# BStemAPI-Python Documentation Release 1.0

**Brain Corporation** 

# CONTENTS

1	bsten	tem package 1				
	1.1	Subpackages	1			
	1.2	Submodules	12			
	1.3	bstem.analog_filter module	12			
	1.4	bstem.battery_check module	12			
	1.5	bstem.camera module	13			
	1.6	bstem.control module	15			
	1.7	bstem.decorator module	16			
	1.8	bstem.device module	17			
	1.9	bstem.disparity_hardware module	25			
	1.10	bstem.display_thread module	26			
	1.11	bstem.exception module	27			
	1.12	bstem.fpga module	27			
	1.13	bstem.hokuyo module	28			
	1.14	bstem.image_wrapper module	28			
	1.15	bstem.infrared module	30			
	1.16	bstem.log module	30			
	1.17	bstem.motion_hardware module	35			
	1.18	bstem.plot module	36			
	1.19	bstem.sensor module	38			
	1.20	bstem.sonar module	41			
	1.21	bstem.tracker module	41			
	1.22	bstem.video module	42			
	1.23	bstem.video_encoder module	43			
	1.24	Module contents	43			
Py	Python Module Index 4					
Inc	dex		47			

**CHAPTER** 

ONE

# **BSTEM PACKAGE**

# 1.1 Subpackages

# 1.1.1 bstem.platform package

#### **Submodules**

# bstem.platform.ad cord module

The AdCord is a bStem expansion board reference design developed by Brain Corporation for mobile robots. It is designed for wheeled platforms where precise DC motor control with quadrature feedback is required. AdCord also provides 8 RC servo outputs. To use this class first create an AdCord object:

```
ad = Adcord()
```

All servos and motors are disabled by default, to enable:

```
ad.enable_motors = True
```

For each servo (0-7) set position in the range [-1, 1] using:

```
ad.servo[0].position = 0.5
ad.servo[1].position = -0.5
```

or by setting pulse width in microseconds in the range [600, 2400]:

```
ad.servo[0].pulse_width = 600
```

For each motor (0-3) set the speed in the range [-1, 1] with:

```
ad.motor[0].speed = 0.5
ad.motor[1].speed = -0.5
```

where positive values are forward and negative values are reverse.

To read current motor position in rads:

```
ad.encoder[0].position
```

and the current motor velocity in rads/sec:

```
ad.encoder[0].position
```

# Set GPIO (0-6) direction using:

```
ad.gpio[0].direction = "in" % set GPIO to input
ad.gpio[1].direction = "out" % set GPIO to output
```

## For output GPIOs, set the current value using:

```
ad.gpio[0].value = 0
ad.gpio[1].value = 1
```

# Read the current GPIO value using:

```
value = ad.gpio[0].value
```

#### The current battery voltage can be read using:

```
ad.battery
```

The AdCord also provides access to all features of bStem (accelerometer, gyroscope, barometer, magnetometer, leds) e.g.:

```
(x, y, z) = ad.gyroscope.value
(x, y, z) = ad.accelerometer.value
...
```

#### **Parameters**

- **fpga\_image** path to AdCord fpga image
- reset reset board on creation
- **enable\_motors** enable motors on creation

#### SERVO DS = 8

(CONSTANT) Device select address for servo/pwm-out

#### battery

Property that returns battery value.

#### enable\_motors

Interface to writeable property that enables motors.

# enable\_pwm\_out

Interface to writeable property that enables pwm-out.

#### name

Returns a string reflecting the cord name (e.g., adcord)

#### reset()

Reset all servos to the mid position and reload FPGA firmware.

## bstem.platform.bstem\_platform module

```
class bstem.platform.bstem_platform.Bstem
     Bases: object
```

This class provides access to sensors and LEDs on the bStem board. To use first create a Bstem object:

```
b = Bstem()
```

Read the gyroscope using:

```
(x, y, z) = b.gyroscope.value
```

Set gyroscope sensitivity in degrees per second (250, 500 or 2000) using:

```
self.gyroscope.dps = 250
```

Read the accelerometer using:

```
(x, y, z) = b.accelerometer.value
```

Read the magnetometer using:

```
(x, y, z) = b.magnetometer.value
```

Read the barometer using:

```
pressure = b.barometer.pressure
```

Set the LEDS (blue, green, yellow) using:

# bstem.platform.lg\_cord module

The LgCord is a beta bStem expansion board reference developed by Brain Corporation for Lego/Mindstorm robots. In combination with a bStem, it can be put in place of Lego's NXT or EV3 controller. It can control up to 6 motors and read from 2 or more sensors (with multiplexing.) Lego cord.

To use this class first create an LgCord object:

```
lg = LgCord()
```

For each motor (0-5) set the speed in the range [-1, 1] with:

```
lg.motor[0].speed = 0.5
lg.motor[1].speed = -0.5
```

where positive values are forward and negative values are reverse.

The LgCord also provides access to all features of bStem (accelerometer, gyroscope, barometer, magnetometer, leds) e.g.:

```
(x, y, z) = lg.gyroscope.value
(x, y, z) = lg.accelerometer.value
...
```

# **Parameters**

- fpga\_image path to LgCord fpga image
- reset reset board on creation

1.1. Subpackages 3

• enable motors – enable motors on creation

```
battery
enable_motors
reset()
    Reload FPGA firmware.
```

# bstem.platform.rc cord module

```
 \begin{array}{c} \textbf{class} \ \texttt{bstem.platform.rc\_cord.RcCord} \ (\textit{fpga\_image='/usr/share/bstem/rccord\_bitmap.bin'}, \\ reset=True, & enable\_servo\_power=False, \\ able\_pwm\_out=True) \\ \textbf{Bases:} \ \textit{bstem.platform.bstem\_platform.Bstem} \end{array}
```

The RCCord is a bStem expansion board reference design developed by Brain Corporation for radio-control robotic applications. RcCord is designed to control traditional RC vehicles, like RC cars, planes, and quad-copters. RcCord can read signals coming from RC receiver units and can output RC servo pulses. To use this class first create an RcCord object:

```
rc = RcCord()
```

All servos are disabled by default, to enable:

```
rc.enable_servo_power = True
```

For each servo (0-8) set position in the range [-1, 1] using:

```
rc.servo[0].position = 0.5
rc.servo[1].position = -0.5
```

or by setting pulse width in microseconds in the range [600, 2400]:

```
rc.servo[0].pulse_width = 600
```

Set GPIO (0-8) direction using:

```
rc.gpio[0].direction = "in" % set GPIO to input
rc.gpio[1].direction = "out" % set GPIO to output
```

For output GPIOs, set the current value using:

```
rc.gpio[0].value = 0
rc.gpio[1].value = 1
```

