 2.65b3*
FOLLOW_YAW_PITCH		0	2	Follow mode by all axes: 0 - off, 1 – Follow YAW 2* - Follow YAW, PITCH and optionally ROLL **frw. ver. 2.65b3*
FOLLOW_DEADBAND		0	255	The dead-band for the deflection angle of the frame in the Follow mode- unit of measurement: 0.1 degree
FOLLOW_EXPO_RATE		0	100	Degree of the exponential curve depth for the Follow mode
FOLLOW_ROLL_MIX_START		0	90	The starting point of the zone transition to the Follow mode, degrees
FOLLOW_ROLL_MIX_RANGE		0	90	The length of the zone transition to the Follow mode, degrees
GYRO_TRUST		0	255	Trust to gyroscope compared to accelerometer
FRAME_HEADING_ANLGE		-1800	1800	The angle of a frame for YAW axis (azimuth). If it's known, it can help to eliminate gyro drift by the YAW axis. For example, gimbal is installed on a tripod and azimuth is frozen (0), or installed on a crane, and azimuth is known from it's controller. The conditions when a gyro drift can be removed: encoders are installed on all 3 axes, or 2nd IMU is installed directly on a frame and YAW encoder is used. Note: this angle is always expressed in the Euler

				order PITCH-ROLL-YAW, regardless of the Euler order configured in the RC settings. This correction has a priority compared to the correction received from the external IMU, if it's connected. *Units: 0.1 degrees*
GYRO_HEADING_CORRECTION		-20000	20000	The offset for gyro YAW axis, to eliminate gyro drift manually by the operator who can observe a picture from a camera. *Units: 0.001 sensor units*
ACC_LIMITER_ROLL ACC_LIMITER_PITCH ACC_LIMITER_YAW		0	1275	Acceleration limiter, separate for each axis. *Units: 1°/sec²*
PID_GAIN_ROLL PID_GAIN_PITCH_ PID_GAIN_YAW		0	255	Additional PID gain. *Units: 0...255 corresponds to 0.1...5.2 gain range, 45 corresponds to 1.0 gain.*
LPF_FREQ_ROLL LPF_FREQ_PITCH LPF_FREQ_YAW		10	400	Low-pass filter cut-off frequency, Hz
TIMELAPSE_TIME		1	3600	The time of time-lapse routine, in seconds.
MAV_CTRL_MODE		0	2	MavLink control mode
H_CORR_FACTOR *(frw.ver. 2.68b7)*	*2.68b7*	0	255	Heading correction factor from external reference
SW_LIM_MIN.ROLL SW_LIM_MAX.ROLL SW_LIM_MIN.PITCH SW_LIM_MAX.PITCH SW_LIM_MIN.YAW SW_LIM_MAX.YAW	*2.68b8*	-3600	3600	Sowtware limits for each motor, degrees (encoder firmware only)
FOLLOW_RANGE.ROLL FOLLOW_RANGE.PITCH FOLLOW_RANGE.YAW	*2.68b9*	0	255	Follow range, degrees
AUTO_PID_TARGET	*2.68b9*	0	255	Stability-precision slider for automatic PID tuing algorithm
RC_MODE_.ROLL RC_MODE_.PITCH RC_MODE_.YAW	*2.69b3*			0 – ANGLE 1 – SPEED 2 – TRACKING
EULER_ORDER	*2.69b3*			0 – PITCH-ROLL-YAW 1 – ROLL-PITCH-YAW 2 – PITCH(M)-ROLL-YAW(M) 3 – ROLL-PITCH(M)-YAW(M) 4 – YAW-ROLL-PITCH

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:

- Disconnect other devices on UART ports, if present.
- For non-windows operating system, additional steps may be required. See notes at the end of this section.
- For “Tiny Rev. A” 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. The "Tiny Rev. B" board does not require DFU drivers.

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)
   Note: for the board versions "Extended", "Tiny Rev.B" and "Pro", jumper is replaced by the flat momentary push button.
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 the most of Basecam controllers
   - **v.3.x (32bit) through USB in DFU mode** – for the “Tiny rev. A” board, or "Tiny Rev. B" in direct USB connection mode (if configured by jumpers). 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 a firmware file. For a factory-made gimbals, it is advised to set the option "Show compatible firmwares only", to hide firmwares that are not compatible with you device.
3. Press "FLASH" button and wait for process to finish.

NOTE: If you want to get access to beta versions of firmware, intended for a testing of new functions, you can enable the flag "Check for beta versions". Please note that beta version may contain critical issues. Also, a downgrading to a previous stable version may cause a loss of all settings. So, it's strictly advised to make a backup of your configuration using the menu "Board" – "Backup manager..".

Upgrading under Mac OS and Linux

Starting from 2.42b7 it's 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.

IMPORTANT NOTE: You need to give execute permissions to this tool. Open the Terminal (Application/Utilities/Terminal), type "chmod u+x", space, then find GUI folder in the Finder, open "bin" directory, select **stm32ld_mac** file and drag-and-drop it inside the Terminal window - full path to a file will be inserted in the command line. Press "Enter" to execute this command.

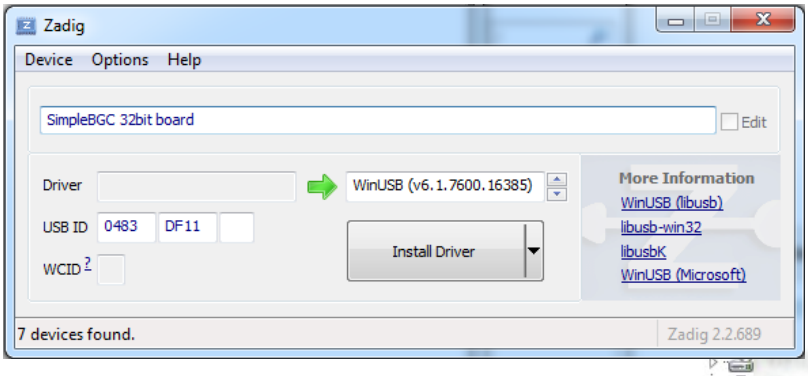
FOR EXPERIENCED USERS: If the tool failed to run under your OS, you can compile it from sources (located in the 'SimpleBGC_GUI/bin/stm32ld-src' folder). Place the result in the 'SimpleBGC_GUI/bin' folder, renaming it to 'stm32ld_mac' for Mac OS, 'stm32ld_linux' for Linux family, and 'stm32ld' for any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

Installing DFU device driver

This driver is required only for "Tiny Rev. A" 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/
2. Run Zadig. In the "Device" menu select "Load preset device.."
3. Select file "SimpleBGC_GUI/conf/SimpleBGC 32bit board.cfg"
4. Install driver **WinUSB**



To check that driver is installed properly:

© Basecamelectronics® 2013-2024

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, it's not (permanently) fatal for your board (it's 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 regular 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 can choose to receive only stable versions issued for your board, or additionally receive beta versions, intended for a testing of new functions.

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 or Wi-Fi 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 is safe to update firmware with the battery connected, but disconnecting the battery is a good practice.

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

A: **it's 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. An URL for downloading a version matched GUI is generally provided in the description of the firmware.

Q: I got an error during uploading: "CreateProcess error=14001" (frw. ver. < 2.67)

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

Q: I got an error "Flash tool execution failed" and string "Cannot open the com port, the port may be used by another application" in the details (frw. ver. <2.67)

A: It may be because COM port number is greater than 99. Go to the Windows device configuration utility, open "Serial ports" settings, and rename port, giving it number below 100. Also ensure that there are no other copies of GUI tool running and having opened connection.

Q: I got an error "No answer from bootloader"

A: Check that there are no external devices connected to other UART ports of the controller (like wireless module, external IMU), or they are not active, i,e, are not sending data to the port. Otherwise, it can confuse bootloader, when it senses any data on incorrect port.

14. System Analysis Tool

This tool lets you capture 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, it's 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.

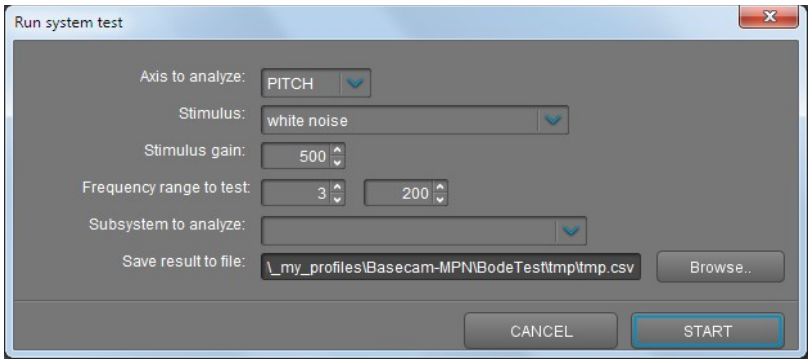
Open-loop vs closed-loop test

In most cases, we are interested in the “open-loop” system response. But if we have an integrator inside the tested system, we find that it has big gain for low frequencies that can lead to over-saturation of an output or makes system go outside of its working range (in other words it means that the gimbal can't make infinite rotation because of physical limits). To prevent this, we keep applying some portion of feedback even in "Open-loop" tests.

If it still does not help, the solution is to run system in a closed-loop mode, mixing stimulus signal with the feedback signal. But at low frequencies, closed-loop system effectively removes any disturbances, making input equals to output. That is the reason why the closed-loop mode is not perfect for analysis systems near low-frequencies.

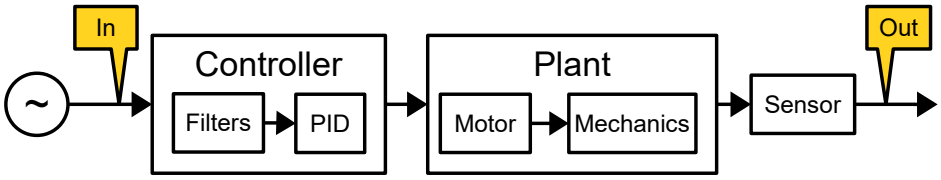
Starting test

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

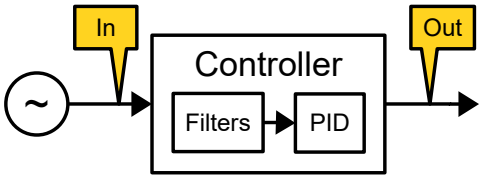


1. Select the axis to test
2. Select stimulus to feed to the input of a system
   - **White noise** (recommended): this signal contains the full set of frequencies distributed uniformly with slight amplification of the high-frequency band, where system response normally decreases
   - **Sine sweep:** signal with constant amplitude and frequency that goes in a specified frequency range.
   - **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. Select **frequency range**. Frequencies outside this range are attenuated. Use values 1-5 Hz for low end and 200-500 Hz for high end.

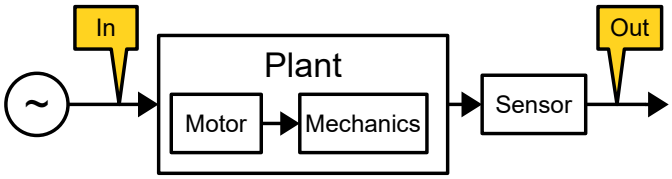
5. Select system to test:
   - **Controller + plant:** input is passed to the PID controller, output is read from the rate sensor (gyroscope).



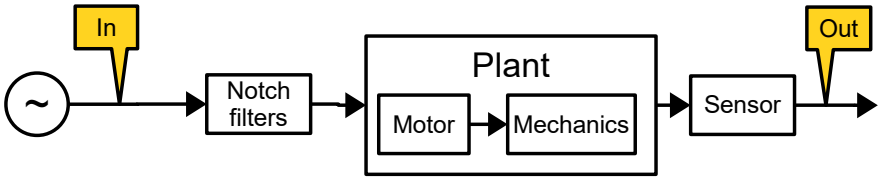
   - **Controller only**: input and output obtained from the PID controller. In this test, motors are disabled and test is not visible. Don't set too big gain (to prevent a saturation inside controller).



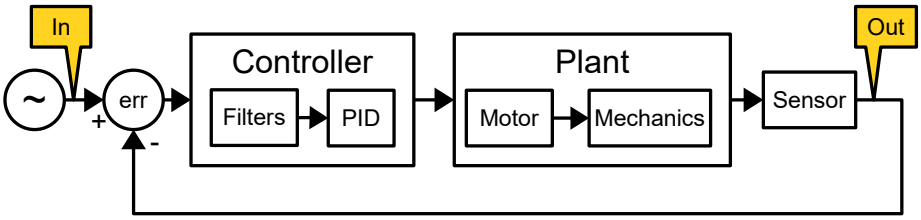
   - **Plant only**: input is passed to motor, output is read from gyroscope sensor. Again, be careful with the gain parameter.



   - **Plant + notch filters**: The same as above but with the digital filters applied.



   - **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). Using this test, you can estimate the gimbal's reaction on a control signal: frequency range, overshoot and phase delay on given frequency.



6. Place gimbal on a steady support, power motors ON and begin test. It's 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.

If any motor starts to spin, most probably it's caused by wrong "Invert" setting. Run auto-calibration of number of poles to detect proper inversion.

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:

See a definition of Bode plot here: https://en.wikibooks.org/wiki/Control_Systems/Bode_Plots

You can analyze the obtained data (input and output samples in rows in CSV format) by more powerful tools like Matlab or similar programs. There are wide sets of utilities from system identification to system tuning, but engineering skills are required to make use of them.

Reading and understanding the test results

You need to understand the basics of system analysis before reading a Bode plot.

In few words, on the "*Plant*" response plot you can see the response of motors and mechanics, and find potential mechanical resonances. Running the "*Plant + notch filters*" test, you can see the effect of Notch filters and compare it with the unfiltered response on the same chart.

On the "*Controller+Plant*" response graph you can evaluate the performance of the stabilization. There are several techniquies to do this:

- **Analyzing phase and gain margins:** find the gain margin (distance to 0 dB in amplitude response at frequency where phase crosses -180 degrees) 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.

- **Analazyng sensitivity function** (displayed in GUI version 2.68b7+) is more informative and more powerful tool. It is described more detailed in the section "Automatic tuning using "Analyze" tool in the GUI" - "Evaluation of the quality of stabilization".

On the "*Overall system*" response graph you can find how effectively a system tracks its input signal on different frequencies. You can estimate max. frequencies that system is able to compensate, where gain is above -3dB 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 amplifies signal on its input and see the contribution of digital low-pass filters.

15. User-written scripts

With a special scripting language user can create a program to control a gimbal. The program is loaded into the controller and is executed by a command from the RC or menu button. Language reference can be downloaded by the link: http://www.basecamelectronics.com/files/v3/SimpleBGC_Scripting_Language_eng.pdf

There is a simple text editor in the *Scripting* tab with syntax checking. Its main functions are:

Saving and loading of files
Scripts are stored in text files. You can use any text editor to edit them.

Syntax checking
After loading a file, application checks the syntax. Errors found are highlighted in red and a short message explaining the reason, is provided. Also, the syntax will be checked by clicking VALIDATE button and when uploading the script to the controller.

Uploading scripts to the controller
There are 5 slots allocated that can hold up to 5 scenarios, the overall size (after compilation) of 27 kilobytes. Script size is displayed near the slot number. Empty slots are marked as <empty>. To delete a script, delete all the text in text editor and write it into the slot you want to clear.