Read the current GPIO value using:

```
value = rc.gpio[0].value
```

The current battery voltage can be read using:

```
rc.battery
```

The RcCord also provides access to all features of bStem (accelerometer, gyroscope, barometer, magnetometer, leds) e.g.:

```
(x, y, z) = rc.gyroscope.value
(x, y, z) = rc.accelerometer.value
...
```

#### **Parameters**

```
    fpga_image – path to RcCord fpga image
    reset – reset board on creation
```

• enable\_servo\_power - enable power to servo IO ports

```
ADC_DS = 24
```

(CONSTANT) Part of device select address for rcCord

PPM DS = 2

(CONSTANT) Part of device select address for rcCord

RC DS = 0

(CONSTANT) Part of device select address for rcCord

RC\_MULTI\_DS = 1

(CONSTANT) Part of device select address for rcCord

 $SERVO_DS = [8, 9, 10, 11, 12, 13, 14, 15]$ 

(CONSTANT) Part of device select address for rcCord

battery

enable\_pwm\_out

enable\_servo\_power

name

Returns the cord name

reset()

Reset all servos to the mid position and reload FPGA firmware.

#### **Module contents**

# 1.1.2 bstem.test package

#### **Submodules**

# bstem.test.augmented\_disparity module

# bstem.test.cord\_info module

```
bstem.test.cord_info.find_file (name, path)
Searches for file name in given path.
```

```
bstem.test.cord_info.get_bstem_cord(reset=True)
```

Finds the type of cord from file /usr/share/bstem/.adcord or .rccord or .lgcord created by the user and returns a default cord accordingly. User need to override the required cord parameters.

```
bstem.test.cord_info.get_cord_property()
```

Finds the type of cord based on the file /usr/share/bstem/.adcord or .rccord or .lgcord as created by the user. The function also returns appropriate fpga binary image file. When bstem\_fpga folder is available within the bstem folder, then specific binary image is used, else default system binary image is used.

1.1. Subpackages 5

### bstem.test.demo video encoder module

#### bstem.test.test bstem module

```
{\bf class} \; {\tt bstem.test.test\_bstem.TestBstem}
```

```
test_version()
```

#### bstem.test.test camera module

#### bstem.test.test control module

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

```
test_multiple_control_loops()
test_single_control_loop()
```

#### bstem.test.test gyro and accel reading speed module

# bstem.test.test image wrapper module

```
bstem.test.test_image_wrapper.test_YUV_forms()
bstem.test.test_image_wrapper.test_color_conversions()
bstem.test.test_image_wrapper.test_crop()
bstem.test.test_image_wrapper.test_reshape_and_types()
bstem.test.test_image_wrapper.validate_crop(YUV_im, x, y, width, height)
```

#### bstem.test.test led module

#### bstem.test.test\_log module

# bstem.test.test\_platform module

# bstem.test.test\_plot module

```
class bstem.test.test_plot.TestPlot (methodName='runTest')
    Bases; unittest.case.TestCase
```

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

```
test_histogram()
```

```
test_histogram_remote()
   test_histogram_remote_passive()
   test_lineplot()
   test_lineplot_remote()
   test_lineplot_remote_passive()

bstem.test.test_plot.plot_rand()

bstem.test.test_plot.random() \rightarrow x in the interval [0, 1).

bstem.test.test_plot.simple_fxn()

bstem.test.test_plot.sin_tuple_fxn()
```

bstem.test.test ppm module

bstem.test.test range sensor module

bstem.test.test\_rc\_and\_adcord module

bstem.test.test sensor module

bstem.test.test\_servo module

bstem.test.test tracker module

```
class bstem.test.test_tracker.TestTracker (methodName='runTest')
    Bases: unittest.case.TestCase
```

Test to ensure that the actual bounding box of the object and the CAMshift calculated bounding box of the object are within the expected delta.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

```
test_tracker()
bstem.test.test_tracker.generate_sample_video()
Generates a video for testing the CAMshift based Tracker for Bstem
```

bstem.test.test tutorials module

bstem.test.test video encoder module

**Module contents** 

# 1.1.3 bstem.tutorials package

**Submodules** 

bstem.tutorials.demo\_sonar\_ir module

```
bstem.tutorials.demo_sonar_ir.demo_range_sensor(sensor_type, num_data=1000) read and process various range sensors
```

1.1. Subpackages 7

### bstem.tutorials.example 10 tracking module

```
class bstem.tutorials.example_10_tracking.TrackerApp
    run()
         Run the tracker app
```

## bstem.tutorials.example\_11\_disparity module

This tutorial shows how to use the disparity hardware with the stereo cameras.

Before running this tutorial, please confirm that both cameras are plugged in.

```
bstem.tutorials.example_11_disparity.hardware_disparity_example (num_frames=1000,
                                                                      cam override=None)
```

Compute disparity using hardware acceleration. The main pro is that one gets reatively detailed disparity estimates with fairly low cpu cost. However, this method is prone to significant noise and as a result may be difficult to use. Noise can be both large holes where disparity can not be reliably computed as well as erroneously estimating objects as near when they are in fact far due to false matches.

```
bstem.tutorials.example_11_disparity.show_disparity(disparity, size)
bstem.tutorials.example_11_disparity.software_disparity_example (num_frames=1000,
                                                                      cam override=None)
```

Compute disparity using software (OpenCV StereoSGBM). The main pro of this method is that it has low noise and fills in pretty well with cpu load comparable to the hardware example. this method is quite blurry and reducing the blurring increases the noise. For more information see: http://docs.opencv.org/modules/calib3d/doc/camera calibration and 3d reconstruction.html?highlight=stereosgbm#cv2.StereoS

#### bstem.tutorials.example 12 audio module

#### bstem.tutorials.example 1 hello world module

```
Example: bstem LEDs Updated 1/21/2014
```

```
bstem.tutorials.example 1 hello world.led example()
```

#### bstem.tutorials.example 2 sensors module

```
Example: bstem Sensors Updated 1/21/2014
```

```
bstem.tutorials.example_2_sensors.bstem_alternate_sensor_examples()
     Shows an alternative way to access the sensors, through individual sensor APIs
bstem.tutorials.example_2_sensors.bstem_sensor_examples()
     Shows the standard way to access the sensors (through Bstem platform class)
```

# bstem.tutorials.example 3a camera module

#### Example: bstem Cameras Updated 1/21/2014

```
bstem.tutorials.example 3a camera.camera example (num frames=50, cam=None)
```

# bstem.tutorials.example\_3b\_video\_recording module

#### Example: bstem Video Recording Updated 1/21/2014

```
bstem.tutorials.example_3b_video_recording.video_recording_example (duration, file_name, cam=None)
```

A simple example demonstrates the recording capability.

To play back use the command:

>> gst-launch filesrc location=[filename]! decodebin! ffmpegcolorspace! ximagesink

## bstem.tutorials.example\_3c\_camera\_config module

#### Example: bstem camera config Updated 2/5/2014

```
bstem.tutorials.example_3c_camera_config.camera_config_example()
Configure each of the properties of the camera in turn.

bstem.tutorials.example_3c_camera_config.show_video(cam)
Show 20ms of video.
```

#### bstem.tutorials.example 4 scheduler module

# Example: bSTEM Control (Scheduler and ControlLoop) Updated 1/21/2014

```
class bstem.tutorials.example_4_scheduler.Acceler(freq)
    Bases: bstem.control.ControlLoop

Reads and integrates the accelerometer
loop()

class bstem.tutorials.example_4_scheduler.Blinker(freq)
    Bases: bstem.control.ControlLoop

Turns on and off an LED
loop()

class bstem.tutorials.example_4_scheduler.Clunker(freq, compute_time)
    Bases: bstem.control.ControlLoop
    simulates slow code by waiting
loop()

bstem.tutorials.example_4_scheduler.timed_loop_example()
```

#### bstem.tutorials.example 5a local logging module

# Example: bSTEM Logging Updated 1/21/2014

```
bstem.tutorials.example_5a_local_logging.collect_data()
bstem.tutorials.example_5a_local_logging.filelogger_example()
bstem.tutorials.example_5a_local_logging.logger_writeToCsv_example()
bstem.tutorials.example_5a_local_logging.main()
```

1.1. Subpackages 9

# bstem.tutorials.example\_5b\_remote\_logging\_client\_server module

# Example: bSTEM Remote Logging [on bstem] Updated 1/21/2014

```
bstem.tutorials.example_5b_remote_logging_client_server.main()
```

# bstem.tutorials.example\_6a\_local\_plotting module

#### Example: bSTEM Remote Plotting Updated 1/21/2014

```
bstem.tutorials.example_6a_local_plotting.main()
bstem.tutorials.example_6a_local_plotting.plot_local_example()
bstem.tutorials.example_6a_local_plotting.plot_rainbow_example()
```

# bstem.tutorials.example\_6b\_remote\_plot\_client\_server module

# Example: bSTEM Remote Plotting [on bstem] Updated 1/21/2014

```
bstem.tutorials.example_6b_remote_plot_client_server.main()
```

### bstem.tutorials.example\_7\_parameter\_tuning module

```
bstem.tutorials.example_7_parameter_tuning.main()
bstem.tutorials.example_7_parameter_tuning.variable_server_example()
```

#### bstem.tutorials.logging client module

#### Example: bSTEM Remote Logging [on PC] Updated 1/21/2014

```
bstem.tutorials.logging_client.collect_data()
bstem.tutorials.logging_client.remotelogger_client(ip, port)
```

# bstem.tutorials.logging\_server module

# Example: bSTEM Remote Logging [on PC] Updated 1/21/2014

```
bstem.tutorials.logging_server.print_logserver_data(logserver, fields) Helper function to print data from a logserver.
```

#### **Parameters**

- **logserver** (*LogServer instance*) Server containing data to be printed
- **fields** (String or Iterarble of Strings) Fields (Sensor names) to print

```
bstem.tutorials.logging_server.remotelogger_server(port)
```

# bstem.tutorials.remoteplot\_client module

```
bstem.tutorials.remoteplot_client.remoteplot_client(ip, port)
```

# bstem.tutorials.remoteplot\_server module

```
bstem.tutorials.remoteplot_server.remoteplot_server(port)
```

#### Module contents

# 1.1.4 bstem.utils package

#### **Submodules**

bstem.utils.analyze\_sonar\_dataset module

# bstem.utils.compass\_calibration module

```
class bstem.utils.compass_calibration.CompassCalibration
```

Calibration returns axis specific Offset and Magnitude measurements, which are incorporated into the Magnetometer.value() Direct uncalibrated values can still be obtained from the Bstem Sensors .raw\_value() or by setting calibrated=False when reading from Magnetometer. Successive Calibrations are appended to the /usr/share/bstem/.compass\_calibration.log file The most recent calibration will be used to report calibrated values.

**classmethod calibrate** (calibration time sec=10, plot results=False, wait for keyboard=True)

## bstem.utils.demo\_ppm\_out module

#### bstem.utils.get\_min\_max\_ppm module

Utility script for determining the maximum and minimum values observed on the bStem PPM inputs. Once the script is running, the user adjust the full range of the control signals going into the PPM input. When the user hits "return", the maximum and minimum values for all 8 channels over the sample period are displayed.

```
bstem.utils.get_min_max_ppm.get_min_max_ppm_values(poll_user_exit=True, cord=None, timeout_s=None)
```

Polls all 8 PPM input channels and keeps track of the maximum and minimum values of each channel.

The get\_min\_max\_ppm\_values function polls all 8 PPM input channels over a period of time to determine the maximum and minimum value for each channel that is observed over the sampling interval. The sampling period is specified by *timeout\_s* or until the user hits return. The maximum and minimum values for each channel are then displayed on the screen.

poll\_user\_exit [{ True, False }, optional] Check for user exit request (return button) on each sample interval

**cord** [bStem RcCord, optional] The cord that contains the PPM input channels. Default will instantiate a new RcCord object

timeout\_s [float, optional] The sampling interval timeout in seconds. The default is no timeout and to wait for the user to hit return to exit

nothing

```
bstem.utils.get_min_max_ppm.hit_return()
Check if user hit return on the keyboard
```

**boolean** True if the user hit return. False otherwise

This only works on linux systems and fails on py.test with error: ValueError: redirected Stdin is pseudofile, has no fileno(). *poll\_user\_exit* added to get\_min\_max\_ppm\_values() to avoid execution in test\_ppm.py unit test

1.1. Subpackages 11

bstem.utils.make\_sonar\_dataset module

bstem.utils.ppm repeater module

**Module contents** 

# 1.2 Submodules

# 1.3 bstem.analog\_filter module

```
class bstem.analog_filter.AnalogFilter(cord,
                                                                               mapping_function,
                                                                                                      fil-
                                                              analog_pin,
                                                    ter_type='MEDIAN', buffer_length=5)
     Bases: object
     A generic base analogiliter class that supports various types of filtering data read from adc in various cord.
     Samples can be continiously added and various statistics can be calulated.
     Notes: - We need to build support for fixed-frequency polling of sensors.
     cord: :class: RcCord RcCord object
     analog_pin [int] Pin to which the sensor is attached
     mapping function [method] 1D Mapping function: Voltage -> Distance Polynomial and coefficients for map-
           ping voltage to distance
     filter_type [str] Filter type for processing samples (MEDIAN or MODE or NONE). When NONE use recent
           sample
     buffer_length [int] size of circular buffer for median filter
     distance
           out [int] The distance estimate using filtered data and mapping function
     get_last_sample()
           returns the last collected analog voltage sample
     get_samples()
           returns entire collected sample for user specific processing
     raw value
           out [float] The raw value of analog voltage estimates of recent sample
```