Restore script from the board
You can download the script from the board for editing. But at the same time, as a result of decompilation, you will lose all comments and original formatting. Therefore it is recommended to store scripts in text files.

Running scripts
RUN button will start the script located in the selected slot. If the text in the script editor window corresponds to the contents of the slot, the current line of the program is highlighted in the process of execution. This is useful for monitoring and debugging. You can stop the script at any time by pressing *STOP* button

Other ways to run the script:

1. Assign command *Run script from slot 1..5* to menu button in the tab *Service*;
2. Assign command *Run script from slot 1..5* to the CMD channel of receiver in the tab *RC*;
3. Assign command *Run script from slot 1..5* to any control channel in the group *Trigger-type controls* in the tab *Adjustable Variables;*
4. Send the command CMD_RUN_SCRIPT through the Serial API.

16. Encoders

The *encoder* is a rotary position sensors, that provide very precise information about motor's shaft rotation. This kind of sensor gives some advantages for stabilizer system

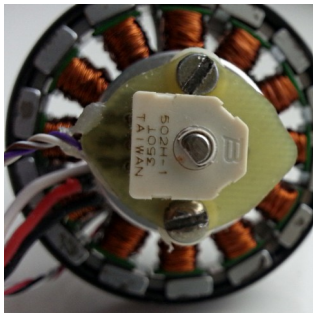
There are 2 versions of firmware exist: regular and extended. The difference is how they support encoders. In the **regular version,** only single encoder is supported, installed on the motor that drives the YAW axis. The motor control algorithm is not changed. In the **encoder-enabled** version, all three motors should be equipped by encoders. The motor control algorithm is changed to a more advanced field-oriented control (FOC).

Encoders in the regular version of firmware

Advantages of using encoder in the regular version of firmware:

- Precise work in the "Follow mode": the target angle will be not lost even in the case of loosing synchronization in a motor, that is important in the aerial use of a gimbal.
- Allow to install frame IMU in the "above YAW" position and to correct the cross-IMU gyro drift most effective way.
- Allows to get correction from an external autopilot or a high-grade IMU sensor installed on a frame.

The most simple type is **analog**. It's recommended to use a special type of potentiometer with the linear characteristic and low friction. Some magnetic encoders also have an analog output mode. Though analog encoders handles infinite 360 degrees of mechanical rotation, actual working range is limited to the value less than 360 and encoder should mounted such way to not go outside it.



Starting from the firmware ver. 2.60, **PWM** and **I2C** encoders are supported (all models that are supported by the "encoder" version of firmware and have I2C or PWM interface). Compared to the analog type, these interfaces are less sensitivity to a noise and support infinite 360 degrees rotation.

Connecting encoder

Analog type

For the potentiometer type of an encoder, the connection is simple: just connect 3.3V and GND terminals to its side outputs, and connect the central output to any of A1..A3 inputs. For the other types with the analog output, connect power according to manufacturer's specifications, and its analog output to A1..A3. Note that supported voltage range is 0..3.3V. Do not use encoders that exceed this range!

PWM and I2C type

Refer to the manual for the extended version to find all supported models and how to make a connection: www.basecamelectronics.com/encoders/

Configuring encoder

- **Type** – chose a model of encoder and it's interface
- **Input** – chose a port where encoder is connected, if applicable for a selected model
- **Gearing ratio** – it used mostly for the analog type of encoder. It defines a mapping between the voltage on the analog input and the angle. Value 1.0 corresponds to 0..3.3v → 0..360 degree of rotation. To estimate proper gearing ratio for your encoder, use GUI as follows:

  White arrow – shows the angle of motor relative to frame
  Compass arrow – shows the azimuth of camera in space

  1. If second (frame) IMU is connected – disable it temporarily (Hardware – Frame IMU sensor – Disabled). Turn motors OFF.
  2. Enter initial values: gearing ratio = 1.0, offset = 1 (or any non-zero value). Write parameters to the board. If encoder is connected properly, the white arrow will start to display rotations of the YAW motor.
  3. Turn the *frame* that way that the white arrow matches the compass arrow. Fix the frame in this position. Then rotate the *camera* by YAW axis only, and check if the white arrow moves exactly like the compass arrow. If it moves in opposite direction, mark "Inverse" check-box. If the white arrow moves faster that the compass arrow, decrease gearing ratio. In opposite case, increase it.

  For other types of interface, angle is obtained in digital form so gearing ratio should be set to 1.0, excepting the case when real mechanical gearing is used. In this case, specify the known ratio of the geared transmission. Use the method described above, to check that it's set correctly.
- **Offset – set the zero angle.** Move the frame in "normal" position* and press the **"CALIBRATE"** button. A new value for this parameter will be set and displayed in the GUI.

* Normal position – position, where the frame points the same direction as the camera. If second IMU is installed, it's axes exactly matches the main IMU's axes.

When the "Offset" parameter is set to non-zero value, encoder's data is used by the firmware in calculations and is displayed in the GUI: "Monitoring" → "ENC_RAW_Y" displays raw data, white arrow in the angle panel displays the relative motor angle.

Encoders in the special "encoder" version of firmware

The "encoder" version of firmware requires to install a precise absolute encoder on **each motor**. It supports analog, I2C, SPI, PWM interfaces and various models of encoders. Advantages of the extended version:

- No need to install second (frame) IMU;
- Improved algorithm of motor control - Field-Oriented Control. It is free of synchronization loss, power consumption is much less (motor is not fully powered permanently like in the regular firmware), with the proper winding can provide higher instant torque;
- System has the full information about the frame attitude relative to the camera platform, that allows to solve tasks like GPS-aided target tracking, IMU correction from external high-grade IMUs or flight controllers.

Because of a high complexity of installing and tuning encoders, we do not consider them in this manual.
All information you can find on this page: www.basecamelectronics.com/encoders/
In the factory-tuned gimbal, it is not recommended to change parameters related to the encoders.

17. Magnetometer sensor

A magnetometer helps avoid horizontal drift of the gyroscope, the same way as an accelerometer does with vertical drift. But the use of a magnetometer is not always justified, since the process of measuring Earth's magnetic field is much less accurate and reliable than measuring gravitational acceleration when using the accelerometer. Furthermore, with installation of a magnetic sensor on the gimbal comes the need to exclude the impact of the distortions caused by structural metal elements, permanent magnets and motor windings, which may further complicate the process and reduce the sensor's effectiveness in application.

The use of the magnetometer is justified when shooting lengthy scenes in the Lock mode to avoid the direction drift or in order to determine the exact orientation of the camera in the 3D space based on all three coordinates, so as to allow for external GPS-aimed control.

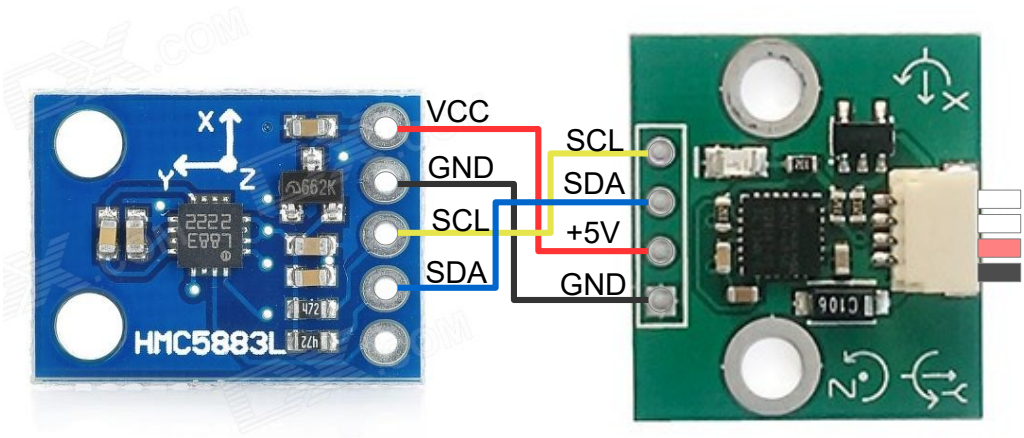
> **NOTE:** in buildings, the Earth's magnetic field can be significantly declined depending on a position. It leads to an incorrect work of a magnetometer sensor. As a result, YAW axis may drift unpredictable. It's recommended to switch OFF the magnetometer when working in studio conditions, if such problem is detected. It can be done by adding a latching switch on the +5V power line. Also, a re-calibration of a magnetometer in each new place is a good practice.

Supported sensors and connectivity

The HMC5883L and HMC5983 sensors are currently supported. A wide variety of modules using this sensor are available for purchase. When selecting a sensor, bear in mind the following requirements:

- It must support +5V
- It must be compatible with 3.3V logic (no LLC 5V for Arduino).
- There should be no pull-ups on the SDA and SCL lines, or they should be connected to the embedded +3.3V voltage regulator, rather then to the +5V power line.

The GY-273 and GY-282 modules meet these requirements. As an example, the module's connection to the main IMU is shown:



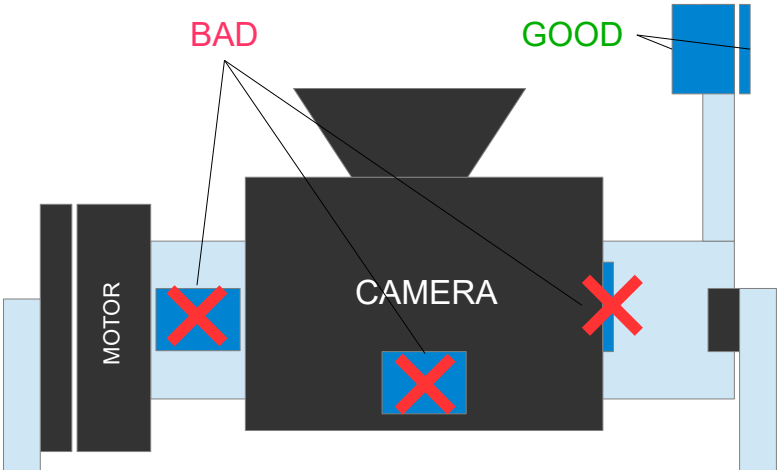
Magnetometer connection

For the GY-282, don't forget to select the I2C mode by soldering a jumper.

Installation on the gimbal

The sensor is mounted on the same platform as the camera. The axes may be oriented randomly, but it is important to ensure that the axes of the magnetometer are positioned parallel to the axes of the main IMU sensor. If the gimbal or the camera contain metal parts with ferromagnetic properties (iron, steel, etc.), the

sensor must be mounted as far from them as possible. The sensor must also be mounted as far as possible from the motors. For instance, the sensor can be offset on a 10-20 cm boom.



Magnetometer placement on the gimbal

it's possible to mount a magnetometer on the same position where the 2nd frame IMU is mounted. In this case it will correct the heading of the 2nd IMU, than the correction will be translated to the main IMU, as described in the section The problem of mutual azimuth drifts of two IMU sensors.

Setting up the magnetometer in the GUI

Once the magnetometer is properly connected to the I2C bus, and 2.50b4 or greater firmware is loaded, several new groups will be displayed in the "Hardware" tab of the GUI, enabling the magnetometer's calibration and setup.
First, specify its mount position (frame or camera).

Position of the axes

To ensure its proper operation, the position in which the sensor is mounted on the gimbal must be specified. First, if the sensor's axes are not indicated on the module, determine their direction by using the key spot on the sensor chip:

In the Axis TOP parameter, specify which axis is aiming up. In the Axis RIGHT parameter, specify which axis is aiming right, as viewed in the direction of the shooting (as shown in the example, Axis TOP = Z, RIGHT = Y). Enter the settings in the controller and wait for it to reboot.

Magnetometer calibration

> **NOTE:** For proper calibration, the sensor must be mounted on the gimbal. Make sure that the “Magnetometer Trust” parameter is set to a value different from 0 (because the value 0 disables the magnetometer).

Calibration may be initiated from the GUI, or by the menu command (you can assign it to the menu button in the "Service" tab). It may be used to calibrate magnetometer in-the-field, without PC connection.

Press the **Calibrate...** button to launch Calibration Assistant. In the window that opens, press the **RESET** button to reset the existing calibration. Then press the **CALIBRATE** button. During the calibration process, controller gathers the measurements of the Earth's magnetic field in various directions. The progress indicator shows the percentage of collected data. Each new data point is marked by the single LED blink and a short beep sound (if motors are powered or buzzer is connected).

Note: for a firmware version 2.66 and above, it's required to rotate sensor in two directions by the 20..40 degrees in first 5 seconds just after calibration starts. After that data is collected as the sensor rotates, and the time of calibration is not limited – it stops after collecting the required number of points.

Then the collected data points are fitted by an ellipsoid. To achieve a high quality of calibrations, it is important to collect points that are well distributed over a sphere. The following algorithm is proposed:

- Point the sensor in the direction of the North or South (roughly)
- Make a full 360-degrees rotation over any of horizontal axes (PITCH, for example). You will collect 30-40% of points.
- Return the sensor to a normal position and turn it by 90 degrees (i.e. to the East or West)
- Make a full 360-degrees rotation over another axis (ROLL in our example). You will collect another 40% of points.
- Collect remaining data points by making random rotations.

When all data points are collected, a calculation will start automatically. It takes several seconds to finish. Once the calibration is complete, its result is automatically recorded in the EEPROM and applied. If the procedure is executed correctly, the extreme values displayed by the gauges, will be exactly ±1.0 on random rotations.

NOTE: it's required to rotate the whole gimbal, not only the camera. The reason is that the position of the camera relative to the motors (or ferromagnetic parts of a construction) may be changed and greatly distort the Earth's magnetic field. See the "Installation on the gimbal" section to avoid such problems.

Monitoring the magnetometer's effectiveness

The gauge indicator in the calibration window displays the absolute difference between the North direction measured by the magnetometer and the same angle measured by the gyroscope. As you know, in the short term, the gyroscope (along with a properly calibrated accelerometer) provides a very accurate reading. If the error remains in the green or yellow sectors during all the camera pans and tilts, this means that

the magnetometer is working correctly and can be used to correct the drift of the gyroscope. If the error increases significantly, the magnetometer should not be used. This may be due to a number of reasons:

1. The orientation of a magnetometer is set incorrectly. Check axis TOP, RIGHT settings;
2. Inaccurate calibration;
3. Improper installation, resulting in an impact of moving metal structures of the gimbal or motor magnets;

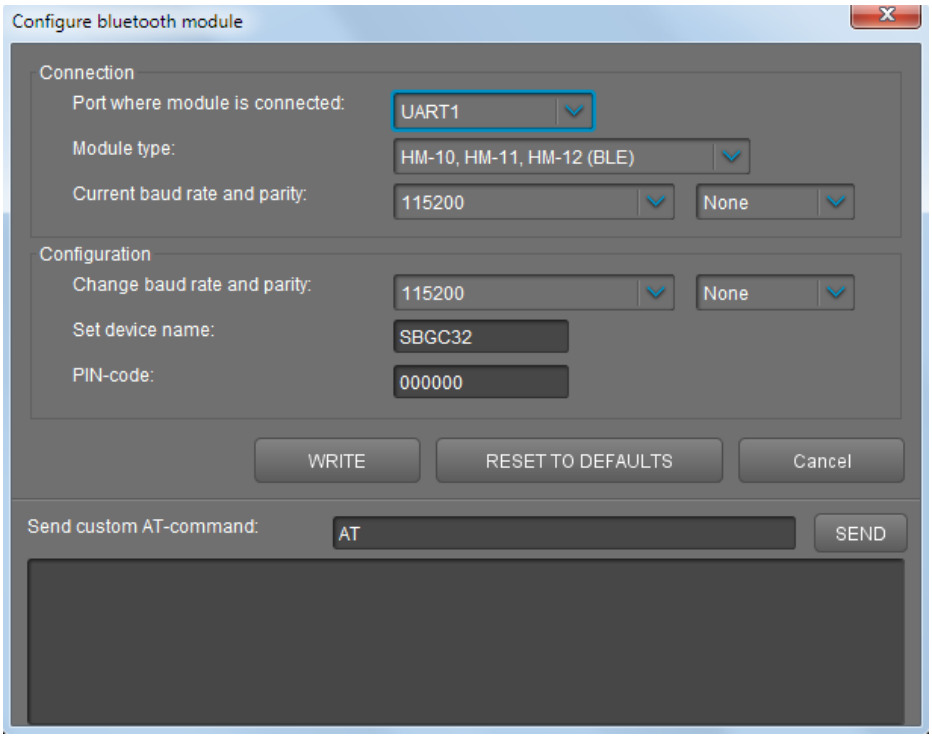
Other settings

- **Magnetometer trust** *(frw. ver. prior to 2.66)*
  "Hardware" – "IMU Sensor settings" – "Heading correction factor" *(frw.ver. 2.66+)*
  the greater the parameter, the stronger the correction of the current heading direction (YAW angle) by the magnetometer. If the result of the effectiveness test is good, you can use higher settings (50-100). Setting it at 0 disables the magnetometer. This parameter does not interfere with the "gyroscope trust" parameter (that is applied only to the accelerometer).
- **Magnetic declination** – Magnetic declination refers to the angle between the geographic and magnetic meridians on the Earth's surface where you are. The exact number can be found in

reference sources or on this map https://upload.wikimedia.org/wikipedia/commons/2/28/Mv-world.jpg This parameter is only required for systems that rely on the precise location of the geographical North (for example, for coordinating camera movements with a GPS when). In most cases, it can be set at 0.

18. Bluetooth module configuration

As it was mentioned in the "Computer Connection" section, for setting up a wireless connection via Bluetooth module it is necessary to configure it properly. To help with this configuration there is a configuration dialog box at GUI which can be run from the "Board → Configure Bluetooth..." menu:



Specify a port of module's connection, module's type and its current settings in the "Connection" section. Possible types of connection are:

UART1 is the main serial-port, present in every SimpleBGC controller, and it is marked as [5V, Gnd, Rx, Tx]. The module's connection is described in the Appendix B.

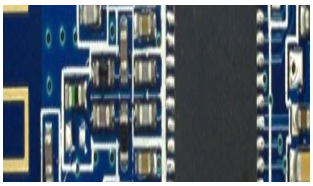
UART_RC is an additional serial-port combined with RC_ROLL (Rx) and RC_YAW (Tx) RC-inputs (see Appendix B). To activate it choose a "RC_ROLL pin mode = SBGC Serial 2nd UART" mode in the RC tab and leave RC_ROLL and RC_YAW physical inputs free. See Appendix B for reference.

UART2 is an additional port present only on some versions of the controller. It is absent on SimpleBGC32 "Regular" and SimpleBGC "Tiny".

Supported modules and their special characteristics

To be able to configure the module you should put it into AT-commands mode and set correct port speed and parity to which it is currently configured. Module's default settings are usually given at time of purchase, but also you can find them in the User Manual for every module.

HM-10, HM-11, HM-12



Is in the AT-commands mode unless connected to a wireless device.

Default settings: Baud: 9600, Parity: none, Data bits: 8, Stop bits: 1, PIN: 000000, Role: Slave

18 Bluetooth module configuration

HC-06, HC-04 and its clones

If module looks like in the picture, it's obviously HC-06 type.
Is in the AT-commands mode unless connected to a wireless device.

Default settings: Baud: 9600, Parity: none, Data bits: 8, Stop bits: 1, PIN: 1234, Role: Slave

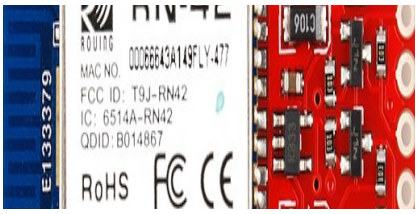
HC-05, HC-03

Looks similar to HC-06, but allows more customization.

To change for the AT-commands mode it is necessary to short its Vcc and Key inputs while the power is off, and then turn the power on. By doing so you turn it into AT-command mode and temporarily change the port's speed to 38400 regardless of the speed value you've set before.

Default settings: Baud: 38400, Parity: none, Data bits: 8, Stop bits: 1, PIN: 1234, Role: Slave

RN-41, RN-42 (BlueSMiRF)



Automatically being switched into AT-commands mode within 60 seconds after the power is turned on unless connected to a wireless device.

Default settings: Baud: 11520 Parity: none, Data bits: 8, Stop bits: 1, PIN: 1234, Role: Slave

Module configuring

1. Connect module with one of UART-ports and choose its type and current settings. You can send a test instruction to check the connection (as a rule it is “AT” command, and “D” for RN-42).
2. Reset the module to default settings by pressing the **RESET TO DEFAULTS** button
3. Set the appropriate port speed. For work via UART1 it should correspond the speed set for controller in GUI ("Hardware" tab → "Serial Speed"). For other ports the speed should be 115200.
4. Set up the Parity setting. If you are not going to update the firmware via Bluetooth-module, choose "None".
5. Set the name for the device that is visible for other devices during wireless connection setup.
6. Set the PIN which has to be entered for devices pairing.
7. Press the **WRITE** button. You will see the configuration results in the log.

You can manually send AT-commands for module configuration by clicking the **SEND** button. Command reference guide can be found in the datasheet for every module. Be careful as some commands can brick the module

19. Using an external IMU sensor to improve the precision of stabilization.

Starting from the firmware version 2.66, it is possible to connect an external high-grade GPS-aided IMU/AHRS sensor and use the information it provides, as a reference for a correction of the internal IMU sensor, or replace it in the attitude and heading estimation.

The set of supported sensors depends on the board's version:

- "Regular", "Tiny":
  - Ardupilot flight controller (FC) running on the hardware class Pixhawk PX4 or better, connected by the MavLink protocol;
  - Basecam GPS_IMU
- "Extended", "Pro":
  - MavLink-enabled FC
  - Basecam GPS_IMU
  - Vectornav VN100 (no GPS) or VN200 (with GPS)
  - Inertialsense µAHRS or µINS (both GPS-aided)

Installing on a gimbal

Several mounting positions are supported relative to the order of motors counting from the camera. All possible orientations relative to motor's axes are supported. Depending on the place of installations, you can select the most optimal mode of correction.

When selecting the mounting position, please consider several important conditions:

- Try to place sensor as far as possible from sources of magnetic fields, like motors. Try to avoid ferromagnetic materials and high-power cables in proximity to the sensor. The magnetometer that is used for the heading correction, is very sensitive to magnetic disturbances.
- A good reception of the GPS/GNSS signal is very important for a normal work of the attitude estimation algorithm. Place the antenna far from the circuits or devices generating EMI noise. Generally a patch antenna is used that is a directional antenna and should be oriented to the sky.
- Two axes of the external IMU should be aligned with the axes of the two nearest motors.
- For the optimal correction, it is required to place external IMU on the camera platform. But it may be hard to find a proper place that complies all conditions mentioned above. For this and other reasons like optimal cabling, clear sky visibility, magnetic interference - external IMU can be placed somewhere on the frame. It increases the requirements for a mechanical precision of the gimbal construction and a precision of encoders (linearity, calibration). Motor axes should be strictly orthogonal each other in home position* and do not deflect under load. Encoder offset should be calibrated very carefuly, giving the system a precise information about the home position.
  ** Configurations with tilted outer or middle motor are supported, too, if tilt angle is set correctly in the settings.*
- In systems that are not equipped with the encoders, it is impossible to get high precision from the external IMU, if it is mounted on the frame. But it's still possible to use correction in a simplified mode: compensate the linear accelerations and the heading drift. At least YAW encoder is strictly recommended.

Connecting to the SBGC32 controller

Currently, only serial TTL interface is supported. The external IMU sensor can be connected to any free serial port. The minimal number of wires are Rx, Tx, and GND. The +5V power supply may be taken from the gimbal controller. The signal lines should be crossed: Rx goes to Tx, Tx goes to Rx.

All available serial ports in the SBGC32 controllers are listed below:

- UART1* - a 4-pin header connector that is present in all SBGC32 controllers. It's paralleled to the onboard USB-UART converter (excepting the "Tiny Rev. A,B" boards that have a dedicated USB port). Simultaneous operation of the GUI and the external IMU is not possible for that reason.
- RC_SERIAL** - shares a functionality of the RC inputs: RC_ROLL (Rx) and RC_YAW (Tx).
- UART2** - in the "Regular" and "Tiny" controllers available as a shared function of the AUX3 pin (Rx only). In the "Extended" it is shared with the SPI interface for encoder connection (MISO=Rx, SCK=Tx). In the "PRO" board version it is a separate port. This port may be labeled as "UART_B" in some circuits.
- USB VCP / UART3 – For the "Tiny" version direct USB port emulating virtual COM port, for the "OEM" it's an additional serial port.
- CAN – for controllers equiped by CAN bus, it's possible to connect GPS_IMU using Serial-to-CAN functionality, and use it as an "External IMU" (see remarks below for versions compatibility).

Remarks:

*If UART1 port is used, special protocol starts with a delay of 5 seconds. It is done in order to be able to connect to the GUI on this port. In the case of connection problems, it is possible to reset all the serial ports to their default configuration by pressing the menu button 8 times in a series.

** It is possible to swap the UART2 and RC_SERIAL port pins by enabling the option "Hardware" - "Serial connection" -"Swap RC_Serial <-> UART2 ports". It may be required to simultaneously connect RC receiver by the serial protocol (Futaba s-bus, Spektrum) using only single Rx line, and the external IMU that need both Rx and Tx lines. After swapping, the RC_SERIAL moves to the AUX3 (Rx only), while the UART2 moves to the RC_ROLL (Rx), RC_YAW (Tx).

For the serial ports other than UART1, it is required to enable them and to configure speed:

- RC_SERIAL is activated by setting the "RC_ROLL pin mode" = "Serial port (Serial API, etc.)" on the RC tab;
- UART2 is activated by enabling the flag "Hardware" — "Serial connection" – "Enable UART2";
- Speed settings are located in the group "Hardware" — "Serial connection".

WARNING: It is necessary to disconnect an external IMU, if firmware update process is failed with the error "No answer from bootloader".

MavLink-enabled FC

Choose a free telemetry port and configure it for baud rate and protocol. See the "Connection" and "FC settings" in the next chapter. It is required to configure additional parameters of the connection and some parameters of the FC as described in the section "Support of the MavLink protocol for FC connection";

Basecam GPS_IMU

Prior to SBGC32 firmware 2.71b6: Connect by serial port only. **GPS_IMU v.1.2** has by default UART1 configured to accept Serial API connection (baud rate: 115200, parity: none), while UART2 is configured for an optional external GNSS receiver; Though, it can be configured to accept Serial API as well.

Starting from SBGC32 firmware 2.71b6, GPS_IMU firmware v2.04: CAN connection option is available for the controllers equiped by CAN bus. Select "CAN" in the "Connection" drop-down list and connect to the CAN port of GPS_IMU sensor.

Vectornav VN100/VN200

Use serial port UART #2 for a connection. Baud rate is 115200 (default value from factory). No special configuration of this sensor is required.

InertialSense µAHRS EVB

Use serial port ser0 ("port 4" of the EVB, shared with the USB connector). The ser1 ("port 6") is not recommended to use as it has known problems in the 1.1.5 release. The +5V supply can be taken from the main controller and connected to the "port 1". See http://docs.inertialsense.com/user-manual/hardware_integration/ for more information.

Connect sensor to PC by the USB cable and change the following parameters in the EvalTool – 'Data sets' – 'DID_FLASH_CONFIG':

- **startupNavDtMs=4** – specifies the rate of messages sent to the SBGC32; see comments below
- insDynModel=4 for automotive use; chose other options like pedestrian or flight that is appropriate for your applications;
- ser0BaudRate=115200 (or ser1BaudRate=115200, if ser1 is chosen for a connection). After the baud rate of ser0 is changed, you need to specify the same rate in the 'Settings' of the EvalTool and reconnect. In SBGC32 settings, set the same value for the UART port where module is connected.

All other parameters should be left with the factory-defaults values. For fine tuning and calibrations refer to the Inertialsense's User Manual.

Compatibility between versions:

SBGC32 firmware	*Inertialsense firmware*	*startupNavDtMs*
below 2.68b7	1.6.x	24
2.68b7 and above	1.7.x	4 (default)

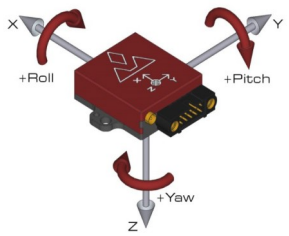
Configuring the main controller

- **Model** — select sensor's model from the drop-down list.
- **Connection** - all available serial ports are described above. Ensure that port is enabled and it's baud rate matches the speed of the external sensor.
- **Mounting position** - the location of the external sensor relative to the gimbal motors in their order starting from the camera. Possible options are:
  - On the camera platform, where the main IMU sensor is located;
  - Below the middle motor;
  - Below the outer motor;
  - On the frame, above the outer motor. Remarks:
    - *this position is not linked to the position of the YAW motor in space (actually, it can be located above or below the camera, or even assigned to the ROLL or PITCH axis in a different order of motors). "Above" or "below" term relates to the order of motor counted from the camera platform.*
    - *To make this mode working properly for a 3-axis gimbal loaded with the non-encoder firmware,*

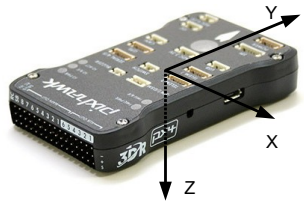
a single encoder must be installed on the YAW motor!

- **Orientation** — a direction of axes of the external IMU relative to a direction of the camera in a "normal" position. Axes configuration you can find in the datasheet for a given sensor.
  Note that for a MavLink-connected FC, we use the following coordinate system: X-right, Y-forward, Z-down (RFD). So, in normal position orientation should be configured such way: axis TOP="-Z", axis RIGHT="X".

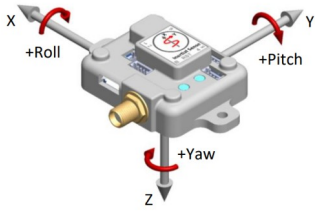
19 Using an external IMU sensor to improve the precision of stabilization.