# 1.4 bstem.battery\_check module

```
bstem.battery_check.do_battery_check (dev, warning_level, shutdown_level, daemon=False, sleep_time=60)
    check the battery level and report warning/shutdown conditions to protect battery. if daemon=True than the program runs in background. sleep_time in seconds enable how often we need to check the battery level and warn the user.

bstem.battery_check.generate_shutdown_message(level)
bstem.battery_check.generate_warning_message(level)
bstem.battery_check.run_daemon(dev, warning_level, shutdown_level)
```

# 1.5 bstem.camera module

 $\textbf{class} \ \texttt{bstem.camera.Camera} \ (output\_format='BGR', \ video\_size=None, \ auto\_pause\_threshold=2, \ number\_of\_snapshots\_after\_an\_update=10, stereo\_mode=True)$ 

Bases: bstem.sensor.Sensor

Camera module implements the interface to bSTEM's stereo cameras. It provides interface for taking snapshots and making video recordings.

Interface for configuring and reading properties of the on-board stereo cameras of bStem such as brightness, saturation, contrast and hue. Currently each setting is applied to both cameras. A combination of settings that removes adaptation is available via the remove\_adaptation method.

Currently, multiple consecutive snapshots and single recording work.

**Some failure modes include::** record; snapshot; snapshot -> second snapshot hangs at starting up snapshot; record -> record hangs at the end record; record -> first record works, the second record hands at the end

Initialize a camera object :param output\_format: determine snapshot return image image format. "RGB" or "BGR" :type output\_format: string :param auto\_pause\_threshold: if the interval between calling snapshot is greater than auto\_pause\_threshold \* 40ms then auto-pausing will be triggered to save on CPU time :type auto\_pause\_threshold: float :param number\_of\_snapshots\_after\_an\_update: define how many snapshot we will take after changing configuration of the camera (default is 10) :type number\_of\_snapshots\_after\_an\_update: int

#### atexit()

helper method to handle python shutdown with extra threads correctly

#### auto\_exposure

Get autoexposure mode

**Returns** True: auto exposure is enabled False: auto exposure is disabled

Return type bool

#### auto\_white\_balance

Get auto white balance mode

Returns True: auto white balance is enabled False: auto white balance is disabled

Return type bool

#### brightness

Get image brightness

**Returns** brightness

**Return type** int

#### central\_auto\_exposure

True if auto exposure is using only the central region of the image.

**Returns** True: central auto exposure is enabled False: central auto exposure is disabled

Return type bool

cleanup()

#### contrast

Get image contrast

Returns contrast

Return type int

#### end recording()

Ends the recording. Stopping gstreamer pipeline.

#### fade\_to\_black

Get fade-to-black dampening factor

**Returns** dampening factor

Return type int

# gain

Get gain, only meaningful when auto exposure is off

Returns gain

Return type int

#### hue

Get image hue

Returns hue

Return type int

#### pause()

Call pause() to pause the pipeline. This can substantially reduce the cpu load if snapshot is not going to be immediately called again. If snapshot is going to be called in a tight loop at or near the frame rate of the camera then pause should not be called.

#### remove\_adaptation()

A combination of parameters that removes all automatic adaptation to lighting conditions and sets a reasonable gain

# saturation

Get image saturation

**Returns** saturation

Return type int

#### sfx

Get SFX control

**Returns** 'disabled' SFX mode disabled 'monochrome' Monchrome 'sepia' Sepia 'negative' Negative image 'solarization' Solarization with unmodified UV 'solarization\_neg\_uv' Solarization with negative UV

Return type string

#### sharpness

Get image sharpness

**Returns** sharpness

Return type int

# snapshot (img\_prefix=None, disable\_auto\_pause=False)

Take a snapshot. Save to file if flag is set Returns a tuple (left\_image, right\_image) of ImageWrapper instances. The format of the returned ImageWrapper is determined by the output\_format the camera is configured to use. ImageWrapper can be treated like a normal ndarray.

- img\_prefix (string) If specified, snapshots will be saved as prefix\_[leftlright].jpg.
- disable\_auto\_pause (bool) if True, don't ever pause the pipeline

```
start_recording(file_name, bitrate=1000000)
```

Starts the recording. Starting gstreamer pipeline. :param file\_name: the video file path that ends with '.mp4' :type file name: string

write\_images (img\_prefix, left\_im, right\_im)

# 1.6 bstem.control module

class bstem.control.ControlLoop (freq=100, tolerance=0.001)

Bases: object

A single control loop.

Initialize parameters of control loop.

#### **Parameters**

- **freq** (int) frequency in Hz that loop function will be called
- tolerance (double) -
- **tolerance** acceptable scheduling error e.g. the loop may execute within [-tolerance, tolerance] *ms* of expected time for any iteration of the loop

# avg\_error

Returns average error in ms of loop iteration

Return type float

avg\_exec\_time

**Returns** average execution time in ms of loop iteration

Return type float

control\_vars\_reset (start\_time)

Reset all counters.

**Parameters** start\_time (long) – loop start time

error

Returns error in ms of last loop iteration

Return type float

exec\_time

**Returns** execution time in ms of last loop iteration

Return type float

freq

**Returns** loop frequency in Hz

Return type int

iter

Returns the current loop iteration

Return type int

#### loop()

This must be overridden by each class that inherits from ControlLoop to implement the main functionality of the loop

#### period

**Returns** loop period in ms

Return type float

remaining(cur\_time)

**Returns** time in ms remaining until next iteration of the loop

Return type int

update(cur\_time)

Called by the controller to schedule loop. Checks if loop needs to run, if so run and update time remaining to next iteration.

Parameters cur\_time (float) - the current time

#### class bstem.control.Scheduler

Bases: object

Container for control loops associated with a single device. Responsible for scheduling executing of all control loops associated with that device.

#### classmethod add (\*args, \*\*kwargs)

Add a new control loop to the device.

**Parameters 100p** (*subclass of ControlLoop*) – loop construct defining periodic behavior and frequency

# classmethod remove (\*args, \*\*kwargs)

Remove an existing control loop.

Parameters loop (subclass of ControlLoop) – loop already added to Scheduler

#### classmethod start (\*args, \*\*kwargs)

Start execution of all control loops.