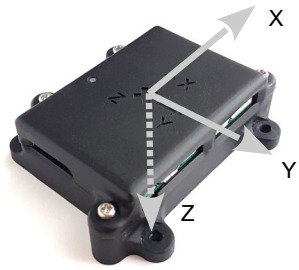
Vectornav VN200



Pixhawk PX4 (MavLink)



InertialSense µAHRS



Basecam GPS_IMU v.1.0

- **ROLL and PITCH angles** — select which mode of correction will be used for an attitude:
  - **Compensate for linear accelerations only** — in this mode gyroscope and accelerometer in the main IMU are used. Accelerometer receives the correction of the linear accelerations from the external IMU, which improves the overall precision of the attitude estimation algorithm in hardly dynamic conditions. This mode is not sensitive to minor errors in the coordinate system conversions. It works even in the non-encoder firmware.
  - **Use as an attitude reference (recommended)** — in this mode, the internal gyroscopic sensor is still used, but the internal accelerometer is completely disabled and replaced by the information coming from the external IMU sensor. This mode is sensitive to the errors in the coordinate system conversion. (this kind of errors appears when there are joints between the external and the internal IMUs).
    The strength of the correction is defined by the parameter "Hardware" – "IMU sensor settings" – "Gyro trust": the less is its value, the stronger is the correction (I,e, more trust is given to the provided reference attitude compared to the attitude estimated by the internal IMU).
- **YAW (heading) angle** — defines which mode of correction will be used for a heading angle:
  - **Don't use** — heading angle is not corrected and may drift. This mode is useful when the external IMU does not work reliably in a bad magnetic environment (inside buildings, for example).
  - **Use as a heading reference** — like in the case of an attitude correction, we still use a gyroscope of the internal IMU and correct it by the information from the external IMU. The strength of correction is defined by the parameter "Hardware" - "IMU sensor settings" – "Heading correction factor": the greater is the value, the faster is the correction.
- **Provided angles replace the estimated angles** — in this mode, the internal IMU sensor is not used anymore for an attitude/heading estimation. Both attitude and heading reference modes should be enabled. Remarks regarding this mode:
  - Its advised to use it only with the high-grade IMUs and only in the encoder firmware. The only exception is when the external IMU is mounted on the camera platform and coordinate conversion is not required.
  - It's NOT RECOMMENDED to use it with the MavLink-connected FC because the current implementation of the MavLink in SBGC32 and Ardupilot causes to receive angles with a big delay and low resolution.
- **Automatically update orientation on frame inversion** — it relates to a case when the frame is "inverted": middle motor rotates by 180 degrees. It works in conjunction with the parameter "Service" – "Frame inversion auto-detection", when the flag "is in inverted position" becomes set.
- **Use as a frame IMU** — if the 2nd IMU sensor is not connected (or not enabled in the GUI), and the external IMU sensor is mounted in one of the "Above the outer motor" or "Below the outer motor" positions, it is used as a replacement for 2nd IMU sensor. It may be useful for a non-encoder

firmware, where the presence of the 2nd IMU is mandatory for a normal operation. In the encoder firmware 2nd IMU generally is not required, but this option still may be useful for a debugging: in the "Monitoring" tab you can switch to the frame IMU sensor by pressing the [F] button, and check the validity of the angles received from the external IMU after all conversion.

- **Enable gyroscope online calibration by external reference*** – if enabled, the signal from gyroscope of the external IMU is used to calibrate biases of the internal gyroscope online. With this option, you can disable calibration of the gyroscope at startup. It helps to properly start system in complicated conditions when the regular calibration is not precise, and there is no option to calibrate gyroscope at startup (for example, a gimbal is mounted on a crane and always bounces). Also, it increases precision if temperature changes significantly during the work (as a rule, high-end IMUs are better calibrated and more immune to the temperature variations).
- **Alignment correction*** – small rotation over X, Y, Z axes, in degrees. It corrects misalignment in position between the external IMU sensor and the internal IMU sensor. This correction is applied after the conversion to the local coordinate system (linked to the camera platform) is made, so X,Y,Z are local axes, not the ext. IMU axes! There is an option to calibrate the alignment correction automatically. It should be done when the internal gyroscope is precisely calibrated and its alignment is configured. To do it, press the **"AUTO"** button next to the fields and make a series of smooth rotations of the whole gimbal along all three axes by ±40..60°. If the ext. IMU sensor is not located directly on the camera platform, try to keep relative position between the ext. IMU and the main IMU unchanged (i,e, motors should not rotate).

** These options are supported in the firmware ver. 2.68b7+ for the "Extended" and "Pro" boards. Work with the Vectornav VN100/200, Basecam GPS_IMU and Inertialsense µAHRS models.*

Checking that everything works properly

Controller sends to the GUI some status information that may be useful for a debugging. It consists of the communication status (counters of the received and lost packets), AHRS health state, the rotation matrix (DCM) converted to the (END) (East-North-Down) system, and the linear acceleration (with the gravity vector subtracted), expressed in a sensor's body coordinates.

Checking that the orientation and position is configured correctly

It's advised to do this checks after mounting the external sensor on the frame, setting up a connection and making initial configuration.

Method1:

1. Disable the 2nd IMU, if it is enabled ("Hardware" – "Frame IMU sensor" – "Mounting position");
2. Set the position for an external IMU "On the frame, above the outer motor";
3. Enable the flag "Use as a frame IMU";
4. In the "Monitoring" tab, switch to the frame IMU by pressing the [F] button. Tilt the frame where the sensor is installed, and observe angles displayed in the dashboard. In a leveled position, it should show zero angles and YAW heading panel should show the actual direction to the North.

Method2:

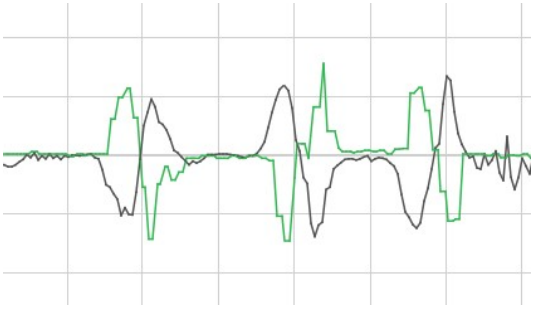
1. Enable both modes "Use as an attitude reference", "Use as a heading reference";
2. Enable the flag "Provided angles replace the estimated angles";
3. The dashboard in the GUI will show the angles, provided by the external IMU and translated to the camera platform. Those angles should match the angles that the main IMU normally calculates. When you fix the camera platform and rotate the frame, angles should not change. The amount of

change will show how big is the "translation error", caused by the imperfectness of the frame and non-linearity of the encoders.

Checking that the linear accelerations correction is applied properly

Do this check only if correction mode is set to the "compensate for linear accelerations only".

1. In the "Monitoring" tab, enable requesting additional information: "Request data – extended".
2. Turn on plots for the variables ACC_X and ACC_CORR_X in the variables panel at the left side.
3. Level camera and move gimbal from side-to-side along the X-axis of the camera platform (X-axis points right relative to a camera). The shape of both plots should match, but be in the opposite



direction:

4. Repeat the same for the Y and Z axes (tilt camera to make Z-axis values close to zero before starting the test). The same result is expected as for the X-axis.
5. Correction should not depend on a position of the camera relative to the frame, and the frame relative to the ground.

The recommended configurations

MavLink FC mounted on a frame, its forward direction matches with the gimbal's "forward"

- Position: "Above the outer motor"
- Orientation: Axis TOP="-Z", axis RIGHT="X"
- "Compensate for linear accelerations only"
- "Use as a heading reference" (in case of the good magnetic environment, otherwise "Don't use")
- For a non-encoder firmware, if 2nd IMU is not present, enable the "Use as a frame IMU" option.
- Gyro trust = 50..100
- Heading correction factor = 50..100
- For the recommended autopilot settings, see the section "Using as an external GPS-aided IMU for a hand-held gimbal"

External IMU is mounted on a frame; its forward direction (X axis) matches with the gimbal's "forward"; there is a vibration dampers between the frame and the gimbal

- Port speed: 115200
- Position: "Above the outer motor"
- Orientation: Axis TOP="-Z", axis RIGHT="X"
- "Compensate for linear accelerations only"

- "Use as a heading reference" (in case of the good magnetic environment, otherwise "Don't use")
- Gyro trust = 40..100
- Heading correction factor = 100..150

External IMU is mounted on a camera platform and aligned to its direction, or is mounted on a frame having precise mechanics and encoders

- Position: "On the camera platform"
- Orientation: Axis TOP="-Z", axis RIGHT="X"
- "Use as an attitude reference"
- "Use as a heading reference" (in case of the good magnetic environment, otherwise "Don't use")
- "Provided angles replace the estimated angles"
- "Enable gyroscope online calibration by external reference"
- Gyro trust = 30..50
- Heading correction factor = 150..200

20. Support of the MavLink protocol for FC connection

Starting from the firmware version 2.60, SimpleBGC32 controller supports a connection to an external flight controller (FC) using MavLink v1.0 protocol. This protocol is used for a communication between autopilots and ground stations (GCS) or OSD telemetry.
MavLink support was tested with the popular open-source flight controller ArduPilot (www.ardupilot.org) running on the Pixhawk PX4 hardware (http://ardupilot.org/copter/docs/common-pixhawk-overview.html).

Benefits of connecting gimbal to a flight controller

Correcting of the gimbal's IMU sensor
The availability of the quality information from the INS (Inertial Navigation System) allows compensating horizon drift during highly-dynamic flight. In this video, you can see the comparison tests of the behaviour of the gimbal with or without correction: https://youtu.be/yqsWTf2uV1g

Controling gimbal automatically
The autopilot can specify the angle at which the camera should point to. It can be used to control the gimbal within the flight program or in the object tracking mode.

Remote editing of gimbal's parameters
GCS (ground control station) can change the values of some parameters of the gimbal. The list of such variables is the same as for the Adjustable Variables.

Obtaining data from the RC receiver
Gimbal controller receives data from the RC receiver connected to the autopilot. You can omit the connection of the receiver to the gimbal controller directly.

Protocol compatibility

SimpleBGC32 provides a limited support of MavLink specification. The limitations are:

- Only protocol version 1 is supported
- Message routing between channels is not supported
- SBGC32 handles only messages where `systemId` and `componentId` matches the specified in its settings
- Broadcast messages are not handled
- Only a limited set of messages is supported (see below)

Connection

As a rule, flight controllers have several telemetry ports operating at MavLink protocol. To connect the gimbal, you have to choose a free port and set baud rate not less than 115200 in the autopilot settings, since the gimbal controller requests a large amount of data at a high frequency. In ArduPilot responsible for this parameters:

- SERIALx_BAUD = 115
- SERIALx_PROTOCOL = 1,

where 'x' is a port number. **Do not choose value 7 (alexmos), when working over MavLink protocol! Correct value is 1 (mavlink).**

Additionaly, set the following message rates for this port:

- SRx_EXTRA1 = 20
- SRx_POSITION = 5
- SRx_RC_CHAN = 20

IMPORTANT! All other SRx_xx parameters should be set to 0 to not overload channel by the unwanted data.

The SimpleBGC32 controller supports up to two MavLink channels. In the GUI tab "External IMU" - "MavLink" you must choose which serial port to use for this interface. All available options are similar to options listed in the section "External IMU sensor/connecting". Next, you need to select the parameters of the serial port, which shall comply with the autopilot settings: "115200, none parity ".

SBGC32 settings

- **Serial port, port settings** – select a port where to wait MavLink commands, from drop-down list. If port differs from UART1, do not forget to enable it like described in the section "External IMU sensor/connecting".
- **System Id** - ID of the entire system, which includes autopilot and gimbal. The default is 1.
- **Component Id** - ID of the subsystem. Set value 154 for a gimbal.
- **Send heartbeat** – each second the 'heartbeat' message is sent, signalling to the system that gimbal is connected and operating normally. Default value is "enabled".
- **Debug** – a random part of the incoming and outgoing messages is forwarded to the GUI for debugging. It is recommended not to enable this option unless necessary.
- **Query RC data** - the RC-receiver data that is connected to the autopilot, is requested and stored into variables API_VIRT_CH_1-16. They can be used to control the gimbal by assigning them to any function on the "RC" tab and others.
- **Mavlink control mode** *(frw.ver. 2.66+)* - how the command DO_MOUNT_CONTROL is handled. In the "disabled" mode, control returns to regular RC mode and MavLink control commands are ignored. Second option is to enable control for ROLL and PITCH axes only. Last option is to enable control for all axes (default).
  Unlike other MavLink settings, this parameter is split per profiles, so you can have different control modes in different profiles. Also, its value can be changed remotely by the adjustable variable "MAV_CTRL_MODE".

Firmware version 2.66x and above: there are additional settings that define how IMU correction is applied: "Using an external IMU sensor to improve the precision of stabilization".

Firmware version before 2.66: system use the provided information only for a heading correction and for a linear accelerations compensation. Only one mounting position is supported: "On the frame, above the outer motor". If the second IMU sensor is connected, it must be mounted in the same position. The orientation should be aligned with the camera's direction in a "normal" position.

> **NOTE:** *To enable the use of the IMU correction and automatic control in 3-axis gimbals with the non-encoder firmware, one encoder still must be installed on the YAW axis.*

FC settings

Configure the selected telemetry port, as described above. Set the MavLink-compatible type of a gimbal, if you want to control it. For the ArduPliot FC, set the parameter MNT_TYPE = 4. Other gimbal-related parameters described in the documentation http://ardupilot.org/copter/docs/common-cameras-and-gimbals.html

Tips:

- In the automated missions, gimbal can be controlled by the MAV_CMD_DO_MOUNT_CONTROL command.
- If the "Follow" mode is enabled in the GUI, it will be disabled when gimbal receive first DO_MOUNT_CONTROL command.

Using as an external GPS-aided IMU for a hand-held gimbal

Autopilot can be used in this role because it has a full set of sensors required for a reliable attitude and heading estimation. The Ardupilot software is based on the Extended Kalman Filter (EKF2) and shows good results working as an external AHRS/IMU reference for the internal IMU sensor in a gimbal. It's recommended to make some tweaks intended to give the more trust to the GPS sensor in calculations:

- INS_GYR_CAL = 0 (do not calibrate gyroscope at the startup)
- INS_TRIM_OPTION = 0 (do not adjust the level of the frame)
- GPS_NAVFILTER = 4 (use automotive filter for GPS)
- EK2_VEL_I_GATE = 1000 (increase the GPS velocity innovation gate size to accept more measurements)
- EK2_POS_I_GATE = 1000 (increase the GPS position innovation gate size to accept more measurements)
- EK2_MAG_CAL = 2 (do not calibrate compass online because of the probability of work in a hard magnetic environment)
- EK2_GPS_CHECK = 3 (check only the number of satellites and HDoP)
- EK2_CHECK_SCALE = 200 (allow big GPS errors to keep using GPS in calculations)
- EK2_GLITCH_RAD = 10 (filter resets to GPS values if the predicted values differs more then this parameter (in meters))

Checking if all is working correctly

If everything is connected properly, a "MavLink" tab will display status information:

AHRS - OK, GPS - OK, RC - OK, Control – OK

- AHRS – we got the frame attitude
- GPS – we received the position and velocity information
- RC – we received RC-data
- Control - Autopilot is controlling gimbal at this moment

If the "Low rate" message is displayed, it means that message is received but its rate is below required. You may need to check the configuration of flight controller to increase the rate of data stream, as described above.

On information how to check that the IMU correction is working, refer to the section "Using an external IMU sensor to improve the precision of stabilization".

Communicating with gimbal using native Serial API protocol

(supported in fw. Ver. 2.70b2 and "extended" family of controllers only)
There is an option to use SBGC32 Serial API via MavLink communication channel, wrapping serial API commands by the `TUNNEL` message (starting from 2.70b5 firmware, also `V2_EXTENSION` message).