Parameters time\_limit (int/float) – length (in seconds) of time to run Scheduler

# classmethod suspend (\*args, \*\*kwargs)

Suspend execution of all control loops. Call start() to resume.

# 1.7 bstem.decorator module

bstem.decorator.singletonmethod(orig\_func)

Initialize singleton instance when method is called if it has not been initialized already.

**Parameters orig\_func** (*function*) – Method intended to affect the singleton instance of object, creating an instance if one does not exist already.

Returns Modified orig\_function.

Return type function

If **Class** has *function(...)* defined with decorator @*singletonmethod*, calling **Class**.*function(...)* will behave as follows:

```
if Class._instance is None:
    Class._instance = Class._Impl()
    Class.function(...) # Usually affects Class._instance
else:
    Class.function(...) # Usually affects Class._instance
```

# 1.8 bstem.device module

Absolute position encoder calculated via a potentiometer

Absolute position encoder.

#### **Parameters**

- fpga (bstem.Fpga) FPGA object for SPI communication
- device\_select (unsigned byte) id of device on the FPGA

#### gain

# position

Get position.

**Returns** position in clock ticks

Return type int

```
class bstem.device.AdcAdCord (fpga, device_select)
```

Bases: object

Analog to Digital converter for the AdCord

Analog-to-digital encoder.

#### **Parameters**

- **fpga** (*bstem.Fpga*) FPGA object for SPI communication
- device\_select (unsigned byte) id of device on the FPGA

```
ADC_DEV_BATTERY = 144
```

 $ADC_DEV_MOTOR_0 = 32$ 

ADC\_DEV\_MOTOR\_1 = 160

 $ADC_DEV_MOTOR_2 = 0$ 

 $ADC_DEV_MOTOR_3 = 16$ 

value (dev=None, pin=None)

Retrieve value from encoder.

- val (int) encoder value
- **dev** (*float* (*BATTERY* = *Voltage*, *MOTOR* = *Current*)) device id (ADC\_DEV\_BATTERY, ADC\_DEV\_MOTOR\_1, ...)

class bstem.device.AdcLgCord (fpga, device\_select)

Bases: object

Analog to Digital converter for the LgCord

Analog-to-digital encoder.

#### **Parameters**

- fpga (bstem.Fpga) FPGA object for SPI communication
- device\_select (unsigned byte) id of device on the FPGA

 $ADC_DEV_MOTOR_0 = 32$ 

value(dev)

Retrieve value from encoder.

#### **Parameters**

- val (int) encoder value
- **dev** (*float* (*BATTERY* = *Voltage*, *MOTOR* = *Current*)) device id (ADC\_DEV\_BATTERY, ADC\_DEV\_MOTOR\_1, ...)

class bstem.device.AdcRcCord (fpga, device\_select)

Bases: object

Analog to Digital converter for the RcCord

Analog-to-digital encoder.

#### **Parameters**

- fpga (bstem.Fpga) FPGA object for SPI communication
- device\_select (unsigned byte) id of device on the FPGA

ADC\_DEV\_BATTERY = 10

value (pin)

Public interface for reading analog input on the RcCord. port: pin number

class bstem.device.Encoder (fpga, device\_select, ticks\_per\_rev=1200)

Bases: object

Interface to read motor information - position and velocity

Quadrature encoder.

#### **Parameters**

- **fpga** (*bstem.Fpga*) FPGA object for SPI communication
- device\_select (unsigned byte) id of device on the FPGA
- ticks\_per\_rev (int (default 1200)) number of encoder ticks per wheel revolution

calc\_velocity(vel, dt)

Calculate motor velocity.

#### **Parameters**

- vel (float) raw encoder postion returned from the FPGA
- dt time step

Returns velocity in rads/s

# Return type float

#### convert\_to\_clicks(val)

Convert encoder rads to clicks.

**Parameters** val (*float*) – angle in radians

Return type int

#### convert\_to\_rads (val)

Convert encoder clicks to rads.

**Parameters** val (*int*) – angle in clicks

Return type float

#### position

Encoder position in radians.

**Returns** position in radians

Return type float

#### position\_clicks

Encoder position in integer number of half-degrees. One full rotation takes 720 ticks.

**Returns** position in radians

Return type float

# velocity

Encoder velocity.

**Returns** velocity in rads/s

Return type float

Bases: object

General purpose input/output pin.

#### **Parameters**

- **fpga** (*bstem.Fpga*) FPGA object for SPI communication
- device\_select\_in (unsigned byte) id of device on the FPGA for input
- device\_select\_out (unsigned byte) id of device on the FPGA for output
- pin\_number (unsigned byte) pin number that is controlled on the FPGA

#### direction

Return current direction for the gpio pin

Returns 0/1

Return type int

#### value

Current value of gpio pin.

Returns 0/1

Return type int

```
class bstem.device.Led(name)
     Bases: object
     Interface to control LEDs on the bSTEM
     LED.
          Parameters name (string) – Led name on file system e.g. "red", "yellow", "blue"
     brightness
          Led brightness.
              Returns Led brightness in range [0,2]
              Return type int
class bstem.device.LgServo (fpga, device_select)
     Bases: object
     Servo interface - position and pulse_width
     Pulse width controlled servo.
          Parameters
                • fpga (bstem.Fpga) – FPGA object for SPI communication
                • device_select (unsigned byte) - id of device on the FPGA
     position
     pulse_width
class bstem.device.Motor (fpga, device_select)
     Bases: object
     Interface to motors
     Pulse-width modulation (PWM) motor driver.
          Parameters
                • fpga (bstem.Fpga) – FPGA object for SPI communication
                • device_select (unsigned byte) - id of device on the FPGA
     disable()
          Disable motor.
              Returns device status
              Return type int
     reset()
          Reset motor after error or disable.
              Returns device status
              Return type int
     speed
class bstem.device.MulticolorLed (fpga, device_select)
     Bases: object
     Interface to control external multicolor LEDs. Currently supports Pololu #2546, see:
     http://www.pololu.com/product/2546
```

Multi-colored LED.

#### **Parameters**

- fpga (bstem.Fpga) FPGA object for SPI communication
- device\_select (unsigned byte) id of device on the FPGA

color

class bstem.device.PPMOutput (fpga, device select)

Bases: object

Pulse-width generator using PPM to combine all 8 channels to one line of output. The minimum pulse-width using PPM is fixed at 0.5ms per channel. The interchannel pulse-width stop period is 0.4ms. There will be a total of 8 channels and PPM widht is updated very 18ms corresponding. Currently ppm position based API is not supported and only pure pulse-width based interface is supported.

fpga [FPGA]:FPGA object

device\_select [int] device select specific pin used to enable ppm slave

get pulse width(channel)

Get servo servo pulse-width in millisecond

pulse\_width(channel, val)

**channel** [int] channel number for calculating the pulse-width [range: 0 upto 7]

**value** [int] pulse-width that need to be output to specific channel in microsecond Expected range for val is 0.5 to 1.7 (between 0.5ms to 1.7ms) Hard-coded minimum of 0.5ms set in the FPGA code (if counter is zero all the ppm output will be set to 0.5ms pulse per channel).

class bstem.device.PowerGauge (fpga, device\_select)

Bases: object

Texas Instruments bq2050H Lithium Ion Power Gauge

Lithium-Ion Power Gauge.

#### **Parameters**

- fpga (bstem.Fpga) FPGA object for SPI communication
- device\_select (unsigned byte) id of device on the FPGA

id

Read battery ID register. Result will be returned by next call to PowerGauge after approximately 4ms.

**Returns** Result of previous call to PowerGauge

Return type unsigned byte

#### max\_cell\_voltage

Read maximum cell voltage register. Result will be returned by next call to PowerGauge after approximately 4ms.

**Returns** Result of previous call to PowerGauge

**Return type** unsigned byte

#### primary\_status

Read primary status register. Result will be returned by next call to PowerGauge after approximately 4ms.

bit 7: Charge status flag, asserted when charge rate detected

bit 6: Battery replaced flag, asserted whenever bq2050h is reset

bit 5: Protector status flag, status of the overvoltage protector:

```
0 = PSTAT input low (< 0.5 V)
1 = PSTAT input high (>2.5 V)
asserted when PSTAT high and cleared when low
```

**bit 4** [Capacity inaccurate flag, used to warn user when battery has been] been charged a substantial number of times since LMD (Last Measured Discharge):

```
0 when LMD update with full valid discharge
1 after 64th valid charge with no LMD update
```

- bit 3: Valid discharge flag
- bit 2: First end-of-discharge warning, warns user when battery almost empty
- bit 1 [Final end of discharge warning, warns user when battery at failure] condition

Returns Result of previous call to PowerGauge

Return type unsigned byte

#### secondary\_status

Read secondary status register. Result will be returned by next call to PowerGauge after approximately 4ms.

bits 6-4: Discharge rate:

```
0 0 0 : drate < 0.5C
0 0 1 : 0.5C < drate < 2C
0 1 0 : 2C < drate
```

- bit 3: Enable interupt flag, test bit used to determine Vsr activity
- bit 2: Valid charge flag, valid charge condition
- bit 0: Overload flag, asserted when a charge rate in excess of 2C detected

Returns Result of previous call to PowerGauge

Return type unsigned byte

#### temperature

Read temperature register. Result will be returned by next call to PowerGauge after approximately 4ms.

**Returns** Result of previous call to PowerGauge

Return type unsigned byte

# static to\_temp\_range (val)

Convert temperature register value to temperature range.

**Parameters val** (*unsigned byte*) – Temperature register value

**Returns** Temperature range

Return type (int, int)

#### static to\_voltage (val)

Convert battery voltage register value to voltage.

**Parameters val** (*unsigned byte*) – Battery register

Returns Voltage

#### Return type float

#### voltage

Read battery voltage register. Result will be returned by next call to PowerGauge after approximately 4ms.

Returns Result of previous call to PowerGauge

Return type unsigned byte

class bstem.device.QuadEncoder (fpga, device\_select, ticks\_per\_rev=1200)

Bases: object

Interface to read motor information from a quadrature encoders - position and velocity

Quadrature encoder.

#### **Parameters**

- fpga (bstem.Fpga) FPGA object for SPI communication
- device\_select (unsigned byte) id of device on the FPGA
- ticks\_per\_rev (int (default 1200)) number of encoder ticks per wheel revolution

#### calc\_velocity(vel, dt)

Calculate motor velocity.

#### **Parameters**

- vel (float) raw encoder postion returned from the FPGA
- dt time step

**Returns** velocity in rads/s

Return type float

#### convert to clicks (val)

Convert encoder rads to clicks.

**Parameters** val (*float*) – angle in radians

Return type int

#### convert\_to\_rads(val)

Convert encoder clicks to rads.

**Parameters val** (*int*) – angle in clicks

Return type float

#### position

Encoder position in degrees. One full rotation increments/decrements with 2 x pi.

**Returns** position in radians

Return type float

#### position\_clicks

Encoder position in integer number of half-degrees. One full rotation takes 720 ticks.

**Returns** position in clicks

Return type int

#### velocity

Encoder velocity.

**Returns** velocity in rads/s

#### Return type float

class bstem.device.Rc (fpga, device\_select, channel)

Bases: object

Pulse width generator

FrSky D4R-II eight channel RC controller.

#### **Parameters**

- **fpga** (*bstem.Fpga*) FPGA object for SPI communication
- device\_select (unsigned byte) id of device on the FPGA
- channel (unsigned byte) RC channel

#### pulse\_width

Current value of gpio pin.

Returns 0/1

Return type int

```
class bstem.device.Servo (fpga, device\_select, pin\_number, pwm\_frequency\_hz=50, pulse\_width\_min\_us=600, pulse\_width\_max\_us=2400)
```

Bases: object

Servo interface - position and pulse\_width

Pulse width controlled servo.

#### **Parameters**

- fpga (bstem.Fpga) FPGA object for SPI communication
- device\_select (unsigned byte) id of device on the FPGA
- pin\_number (unsigned byte) pin number of the servo port
- pwm\_frequency\_hz (integer) frequency of pwm signal (default = 50 Hz)
- pulse\_width\_min\_us (float) minimum size of the pulse width in microsecond (default = 600 us)
- pulse\_width\_max\_us (*float*) maximum size of the pulse width in microsecond (default = 2400 us)

#### position

Return the current position value that is used by the servo pwm counter. The value - [0, 1] when PWM pulse is being generated. The value < 0, when PWM pulse is disabled.

#### pulse\_width

Get servo servo pulse-width in microsecond

#### pwm\_frequency

Return the current PWM frequency in hertz

# set\_position\_minmax(min\_val, max\_val)

Set the min/max value of the position parameter. Change is servo position is evaluated to ensure that the position stays within the required range specified here. If the user wants the servo position to be in the range 0.3 tp 0.7, then call,

servo.set\_position\_minmax(0.3, 0.7)

- min\_val (float) position minimum value
- max\_val (*float*) position maximum value

class bstem.device.SmartServo (fpga, device\_select, abs\_encoder, quad\_encoder)

Bases: object

Servo interface for smart servos with absolute position and with standard quad encoder position and velocity Smart servo.