It allows to reveal the full potential of SBGC32-powered gimbal in the on-board computer or ground station, overcoming the lack of support of gimbals in the standard MavLink protocol.

Our GUI can communicate with the gimbal via MavLink channel as well. For example, if FC has a serial connection (either by cable or by radio) to the computer running GUI, and gimbal is connected to the FC's local port via MavLink 2.0 protocol, GUI can connect to the gimbal via MavLink tunnel, providing all functionality excepting the firmware update.

How to configure:

- Configure gimbal and FC to communicate using MavLink 2.0 channel following the above-mentioned instructions. It's crucial to enable the support of MavLink V2.0 in both FC and gimbal ("External IMU" – "MavLink" - "MavLink V2.0" checkbox).
- GUI Menu "File" – "Settings" – "MavLink tunnel" – select tunnel type (or check "Enable" in old versions) and set gimbal's componentID as configured in the "MavLink" section.
- Select COM port where FC is connected and press "CONNECT"; ensure it's not occupied by GCS.
- For developers: in case of troubles establishing the connection, checking the runtime logging may be helpful; for that, run GUI in console mode by 'run_console.bat'

NOTE: Currently, the stock ArduPilot software works only with V2_EXTENSION message, TUNNEL is not routed.

For developers: a list of supported messages

Specification of the "common profile" messages: https://mavlink.io/en/messages/common.html
The list of supported messages provides minimal compatibility with the most popular ArduPilot FC. New messages will be add later.

Requesting telemetry data from gimbal

Autopilot / GCS can request telemetry using one of the following methods:

Requesting method	Data stream / Remarks	Telemetry message to be sent
REQUEST_DATA_STREAM	MAV_DATA_STREAM_RAW_SENSORS	RAW_IMU
	MAV_DATA_STREAM_RC_CHANNELS	RC_CHANNELS_SCALED
	MAV_DATA_STREAM_RAW_CONTROLLER MAV_DATA_STREAM_EXTRA1 MAV_DATA_STREAM_EXTRA2	ATTITUDE
MESSAGE_INTERVAL		RAW_IMU RC_CHANNELS_SCALED ATTITUDE*
TIMESYNC	Supported in V2 protocol only, frw.ver. 2.72b0+	TIMESYNC

* `ATTITUDE` remarks:

- the Euler order it is not the same as used commonly in autopilots (ROLL-PITCH-YAW). In our system it's configured in the "RC" tab, "Order of Euler angles". It should be taken into account when

converting Euler angles to quaternion or DCM. The current order can be obtained by `PARAM_VALUE("EULER_ORDER")`, 0 – PITCH-ROLL-YAW, 1 – ROLL-PITCH-YAW.
- "euler_speed[3]" field is a rotation vector X,Y,Z rather then Euler rotations.
- "timestamp" field is time when the angles were actually updated.

Telemetry data provided by autopilot

Gimbal requests the following telemetry data:
- `ATTITUDE` – use it as a reference for the frame heading angle
- `GLOBAL_POSITION_INT` – velocity information to compensate lateral accelerations
- `RC_CHANNELS_RAW or RC_CHANNELS` – assigned to API_VIRT_CH1..16

It requests this data using message `REQUEST_DATA_STREAM`, specifying streams
`MAV_DATA_STREAM_EXTRA1, MAV_DATA_STREAM_POSITION, MAV_DATA_STREAM_RC_CHANNELS`

Updating runtime parameters

Autopilot / GCS can get or set values of adjustable variables. New value take effect in runtime only, it's not saved to EEPROM.

- `PARAM_REQUEST_LIST` – request a list of available variables
- `PARAM_VALUE` – get current value*.
- `PARAM_SET` – set new value*.

** Pay attention to param_type – it's int32 or uint32, not float! Use the same param_type that you get in the PARAM_VALUE, to set a new value in the PARAM_SET.*

Starting from firmware 2.70b3, it's possible to save new value to EEPROM by replacing the leading "G" to the "#" in its name. For example, sending PARAM_SET("G_PID_P_R", 100) will change the runtime value to 100, while sending PARAM_SET("#_PID_P_R", 100) changes value and saves it to EEPROM. Note that the statbilization is interrupted for a short time (10-20ms) during this process.

Controlling gimbal

There are two ways to control gimbal:

1) Sending an angle setpoint in `COMMAND_LONG:`

- `DO_MOUNT_CONTROL` – in angle mode, sets the provided angles as a target. In angular rate mode, sets the angular rate as a setpoint

- `DO_MOUNT_CONFIGURE` – set mode of operation
  - `MODE`
    - `MAV_MOUNT_MODE_NEUTRAL` – go to home position: ROLL, PITCH are leveled, YAW in a neutral position relative to frame.
    - `MAV_MOUNT_MODE_MAVLINK_TARGETING` – get target angles from the DO_MOUNT_CONTROL command
    - `MAV_MOUNT_MODE_RC_TARGETING` – return control to the RC or "Follow" mode (MavLink control is turned OFF)
  - `INPUT MODE[3]`
    `bits 0..7: mode`
    - `MAV_INPUT_MODE_ANGLE_BODY = 0` - (for YAW only) angle is in body coordinates (relative to frame). If frame rotates, gimbal strictly follows
    - `MAV_INPUT_MODE_SPEED = 1` - angular rate (in global frame)

- `MAV_INPUT_MODE_ANGLE = 2` - angle (in global frame)
- `MAV_INPUT_MODE_SPEED_FOLLOW = 3` - angular rate (in global frame) + mix follow*
- `MAV_INPUT_MODE_ANGLE_FOLLOW = 4` – angle (in body frame) + mix follow*

- `bits 8..23: flags, may be combined with the mode` *(frw.ver.2.72b0)*
  - `INPUT_MODE_FLAG_SHORTEST_DISTANCE = 1` – move to the target angle by the shortest distance
  - `INPUT_MODE_FLAG_TARGET_PRECISION = 2` – in ANGLE mode, tracks the target angle aggressively (reduces smoothing and speed attenuation near the target). This mode assumes that the command is sent at a constant high rate.

** External control is mixed with the follow mode, if it's enabled in the gimbal settings. Angle is in a body frame (relative to the home position). If follow mode is disabled, angle is in a global frame.*

NOTE: By default, YAW angle is considered relative to frame (legacy behavior in ArduPilot FC), so we add YAW angle of the frame as received in the message `ATTITUDE`. But it can be changed by setting the flag "YAW angle is absolute" in the GUI.

2) Passing RC signal using message `RC_CHANNELS_RAW or RC_CHANNELS` and assigning RC channels `API_VIRT_CH_1-16` to the functions in GUI.

Accessing Serial API protocol

Messages related to the section [Communicating with gimbal using native Serial API protocol]

`TUNNEL (#385)`

```
payload_type (2u)
target_system (1u)
target_component (1u)
payload_length (1u)
payload (variable length)
```

`V2_EXTENSION (#248)`

```
payload_type (2u)
target_network (1u) = 0
target_system (1u)
target_component (1u)
payload_length (1u)
payload (variable length)
```

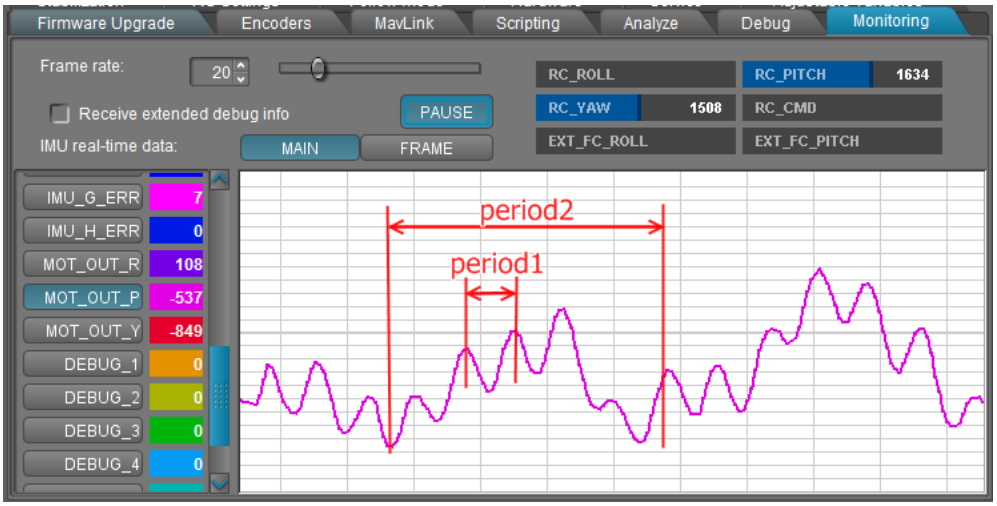
Both commands have similar parameters. The '`payload_type`' parameter encodes the Serial API's command Id: `command_id=payload_type-32767`. The '`payload`' bytes corresponds to the *payload* of the command (without header and checksum!). Gimbal responds to the incoming messages by the same command, using originator's *system_id* and *component_id* as '`target_system`' and '`target_component`'. Flight Controller should route this message from/to target node transparently.

21. Correction of the motor's "cogging" effect

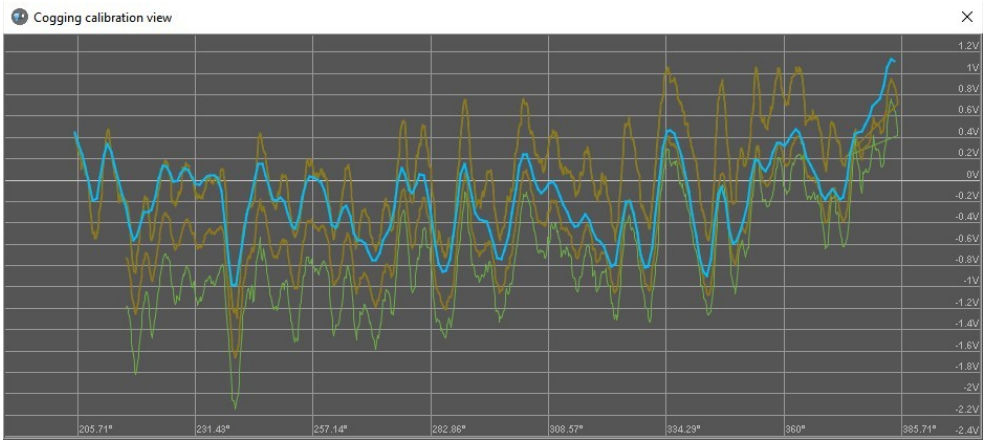
NOTE: This function is available only in the controllers “Extended” series and “Pro” loaded with the encoder firmware version 2.62b6 and higher.

When selecting the most appropriate motors for a gimbal, the choice is often made of motors with discrete rare earth magnets in the rotor. As compared to ring-shaped ferrite magnets, they provide greater effectiveness, owing to the creation of a stronger magnetic field. However, there are drawbacks, as well – the field is not spread as uniformly as in the case of a ring magnet. Because of this, there always exists cogging in the motor appearing as “sticking”. In an ideal motor, the rotor must move freely in relation to the stator in the absence of a current, practically with zero effort. In fact, the rotor occupies several fixed intermediary positions and holds itself in them. The force required for movement may vary for different motors, and determines its quality. Besides the “sticking” effect, other types of nonlinearity are possible, for example, nonlinear distribution of the magnetic field in the stator, depending on the sinusoidal voltage applied to the three phases, since motors may be designed for trapezoidal and not sinusoidal control (see a difference between BLDC and PMSM types).

It is possible to visually evaluate the nature of distribution and the value of nonlinearity on the graph in the tab “Monitoring”, by enabling the variables MOT_OUT_X and slowly turning the motor with the help of a joystick or a control panel:



Firmware and GUI 2.69b5+ has a dedicated tool to visualize the calibration process and the resulting LUT:

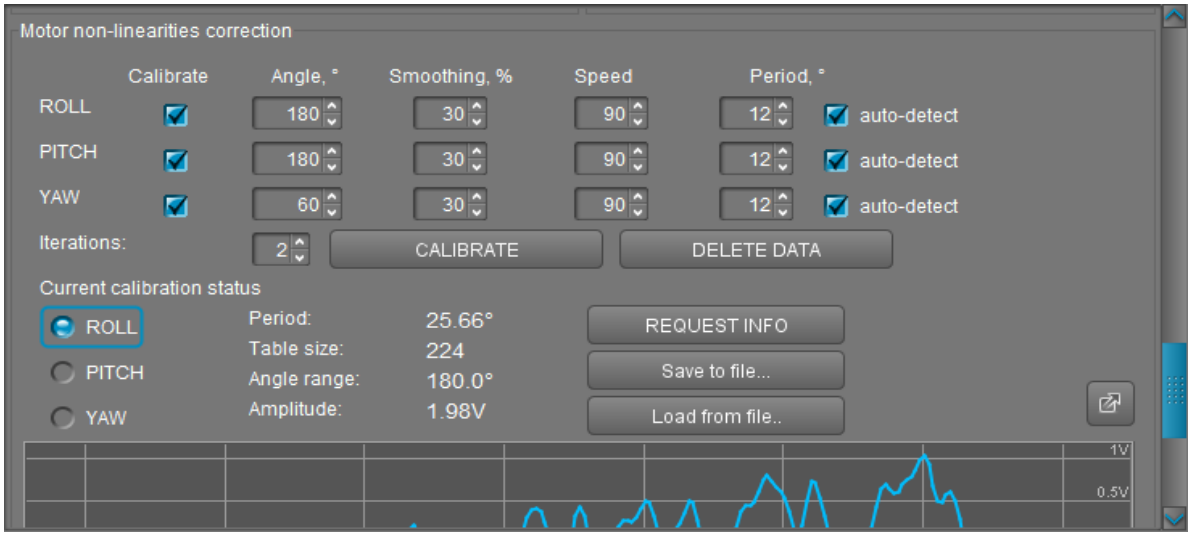


When the stabilization is working, cogging appears as a vibration of the camera when tilting the frame or turning the camera. It is not possible to avoid these artifacts completely with the help of system settings,

they can only be reduced by setting a higher PID gain. This problem is bigger when the number of poles in the motor is larger, causing as a rule, the shorter period of nonlinearity. At a given speed they might become even stronger, because in the feedback system, starting with some frequency (some tens of Hertz), there is a characteristic rise in the gain coupled with the inversion of the phase, when it actually amplifies an error instead of correcting it. Nonlinearities of a multi-pole motor might enter this frequency range at some speed of rotation, and become amplified causing unwanted vibrations.

For correcting cogging of any kind, a special calibration can be used. It builds a table (Look-up table, LUT), which encodes a dependence of offset voltage, which is required to be fed to the motor, from the angle of the motor measured by an absolute encoder.