#### **Parameters**

- fpga (bstem.Fpga) FPGA object for SPI communication
- device\_select (unsigned byte) id of device on the FPGA

abs\_gain

abs\_position

#### position

Encoder position in degrees. One full rotation increments/decrements with 2 x pi.

**Returns** position in radians

Return type float

#### position clicks

Encoder position in integer number of half-degrees. One full rotation takes 720 ticks.

**Returns** position in clicks

Return type int

speed

## velocity

Encoder velocity.

Returns velocity in rads/s

Return type float

# 1.9 bstem.disparity\_hardware module

Bases: object

Interface for computing disparity using hardware acceleration

- width (int) width of the images to be passed in must be a multiple of 8
- height (int) height of the images to be passed in must be a multiple of 8
- **image\_change\_threshold** (*int, default 2*) how much in pixel values a region (8x8 pixels) of an image must change by to indicate significant change, and thus allow disparity to be computed there.

- **speed\_up** (*boolean*, *default False*) configure the hardware to run up to 50% faster but with the trade off of not being able to run multiple instances of disparity
- merge\_left\_right\_estimates (boolean, default False) Combine the standard disparity estimate (right to left) with an approximate estimate of disparity (left to right) to improve signal to noise. This approximation is only valid when the camera motion is much smaller than the disparity offset.
- minimum\_disparity (int, default 0) Any disparity value less than minimum\_disparity will be set to zero. This is most relevant when merge left right estimates is True.

```
add_images (left_im, right_im)
```

add an image to have motion computed on

#### **Parameters**

- left\_im (ImageWrapper) left input image
- right\_im (ImageWrapper) right input image

The disparity result can be retrieved by calling get\_disparity() and will be available after approximately 2 calls to add images().

```
static aligned crop (left im, right im, crop x, crop y=0)
```

crop (equivelantly translate) left and right images relative to one another so that they are aligned (rectified) to one another at infinity or to exploit the full dynamic range of the hardware.

#### **Parameters**

- left im (ImageWrapper) left input image
- right\_im (ImageWrapper) right input image
- **crop\_x** (*int*) how much to remove in pixels from the left side of the left image, if negative then it applies to the left side of the right image instead.
- **crop\_y** (*int*, *default* 0) how much to remove in pixels from the top of the right image, if negative then it applies to the top of the left image instead.

Returns left\_im, right\_im

The cropped images will be constructed to have a width a multiple of 32 and height a multiple of 16 as required by the hardware.

```
get_disparity()
```

**Returns disparity\_map, frame\_num, left\_im, and right\_im** or None,None,None,None if data isn't available yet

Disparity is aligned to the left image, so as to match OpenCV's convention.

```
start()
```

must be called before calling add\_images or get\_disparity

stop()

should be called when done processing for now, but may start up again

# 1.10 bstem.display thread module

Helper class to display images outside the compute thread

#### atexit handler()

helper method to handle python shutdown with extra threads correctly

#### put\_window\_queue (name, img)

puts an image into the queue to be eventually displayed name – the window name used by cv2.imshow img – the image to be shown, must be an ndarray or ImageWrapper

```
waitKey_thread()
```

# 1.11 bstem.exception module

```
exception bstem.exception.DeviceNotEnabled(value)
```

Bases: exceptions. Exception

Raised when attempting communication with a device that is not enabled

```
exception bstem.exception.DeviceNotFound(value)
```

Bases: exceptions. Exception

Raised when unable to communicate with driver for the requested device

# 1.12 bstem.fpga module

```
class bstem.fpga.Fpga (spi=1)
```

Bases: object

Serial Peripheral Interface (SPI) FPGA communication

 $SPI_1 = 1$ 

 $SPI_10 = 10$ 

program (filename)

Write image file (.bin) to the FPGA.

**Parameters** filename (string) – path to image file

spi\_send (device, command, val)

Write command to FPGA SPI bus and receive response.

#### **Parameters**

- device (unsigned byte) device id
- command (unsigned byte) the command
- val (int) value to written to the device

**Returns** response from FPGA

Return type int

#### spi\_send\_mult (commands)

Write multiple commands to the FPGA SPI bus and receive responses.

- device (unsigned byte) device id
- **command** (*list[list[unsigned byte, unsigned byte, int], ...]*) list of lists containing (device, command, val)

```
Returns tuple of responses from the FPGA Return type tuple(int, ...)
```

# 1.13 bstem.hokuyo module

```
To use this code, you need to setup a static ip address for the lidar.
```

Add the following:

```
iface eth8 inet static address 192.168.0.2 gateway 192.168.0.1 netmask 255.255.255.0
```

to /etc/network/interfaces.

When you boot, run the following command:

```
sudo ifup eth8
```

```
class bstem. hokuyo. HokuyoDriver (ip='192.168.0.10', port=10940)
Hokuyo 10LX Lidar interface
```

param ip: ip address of the sensor (default: 192.167.0.100) param port: port of the sensor (default: 10940)

#### close()

Close socket to sensor

```
distance (start_step=0, end_step=1080)
```

Return a single scan of distance data in mm. A complete scan has range [0,1080] steps. Use parameters to select a sub-region of the scan space.

param start\_step: starting step of scan param start\_step: end step of scan

returns a numpy array of int, one element per distance sample in mm. If read fails, returns an empty array.

```
sensor_info()
```

Return sensor info

#### sensor\_params()

Return sensor param info

# start()

Start capturing data on the sensor

#### stop()

Stop capturing data on the sensor

#### version()

Return version info

bstem.hokuyo.main()

# 1.14 bstem.image\_wrapper module

```
class bstem.image_wrapper.ImageWrapper
    Bases: numpy.ndarray
```

ImageWrapper provides an interface to RGB, BGR and YUV (NV12) data in order to record type, cache and provide the same functionality (crop, resize, convert, etc) for all 3 image types. Further, it subclasses numpy's ndarray, which allows it to be treated just like and ndarray, for whichever image format it was created for. This means if an ImageWrapper is created with BGR data, the ImageWrapper instance can be passed to openCV

methods directly. The shape of the ImageWrapper instance will always be (height,width,3) with the exception for YUV data which will always have size height\*3/2\*width but may take on multiple different shapes for performance reasons.

Beyond caching, the ImageWrapper class also tries to minimize memory copying and maximizing lazy evaluation. As a result, one can crop and downscale without performing any memory copying, which has a significant performance benefit.

#### BGR

Returns an ImageWrapper of format BGR from the current data, converted if needed. The shape will be (height,width,3) and can be treated like any ndarray.

#### RGB

Returns an ImageWrapper of format RGB from the current data, converted if needed. The shape will be (height,width,3) and can be treated like any ndarray.

#### UV

Returns the U and V channel as an (height/2,width/2,2) ndarray, converts if needed. data[:,:,0] is the U channel and data[:,:,1] is the V.