Calibration starts in the section GUI "Stabilization":



Parameters of calibration:

- **Calibrate** – Choice, which axis to calibrate or which calibrations to eliminate.
  IMPORTANT NOTE: before a new calibration, it is necessary to manually delete the old calibration, if there is one (system will be restarted).
- **Angle, °** – gives the range of calibration, starting with the current angle of the motor, in the positive direction. Since the size of the table is limited (now it is 4096 count max), the greater the angle, the lower the resolution of the correction table, so it makes sense to calibrate only the working range of the gimbal, and not all 360 degrees of the motor.
- **Smoothing, %** - Smoothing. The greater it is, the more high-frequency nonlinearities and calibration errors smoothen out, and as a result, less appear during the working of the motor in form of noise. Default smoothing 30.
- **Speed** – speed of movement of the motor to construct the curve. More the poles in the motor, smaller will be the period (step) of nonlinearity, and that much slower the speed should be. Indicative figures for choice of speed: 14 poles - speed 50-100, 42 poles – speed 30-40. For a geared motor, it should be set proportinally slower speed.
- **Period, °** - Period (or step) of nonlinearity. By default it is determined automatically by the best matching sinusoidal signal. This value defines the choice of table size, the cut-off frequency for smoothing and high-pass filtration. As a rule, the period depends on the number of poles of the motor – the more they are, the smaller will be the period. But more than one period might present (for example, the graph above shows a case with two periods). A period may be set manually for those nonlinearities which need to be compensated on priority, for example if automatically detected period is too big, it causes higher frequencies of non-linearities to be attenuated.

- **Iterations** – a number of passes. The resulting LUT is an average of all passes. Normally 2 passes is enough.

Indispensable conditions for carrying out calibration of nonlinearities

- The gimbal must be fully configured: encoders calibrated, PID parameters set to maximum performance, providing precise stabilization. If encoder linearization is planned, it must be done before the cogging calibration!
- The system must be well balanced. A small disbalance is permitted.
- The frame must be fixed firmly (gimbal must be located on the stand, and not in the hands)
- There must not be any obstacles to the free rotation of the motors through the specified angles.
- It is desirable to carry out calibration with the least possible weight of the camera, so as to reduce the inertia of the system. Choose the lower speed if inertia is high.

Switch on the gimbal, set out the initial angle of the motor with help of a joystick or remote control (RC), and start calibration. At the end, information about the results of calibration is displayed.

- **Period** – automatically determined period of nonlinearities
- **Table size** – table size, sufficient for encoding of this period and the given range of angles.
- **Amplitude** – Maximum amplitude of nonlinearities. Expressed as voltage, required for overcoming the force of nonlinearity for a given motor.

It is possible to save the calibration to a file and restore it from a file. For switching off the correction, it is necessary to delete the calibration. For starting a new calibration, it is necessary to delete the previous one and wait for the system to restart.

NOTE: this kind of calibration is very demanding to RAM space. If calibration failed (controller hangs or emergency stop error arise), you need to free up some space in RAM by temporarily disabling extra functions like scripts, adjustable variables, disabling extra UART/MavLink ports. If there was made calibrations for other motors, you can temporary save them to files and delete from the controller, restoring later when all axes will be calibrated.

WARNING: calibrating the encoder's field offset or linearity LUT will make cogging correction table absolete, because it uses encoder signal as a reference for LUT index. The same is true for mechanical adjustments of encoder-to-motor position. You have to re-calibrate cogging correction after that.

22. Using 4th motor for a gimbal frame positioning

The "Extended" and "Pro" board versions support a control of the 4th motor, either direct drive or geared, used not for a stabilization, but for a frame positioning, to provide optimal conditions for the operation of the main three motors.

Hardware and connection options

The following connection options are supported:

- An external controller that receives information from the gimbal controller via the Serial API and independently controls the 4th motor. The CMD_REALTIME_DATA_CUSTOM command provides the MOTOR4_CONTROL structure, which has all the necessary information for controlling the 4th axis
- A servo driver receiving the control task from the gimbal controller via the PWM signal. The period of the PWM signal is constant, and the duty-cycle determines the direction of rotation and varies within the specified limits from the neutral point (configured by the parameters, see below).
- CAN_Driver, controlled over the CAN bus and driving a 3-phase BLDC motor with an encoder. More information about connecting and configuring the CAN_Driver can be found on our website https://www.basecamelectronics.com/can_driver/

Since the 4th motor does not participate in stabilization, it does not require high speed and low backlash. It can be a small motor with a reduction gear, providing enough torque and speed for a positioning of the gimbal's frame. It can be either a DC brushed or brushless motor, depending on the selected driver. However the optimal result is achieved when the driver controls the speed of the motor, i.e., the rotational speed is proportional to the magnitude of the input signal observing polarity. But it is possible to use a simpler driver that controls the current (momentum) or just the output voltage level.

For normal operation of the servo drive control algorithm, it is necessary to ensure a correct balancing so that the required holding torque is minimal and does not depend on the orientation of the camera relative to the frame. The best option would be if the 4th axis passes through the center of masses of the gimbal.

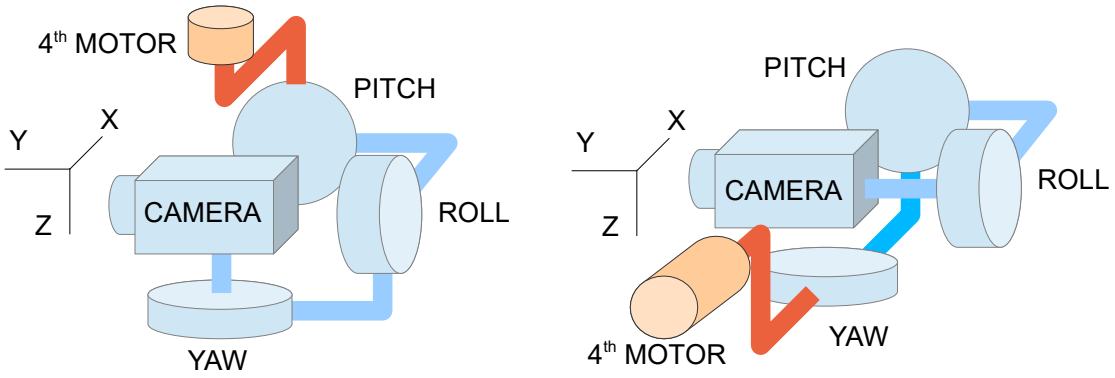
Mounting options

Automatic control of the 4th axis is performed to solve the following tasks:

1. Maintaining the orthogonality of 3 main stabilizer axes
2. Maintain one of the main stabilizing motors inside its working range of angles. Automatically will be selected the motor that is closest aligned to the 4th axis (i.e., the correction of which will be most effective).

The 4th axis is set based on the priority of the task, that need to be solved. A mandatory condition is that it must coincide with one of the three axes of the coordinate system tied to the gimbal frame.

22 Using 4th motor for a gimbal frame positioning



Example 1: The order of the main stabilization axes is "CAMERA-YAW-ROLL-PITCH". Obviously, in such a configuration, the working angles of the YAW motor are limited, as a result the pan motion of this gimbal is limited when the frame is stationary. To overcome these limitations the suspension frame should rotate repeating the rotation of the YAW motor. To do so the 4th axis is installed vertically, parallel to the YAW axis in the normal position, and the Z axis is selected in the settings for the 4th motor.

Example 2: The order of the main stabilization axes is "CAMERA-ROLL-PITCH-YAW". The frame is fixed in the normal position (the YAW axis is always vertical). When the camera is tilted down by 90 °, the axis of the ROLL motor becomes parallel to the axis of the YAW motor and a gimbal lock occurs (one of the stabilization axes degenerates). It is necessary to place the 4th axis horizontally (for example, parallel to the PITCH axis) and tilt the YAW motor axis after the camera to prevent blockage. However if the PITCH axis is not parallel to the 4th axis, the prevention of blocking is still possible, but the scenario is more complex.

Hardware settings

- **Axis of rotation.** Specify along which axis of the frame coordinates the axis of the 4th motor is oriented. In the normal position, the frame coordinate system coincides with the camera coordinate system. The END system is used: looking in the direction of the camera, X points right, Y points forward and Z points down.
- **Inverted** – check if you want to invert the direction of rotation.
- **Output** – command signal output. Options:
  - ***SBGC32 Serial API*** – the hardware output is not enabled. The rotation task for the 4th motor can be requested via the command SMD_REALTIME_DATA_CUSTOM; In addition, it is possible to configure its sending with a specified frequency by using the command CMD_DATA_STREAM_INTERVAL. These commands are also available when assigning a hardware output, and can be used for a debugging. For detailed information on these commands, see the *Serial API* manual.
  - ***PWM Servo #1..4*** - PWM output on one of the controller ports. You can customize the neutral point and the duty-cycle range (see below). The frequency (PWM period) is set by the parameter "RC Settings" - "PWM Bypass" - "PWM rate, Hz".
    - For the controller version "Extended" it is an alternative function on the ports Servo1 = FC_ROLL, Servo2 = FC_PITCH, Servo3 = RC_PITCH, Servo4 = AUX1;
    - For the controller version "Pro": Servo1 .. Servo3 have independent outputs on the logic

board, and Servo4 = FC_ROLL on the interface board.

- ***CAN_Drv #1..7*** – Basecam CAN_Driver module, connected via CAN bus and driving a 3-phase brushless motor. An absolute encoder is required - any model supported by the CAN_Driver. Note that Hall sensors or sensorless mode is not supported at this moment.This option is the most accurate for positioning the 4th axis. For details on the technical specifications, connection and configuration of the module, see its documentation.

- **Speed scale factor** . The preferred way to control the 4th motor is in the speed reference mode. For the proper functioning, it is necessary to synchronize the speed of rotation of the camera (for example, using the signal from the remote controller) with the speed of rotation of the 4th motor. If CAN_Driver is used, this factor is equal to the geared reduction factor, or 1.0 for the direct drive. Adjustment method: Set all parameters P, I, D for the 4th motor to 0 (this turns off the error correction) and rotate the camera by the RC command along the axis coinciding with the axis of the 4th motor. The 4th motor must rotate the frame at about the same speed, providing the desired correction even without a feedback from the angle.
- **PWM servo output configuration** – If PWM Servo # 1..4 output is selected, you can set the duty-cycle for a neutral point and for the range (values in microseconds).
  - **Feedback gain for speed**. Used only in PWM mode. It allows to improve a performance of the servo if it has no built-in speed feedback in a control algorithm. Before adjusting this parameter, set the parameters P, I, D to 0. Gradually increase the value by making the camera's test rotations and monitoring the reaction of the 4th motor, as described above for the speed scale factor adjustment. The greater the factor, the more accurate the 4th motor repeats the movements of the camera. But too high value can lead to instability and oscillations.

PID controller settings

In this block, the proportional coefficients of the targets for correction are adjusted, as well as the speed and the required correction accuracy:

- **Motor limits avoidance area, %** - Specifies the width of the avoidance area as a percentage of half of the full range of rotation for each motor. For example, if you put 100%, the system will tend to maintain its middle position. If you put 10%, the correction will not be applied until either of the motors approaches the 10% width area near one of the borders. If a particular motor has 360º freedom, this correction is not applied to it.

- **Motor limits avoidance weight, %** . Specifies how important the correction of avoidance of borders is compared to the correction of the "gimbal lock" condition (case when one of the stabilization axes degenerates due to the coincidence of two motors).

- **P, I, D** - the gains of the PID controller, which eliminates the error – the angle between the current position and the most optimal position that 4th motor could gain. P - proportional, I - integral and D - differential coefficients. Tips for setting up:

  - Set all the coefficients equal to 0 and gradually increase the proportional coefficient P, performing the test rotations of the camera, so the algorithm begins to perform correction, then watch how accurately the 4th motor fulfills it.

  - The coefficient I is integral. It is needed to improve the accuracy of correction near the neutral

position, and only if the 4th motor driver does not have a built-in integral coefficient! For example, for the CAN_Driver it is not needed, since it has a full-fledged PID controller.

- The coefficient D is damping. It must be used if the *quality factor* of the mechanical system is high (i,e, system tends to oscillations if only the P-gain is set). For drivers with an integrated speed controller, it's also not needed.

- **D-term LPF cut-off frequency, Hz** – is a low-pass filter for the D-component, which is the most noisy, as a result can lead to jerks if D-gain is set high.

23. Tips on designing mechanics of a gimbal

If talking about the performance of a gimbal, we have to separate the instant stabilization quality and the precision of tracking of the commanded angles.

Instant stabilization quality depends on how effectively external disturbances are rejected. If there are visible shakes or jitter in a picture from the camera, it means poor instant stabilization quality.

The precision of tracking of commanded angles depends on how precisely the IMU sensor measures angles. If gimbal slowly drifts (always or after some specific movements like fast panning) – it is a subject to improve IMU quality, as described in corresponding sections of this manual.

Also, a significant part is how **responsible** is gimbal in action, i,e, how the remote and "Follow mode" controls are handled. This behavior is defined by the parameters described in the "RC settings" and "Follow mode settings" sections of this manual.

Below we give several advises what to keep in mind when designing mechanics of a gimbal, to improve the instant stabilization quality.

Stiffness is very important!

In conventional gimbal structure when all motors are linked in sequence by arms, the possible play in bearings or flexibility or arms may cause significant declinations of the camera. In a static, it is not a problem: tension caused by the gravity force is compensated. But once a vibration comes from the frame, it causes oscillations and motors can't compensate it (because of the order of motors or because of the limited speed of response of PID controller). Even perfectly tuned PID controller can't reject disturbances in a full range of frequencies; normally, it is limited by 10-20Hz. The sources of vibration having a wide spectrum, as an example:

- motors on UAV frame for the UAV-mounted gimbals
- steps when operator walking, for the hand-held gimbals
- a rotation of motors that have a high cogging effect (see below)

Making mechanics more rigid gives the following advantages:

4) ability to set stronger PID gains with the higher effective frequency, as a feedback path has less phase delay and resonances (if present) having frequencies shifted higher
5) less amplitude of declinations at the end of the arms under vibrations

Tips:

1. Reinforce rotating joints and arms to keep a possible play and declination under full load at an acceptable low level.
2. If the installed camera has big lens and only one fixing point on a body, it forms a kind of oscillator. Add an extra point of support for the lens, to increase the overall stiffness.

Isolate vibrations as much as possible!

Using dampers is a good idea because they effectively isolate all external disturbances that come from gimbal's frame at high frequencies. Not a problem if dampers are too soft and low frequencies still passes through them or even amplified – PID controller is able to compensate them because it is very effective at low frequencies. Other obvious points why it is crucial to avoid vibration passing to the camera:

- IMU sensor is sensitive to vibrations – i,e, sensors may get confused by vibrations that come to it

© Basecamelectronics® 2013-2024

- An optical system of the camera may be sensitive to vibrations, especially if the camera is equipped by the optical stabilization system

To properly select characteristics of vibration dampers, you can use the "Analyze" tab to define the working range of the stabilizer from the sensitivity curve: the damper should isolate frequencies that out of reach of PID controller. Also, damper acts as a spring, that causes the system "mass + dumper" to swing at some frequency. The general rule is to keep this frequency low enough, where the PID controller can compensate swinging.

Adding dampers to a gimbal is a challenging task because once damper is installed, the outer motor (generally the YAW motor) loses reaction mass that it needs to operate properly. It will be hard to tune PID controller and set strong gains because motor's stator (part connected to a frame) has no mass to push off to provide a required torque, and complicated non-linear behavior of a damper interfere PID controller. There are several ways to solve this problem:

7. Designing non-uniform dumper, having degrees of freedom for sliding movements and rotations over the ROLL and PITCH axes, but being rigid for the rotation over the YAW axis; but it is a hard engineering task.
8. Connecting a reaction weight to the stator of YAW motor: for example, a massive ring, having maximum momentum of inertia along the motor's shaft. It may be either a four counterweight on an X-frame, and so on.

PID tuning is very important!

Some problems may be caused by badly tuned PID gains. Every PID controller amplifies disturbances in a small range of frequencies. If it is poorly tuned, it amplifies the unwanted range. The "Analyze" tool in the GUI can give detailed information about the performance of PID controller versus frequency and can help to tune a system to prevent excessive amplification and tendency to self-excitation.

Choose motors with a minimal cogging effect!

"Cogging effect" is a force that motor produces in different positions when not powered. The force depends on the angle and is structured. This negative effect significantly impacts the quality of stabilization, and there are no ways to fix it other than changing the motor.

Many gimbal motors on the market have a pretty low cogging effect, acceptable for the most use cases of a gimbal. However, if your target is to make a high-precision stabilizer that works with cameras having 10x zoom and more, it is extremally important to choose motors with near-zero cogging!

Another points to keep in mind:

- The less is number of poles, the better: frequency of disturbances caused by the cogging when the motor is rotated is shifted to the lower end and the PID controller can compensate them better
- Motors with the O-ring shaped magnets have much less cogging compared to spare rare-earth magnets (but as a drawback, such motors provides less torque)

Our system provides an option to compensate the cogging effect by calibration. But it does not eliminate it completely, only decreases several times.

Prevent static friction!

Friction in rotating joints should be minimized. Especially it's about static friction! Several things to keep in mind:

- Choose good bearings that do not lose performance under the highest projected load;
- Select proper bearing type, taking into account the direction of forces acting on them

© Basecamelectronics® 2013-2024

- Prevent wires that pass through joints, from scrabbling.
- Low-quality slipring is a possible source of excessive friction.

© Basecamelectronics® 2013-2024

24. Encoders non-linearity calibration

NOTE: This function is present only in the "Extended" family of controllers with encoder firmware version 2.69b5 and higher.

Most encoders used in gimbals have a common feature - the dependence of linearity on the accuracy of mechanical components installation. For example, magnetic on-axis encoders are very sensitive to the alignment of axes of their components, the stability of the distance to the sensor, the strength of the magnetic field and the magnetic environment (hard and soft iron presence nearby). But even under ideal conditions, the manufacturer admits an inaccuracy of about 1 degree, which is appeared in non-linearity of the readings depending on the angle.

In the design of high-precision stabilization systems, if high-end IMU sensors are used, the presence of such an error of 1-2 degrees in encoders causes errors in operation and does not allow to use the full potential of the IMU sensor. Also in middle-class systems with a conventional IMU sensor, errors in design of the encoder-motor assembly or poor-quality components can lead to a significant additional error in the accuracy of measuring angles.

The SBGC32 system provides a user-friendly calibration procedure that corrects the non-linearity of encoders, regardless of the factors that cause them, if they do not change over time. For this purpose a high-precision gyroscope of the main IMU sensor is used. An encoder error correction table (Lookup-table, hereinafter LUT) is built after comparing the reference rotation measured by the gyroscope with the encoder angles over the entire working range of the motor angles. If a high-precision gyroscope is not available, then even with a conventional IMU sensor it is possible to perform such calibration due to applied algorithms for mutual calibration and matching of the gyroscope and encoders. As a result, in addition to correcting encoder non-linearity, this tool provides another important function - it allows you to check the alignment of IMU sensor with the mechanics and accuracy of other calibrations required in the gimbal.

ATTENTION: Calibration of encoder non-linearity must be performed before calibration of motor non-linearity!

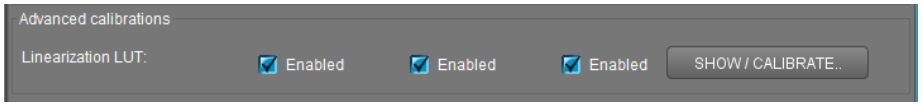
Calibration conditions

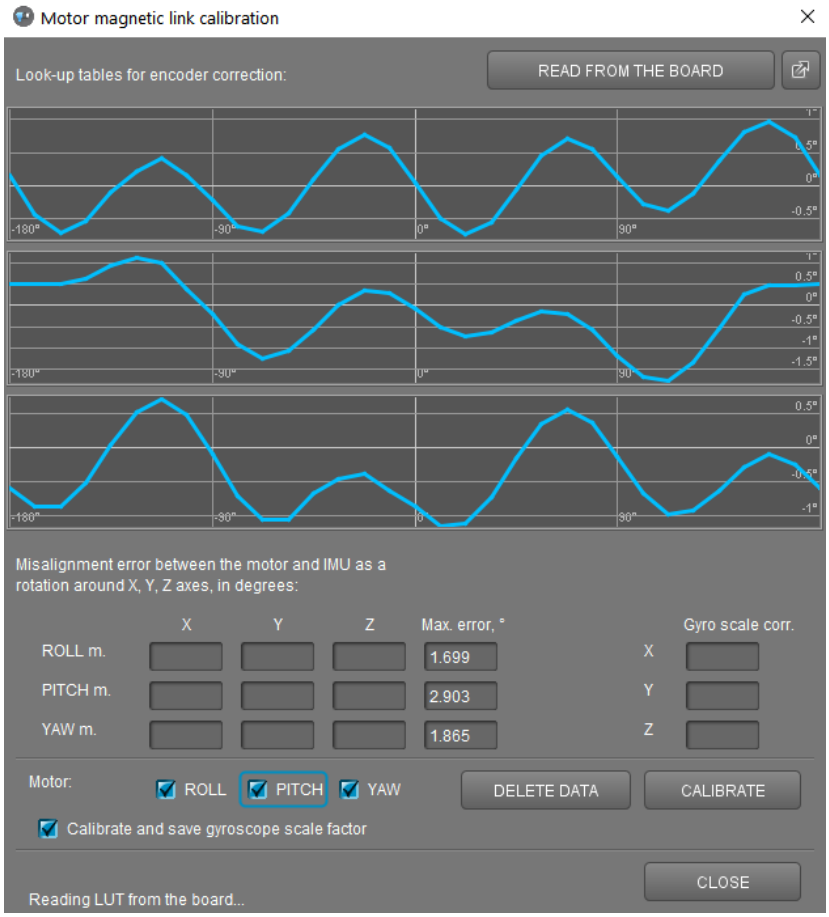
- It is required to ensure complete immobility of the gimbal frame (rigid stator fixation of the external motor)!
- Encoders to be set on all three axes with accurate offset calibration performed in advance.
- Fully tuned gimbal, well-performing the basic stabilization function.
- In case of absence of previously performed calibration of gyroscope scale factor: rotational freedom of the motor is 360 degrees or close to it.
- Only systems with 90° angles between the axes of motors are supported.

Calibration procedure

Calibration is performed in a fully automatic mode on an already assembled and tuned gimbal (it is not required to calibrate encoders separately from gimbal or install any additional sensors).

In GUI on tab “Encoders” - “Advanced calibrations” by the “**SHOW / CALIBRATE ..**” button a separate window opens where calibration can be launched, displayed the calibration process and the results are displayed in a numerical form with the LUT in a graphical form:





Use checkboxes to select the motors you want to calibrate and press the “CALIBRATE” button.

Calibration of each motor consists of three stages, at each of them, several series of rotations are performed. The range of angles is selected based on limits specified for each motor on tab ”Encoders” - “Settings” - “Limits, (min, max), deg.” If the motor rotation is not limited, a range of ±180° from the normal position is selected. For a motor with a reduction, it will be less.

Calibration stages:

1. Alignment of IMU axes and the motor

For LUT calibration a very precise alignment of motor rotation axis and the corresponding axis of the gyroscope is required. Discrepancies found are transmitted to GUI and displayed in “Misalignment error between the motor and IMU ...” section, for each calibrated motor separately.

They can be used to check the quality of other calibrations which affect the alignment of the axes: zero offsets in other encoders, the accuracy of IMU sensor installation and the accuracy of mechanics. For example, for the first motor axis in the order of reference from the camera platform, mechanically interconnected with IMU, the discrepancy can be caused only by imperfect orientation of IMU sensor. It can be calibrated in “Hardware” - “IMU Sensor Settings” - “Misalignment correction” section, or simply copy the values found in this procedure there.

For the second motor, the divergence is additionally affected by a zero offset in the first motor encoder and the accuracy of the mechanical connection of the first and second motors, as well as the non-orthogonality of the gyroscope axes. For the third motor, the zero offset of the second motor and the accuracy of the me-

chanical connection between the second and third motors are added to them.

Due to the presence of such a number of factors, the discrepancies found in this procedure are not automatically applied and can only be used for verification and correction, since the discrepancy of the axes negatively affects the accuracy of stabilization.

2. Gyroscope scale factor calibration

This step is not required if the scale factors of the gyroscope are already calibrated in another way (or IMU already has precise factory calibrations). Accuracy up to the 4th digit is required, since the accuracy of the reference rotation measurement directly affects the accuracy of the future correction table.

To skip this step, uncheck the "**Calibrate and save gyroscope scale factor**" option. If there is still a need for calibration, then a precondition for its performance is a free angle of motor rotation, which has to be close to 360°. The found scale factors will be saved and used by the system in calibration and in normal work further. They are available for editing in the section "Hardware" - "Calibrate IMU sensors .." - "Advanced" - "Gyroscope" - "Scale factors".

3. LUT calibration

Rotation is performed in both directions, the result is averaged, filtered and saved in EEPROM. About 30 samples are used to store the LUT, between them the values are interpolated. The result is loaded from the board and displayed in a graphical form.

> **IMPORTANT!** Complete immobility of the frame must be ensured throughout the calibration . In all positions of the gimbal a stable feedback loop operation must be maintained. There should be no obstructions for free rotation of the motors according to a given program.

Applying the correction

To apply the found LUT correction, it is required to enable the corresponding checkboxes on the "Encoders" tab in "Advanced calibrations" - "Linearization LUT" group. They are automatically set after calibration performed. To disable the correction, it is necessary to uncheck the boxes or delete the LUT from the memory using the "**DELETE DATA**" button.

Correction is applied to the encoder data immediately after reading it, and further system uses the corrected angle. Therefore, it is important to calibrate the non-linearity before the other calibrations (or repeat them after this calibration).

25. 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 the "Hardware" settings.
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.
When connecting power or USB to the board, green LED doesn't light, whereas red LED lights	*GUI fails to connect:*	
	-MCU starts in the bootloader mode -Firmware isn't loaded or it's loaded a wrong version	-Check the FLASH jumper is NOT installed (only for controllers having jumper instead of tactile button) -See firmware recovery

		instructions in the section 12.Firmware update
	GUI connects:	
	- LED mode is changed from default “always on”	-Check LED mode settings in the “Service” tab
When power or USB is connected, neither green or red LED light	- 5V or 3.3V voltage regulator burned out - there is short curcuit or excessive load caused by an external device connected to 3.3v or 5V outputs	- Check voltage level on 5V and 3.3V outputs by a voltage meter; if it's below normal level and no external device is connected – its hardware failure and needs repair.

26. Appendix A: versions comparison chart

Functionality vs hardware versions

	Regular, Tiny	Extended, Extended Long	CAN_MCU	Pro	
High-end encoders Zettlex IncOder	-	●	-	●	
CAN-bus interface to the modules CAN_IMU, CAN_Driver	-	●	●	●	CAN_IMU CAN_Driver
Dedicated SPI and PWM ports for encoders connection	shared with other I/O pins	●	-	●	
External high-grade IMU - other vendors (Vectornav VN100/VN200, Inertialsense uAHRS)	-	●	●	●	section 18
External IMU - gyroscope biases online calibrations	-	●	●	●	
Advanced calibrations of gyroscope and accelerometer	-	●	●	●	section 4
Automatic multipoint accelerometer calibration	-	●	●	●	
Collect summary usage statistics in EEPROM	-	●	●	●	
Motor cogging effect calibration	-	●	●	●	section 20
Encoder non-linearity calibration		●	●	●	section 23
Support of 4th axis to align frame to optimal position	-	●	●	●	section 21
Electronic main power switch, power control and protection	-	-	●	-	
Functions common for all boards:					
I2C magnetometer	●	●	●	●	section 16
Encoder-enabled firmware available	●	●	●	●	section 15
I2C_Drv - external motor driver via I^2C bus	●	●	●	●	I2C_Drv
MavLink-connected autopilot used for gimbal control and main IMU correction	●	●	●	●	section 19
External high-grade IMU - Basecam GPS IMU	●	●	●	●	section 18

Encoder-enabled vs regular version of firmware

	Regular	Encoder-enabled
Encoders installation	YAW axis (optional)	on each motor
Encoder interfaces: I2C, PWM, Analog	●	●
Encoder interfaces: SPI	-	●

26 Appendix A: versions comparison chart

Motor control algorithm	synchronous	field-oriented	
Energy-efficient motor control	-	●	
Operation in "Follow" mode	not reliable: may loose synchronization; not precise	never loose synchronization; precise	section 8
Unlimited range of working positions (upside-down, rolled to 90, brief-case mode)	-	●	
Does not requires 2nd IMU on the frame for normal operation	-	●	section 4
Support of tilted middle or outer motor	-	●	section 4
Use external high-grade IMU located on the frame for a correction of the main IMU	○	●	section 18
Set the software-limited range of rotation for each motor	-	●	
Camera goes to home position at startup and can detect hardware limits (required for working range >360º)	-	●	

27. Appendix B: running GUI on Mac OS PC

Step-by-Step Tutorial to make GUI work on M1 Macs

Step 1: Launch the SimpleBGC_GUI.jar in the folder.

If SimpleBGC_GUI.jar icon does not launch the application, you will have two other methods:

1. Open Finder -> Find terminal in the search bar -> Open terminal -> Use this command in sequence.
   a) If the SimpleBGC_GUI folder is in the desktop, switch terminal directory by running the command

   ```
   cd desktop
   ```

   b) Open SimpleBGC_GUI folder running the command

   ```
   cd GUI
   ```

   c) Launch SimpleBGC_GUI using the command

   ```
   java -jar simplebgc_gui.jar
   ```

Image 1. Command sequence

Step 2: Connect the controller to the computer via USB.

Step 3: Select serial port

From the top of the GUI, select the appropriate serial port from the list (e.g. /dev/tty.usbserial-...) (Image 2) Try connecting to the port. If the port is not visible or if the connection fails, move on to the next step.

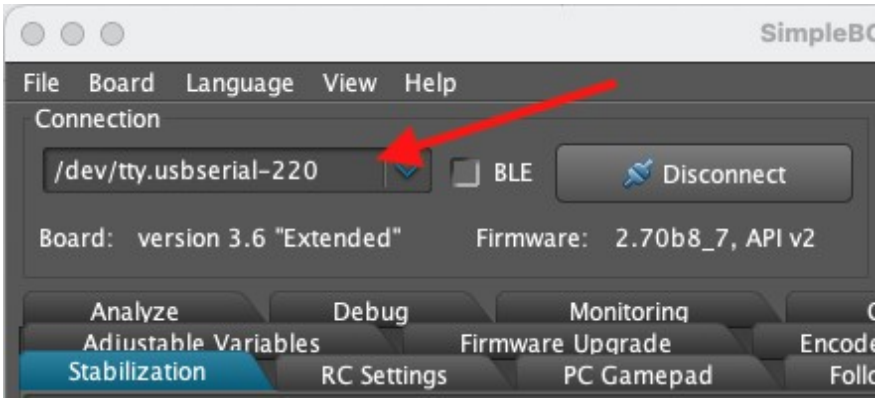


Image 2. Serial port selection

Troubleshooting

Step4: Check Java versions

- Verify versions of Java installed by running this command.

```
/usr/libexec/java_home -V
```

```
vladislavs@MacBook-Air-Vladislavs ~ % /usr/libexec/java_home -V

Matching Java Virtual Machines (7):
    19.0.2 (arm64) "Oracle Corporation" - "Java SE 19.0.2" /Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home
    19.0.2 (arm64) "Eclipse Adoptium" - "OpenJDK 19.0.2" /Library/Java/JavaVirtualMachines/temurin-19.jdk/Contents/Home
    19.0.2 (x86_64) "Azul Systems, Inc." - "Zulu 19.32.13" /Library/Java/JavaVirtualMachines/zulu-19.jdk/Contents/Home
    15.0.2 (x86_64) "Oracle Corporation" - "Java SE 15.0.2" /Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home
    11.0.18 (arm64) "Eclipse Adoptium" - "OpenJDK 11.0.18" /Library/Java/JavaVirtualMachines/temurin-11.jdk/Contents/Home
    1.8.0_362 (arm64) "Azul Systems, Inc." - "Zulu 8.68.0.21" /Library/Java/JavaVirtualMachines/zulu-8.jdk/Contents/Home
    1.8.0_362 (x86_64) "Eclipse Temurin" - "Eclipse Temurin 8" /Library/Java/JavaVirtualMachines/temurin-8.jdk/Contents/Home
/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home
```

Note: If there are too many versions installed, they may interfere with each other. For a simplicity, leave only one stable version of Java. Remove all existing Java versions using the command

```
sudo rm -rf /Library/Java/JavaVirtualMachine/
```

```
Matching Java Virtual Machines (5):
    19.0.2 (x86_64) "Azul Systems, Inc." - "Zulu 19.32.13" /Library/Java/JavaVirtualMachines/zulu-19.jdk/Contents/Home
    15.0.2 (x86_64) "Oracle Corporation" - "Java SE 15.0.2" /Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home
    11.0.18 (arm64) "Eclipse Adoptium" - "OpenJDK 11.0.18" /Library/Java/JavaVirtualMachines/temurin-11.jdk/Contents/Home
    1.8.0_362 (arm64) "Azul Systems, Inc." - "Zulu 8.68.0.21" /Library/Java/JavaVirtualMachines/zulu-8.jdk/Contents/Home
    1.8.0_362 (x86_64) "Eclipse Temurin" - "Eclipse Temurin 8" /Library/Java/JavaVirtualMachines/temurin-8.jdk/Contents/Home
/Library/Java/JavaVirtualMachines/zulu-19.jdk/Contents/Home
vladislavs@MacBook-Air-Vladislavs ~ % sudo rm -rf /Library/Java/JavaVirtualMachines/zulu-19.jdk
vladislavs@MacBook-Air-Vladislavs ~ % sudo rm -rf /Library/Java/JavaVirtualMachines/zulu-8.jdk
vladislavs@MacBook-Air-Vladislavs ~ % sudo rm -rf /Library/Java/JavaVirtualMachines/temurin-8.jdk
vladislavs@MacBook-Air-Vladislavs ~ % sudo rm -rf /Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk
vladislavs@MacBook-Air-Vladislavs ~ %
```

Step5: Giving the permissions

GUI uses a serial communication, that needs to create a lock file. Due to security constraints, you need to create the lock file yourself:

1. Start terminal (navigate to /Applications/Utilities and double click on Terminal)
2. Make new folder "/var/lock" (if not exists) using the command.

```
sudo mkdir /var/lock
```

3. Change permissions using the command

```
sudo chmod 777 /var/lock
```

4. Allow to run non-signed applications in System Preferences > Security & Privacy > General > Allow Applications downloaded from: Anywhere.

Step 6: Installing Homebrew, Rosetta 2 and correct version of Java

1. Install Rosetta 2 using the command.

```
softwareupdate --install-rosetta
```

2. Install Homebrew x64 mode using the command

```
arch -x86_64 /bin/bash -c "$(curl -fsSL
```

```
https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
```

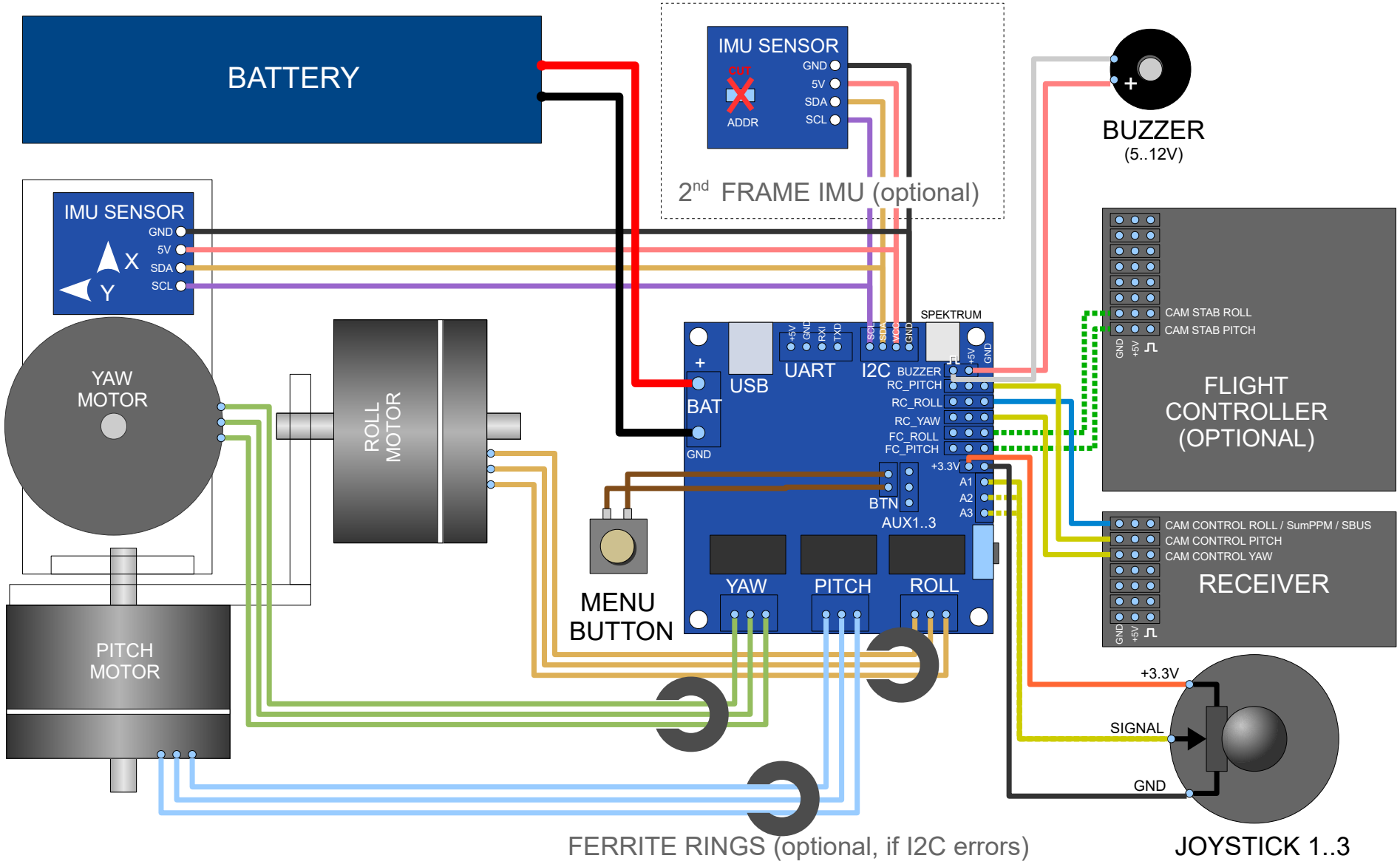
3. Add cask versions.

```
arch -x86_64 /usr/local/bin/brew tap homebrew/cask-versions
```

4. Installing stable Java version Zulu 11

```
arch -x86_64 /usr/local/bin/brew install --cask zulu11
```

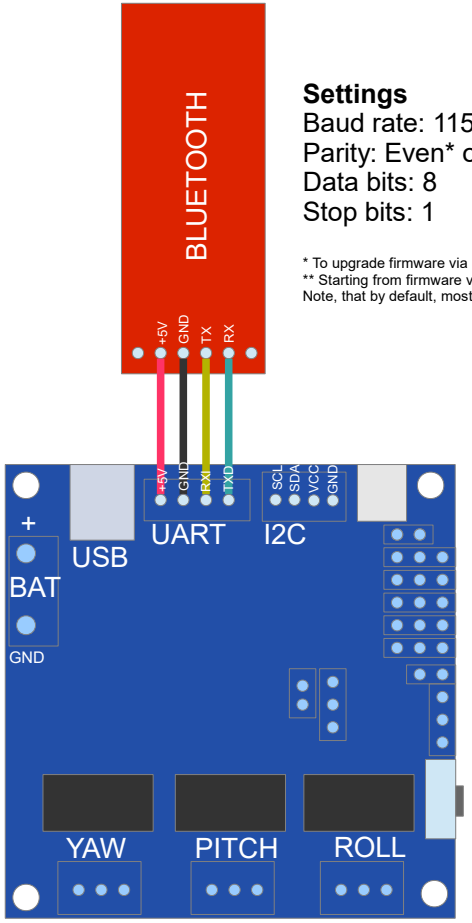
SimpleBGC 3.0 (32bit) connection diagram



© 2013-2024 Basecamelectronics®

SimpleBGC 3.0 (32bit) bluetooth connection

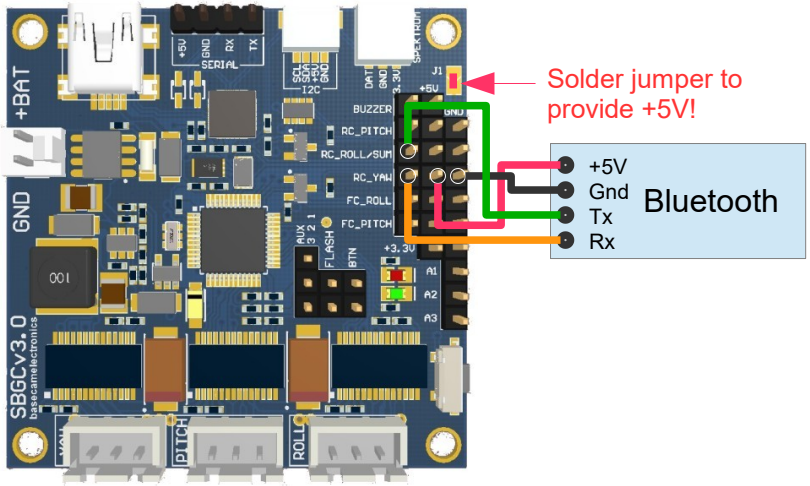
Regular connection:



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

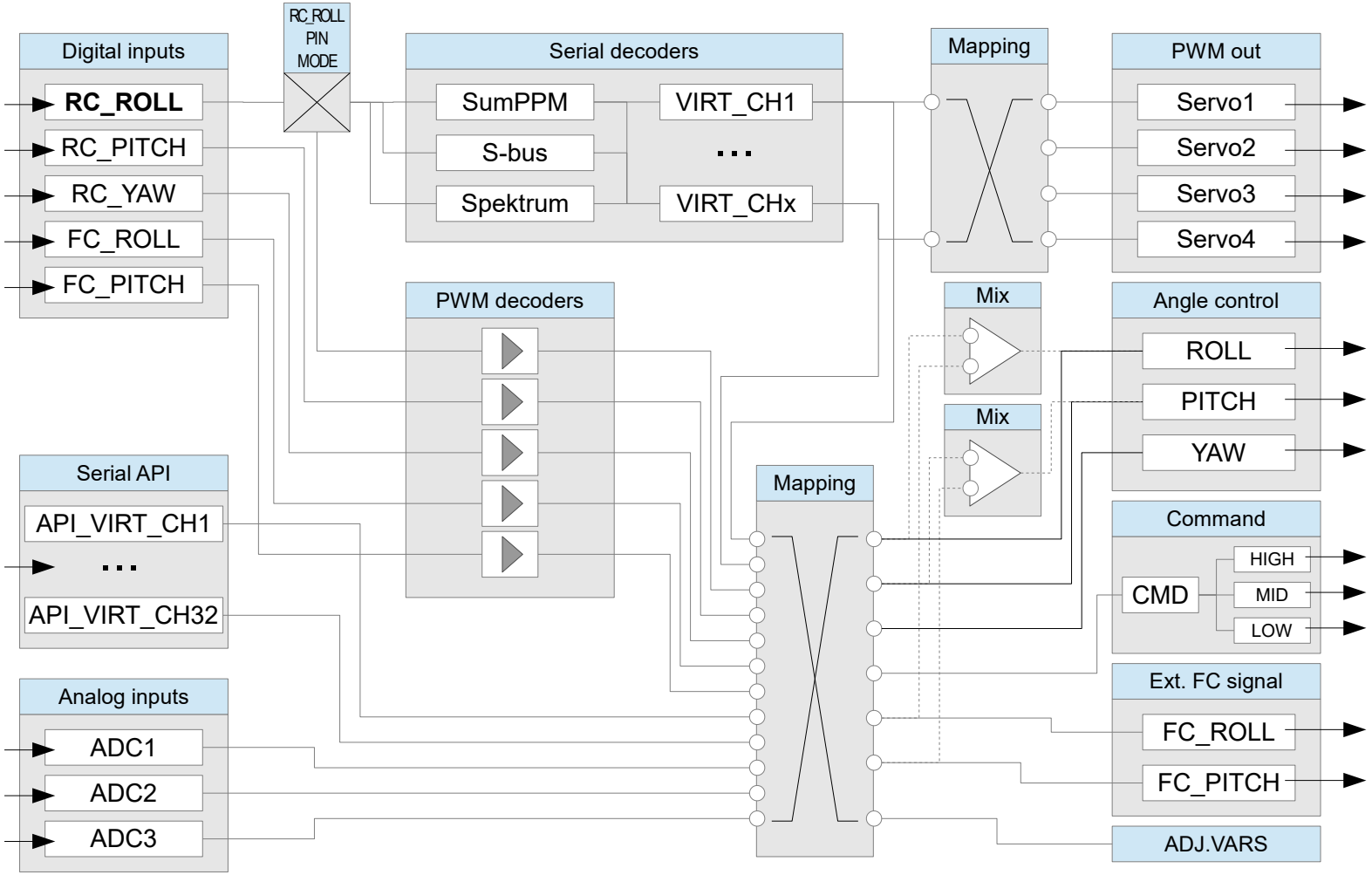
* To upgrade firmware via Bluetooth, only 'Even' parity will work.
** Starting from firmware ver 2.41, 'None' parity is supported, too.
Note, that by default, most modules configured with 'None' parity.

Optional connection:
(RC_ROLL pin mode = SBGC Serial 2nd UART)



SimpleBGC 32bit RC signal routing diagram

firmware ver. 2.43+



© 2013-2024 Basecamelectronics®