Y

Returns the Y channel as an (height, width) ndarray, converts if needed.

#### YUV

Returns an ImageWrapper of format YUV (NV12) from the current data, converted if needed. The shape will be (height\*3/2,width) and can be treated like any ndarray:

```
Y = data[0:height,:]
UV = data[height::,:].reshape((height/2,width/2,2))
```

where UV[:,:,0] is U and UV[:,:,1] is V.

#### crop(x, y, width, height)

crop the image :param x: horizontal start position of crop :type x: int, 0 to image width :param y: vertical start position of crop :type y: int, 0 to image height :param width: width of crop, i.e. x:x+width is the range for crop :type width: int, 0 to image width :param height: height of crop, i.e. y:y+width is the range for crop :type height: int, 0 to image height

Returns a new ImageWrapper of the same format cropped.

#### downscale (downscale\_x, downscale\_y)

Computationally fast downscaling:

```
:param downscale_x: downscale size in x (width)
:type downscale_x: int
:param downscale_y: downscale size in y (height)
:type downscale_y: int
```

For YUV width/2 % downscale\_x must be 0 and height/2 % downscale\_y must be 0

For RGB and BGR width % downscale\_x must be 0 and height % downscale\_y must be 0

Returns an new ImageWrapper of the same format but downscaled.

#### has\_YUV\_data()

Test to see if YUV data can be read without converting first.

## height

```
resize (width, height, interpolation=1L)
```

resize image, can be slow :param width: the desired image width :type width: int :param height: the desired

image height :type height: int :param interpolation: interpolation method, see cv2.INTER\_\* constants :type interpolation: int, defaults to cv2.INTER\_LINEAR

Returns a new ImageWrapper of the same format resized.

width

# 1.15 bstem.infrared module

Bases: bstem.analog\_filter.AnalogFilter

Interface to Sharp 2D120X (4-30 cm) IR sensor.

mapping\_function implements linear interpolation between distance-voltage measurements from Sharp data sheet: http://www.sharpsma.com/webfm\_send/1205

Points extracted using: http://arohatgi.info/WebPlotDigitizer

Linear interpolation outperformed inverse and exponential fits.

See SharpInfrared class for further documentation

Bases: bstem.analog\_filter.AnalogFilter

Interface to Sharp 2D120X (10-80 cm) IR sensor.

mapping\_function implements linear interpolation between distance-voltage measurements from Sharp data sheet: http://www.sharpsma.com/webfm\_send/1208

Linear interpolation outperformed inverse and exponential fits.

Points extracted using: http://arohatgi.info/WebPlotDigitizer

See SharpInfrared class for further documentation

# 1.16 bstem.log module

```
class bstem.log.CsvOutput
```

Inherit this class in order to use CsvOutput.writeToCsv

```
class bstem.log.FileLogger (filename, func, names, freq)
```

Bases: bstem.log.LoggerBase

Logging variables to a CSV text file. Values are written continuously during execution of the scheduler.

Initialize logger.

- **filename** (*string*) the file
- **func** (*function returning tuple of values*) function returning a tuple of values each time it is called. Each member in the tuple will be logged as a separate column in the CSV file.
- names (tupe of strings) A tuple of names corresponding entries returned by func
- **freq** (*int*) logging frequency in Hz

```
close()
           Close file and remove logger from the scheduler.
     loop()
           The control loop
class bstem.log.LogServer (port=1234)
     Bases: bstem.log.CsvOutput, collections.defaultdict
     Server for variable logging.
     Initialize logger.
           Parameters freq (int (default 1234)) – port to listen for incoming connections
     clear()
           Remove all items from the dictionary.
     close()
           Set the server to stop listening for data and terminate.
           Return a shallow copy of the dictionary.
     default_factory
           Return default_factory of the dictionary
     get (key, default=None)
           Return the value for key if key is in the dictionary, else default. If *default is not given, it defaults to
               "None", so that this method never raises a "KeyError".
     items()
           Return a copy of the dictionary's list of "(key, value)" pairs.
     iteritems()
           Return an iterator over the dictionary's "(key, value)" pairs.
     iterkeys()
           Return an iterator over the dictionary's keys.
     itervalues()
           Return an iterator over the dictionary's values.
     keys()
           Return a copy of the dictionary's list of keys.
     pop (key, *default)
           If key is in the dictionary, remove it and returns its value, else return default. If default is not given
               and *key is not in the dictionary, a "KeyError" is raised.
     popitem()
           Remove and return an arbitrary "(key, value)" pair from the dictionary.
     read (fields, default=0)
           Returns tuple most recent _Listener readings
               Parameters
                    • fields (string or string tuple/list) – string/tuple/list of keys to query
                    • default (object) – object to return in event of a no-read
           example:
```

```
ls.read_fields(('x_pos', 'y_pos')) returns most recent readings of 'x_pos' and 'y_pos'
```

If there are no readings present for a given label, the respective tuple entry will be logged as 'default'

```
reset_subscriptions()
```

Resets the list of observers.

```
setdefault (key, default=[])
```

If key is in the dictionary, return its value. If not, insert key with a value of default and return default.

```
subscribe (client, fields)
```

**Registers an observer object with this LogServer. Any time a key is** updated, LogServer will call update() on all objects observing that key.

#### **Parameters**

- client (object with no-argument update() method) observer object
- fields (string tuple/list or string) keys to watch

#### unsubscribe (client)

Removes an observer object subscription with this LogServer.

**Parameters** client (Object with no-argument update() method) – Observer object

```
update (*E, **F)
```

Updates the values of the dictionary ("D") based on the following:

```
If ''E'' is present and has a ''.keys()'' method, then it does ''for k in E: D[k] = E[k]''. If ''E'' lacks a ''.keys()'' method, then it does ''for (k, v) in E: D[k] = v''. All cases are followed by ''for k in F: D[k] = F[k]''.
```

#### values()

Return a copy of the dictionary's list of values.

#### viewitems()

Return a new view of the dictionary's items (''(key, value)" pairs).

# viewkeys()

Return a new view of the dictionary's keys.

## viewvalues()

Return a new view of the dictionary's values.

```
writeToCsv (filename=None, precision=12)
```

Write all stored values to a CSV file.

# **Parameters**

- **filename** (*string*) the file to log to (**will overwrite**)
- **precision** (*int*) number of decimal places to output (float/complex)

```
class bstem.log.Logger (func, names, freq)
```

```
Bases: bstem.log.CsvOutput, bstem.log.LoggerBase
```

Variable logging to text file or remote server. Log data is stored in memory during logging.

Initialize logger.

- **func** (*function returning tuple of values*) function returning a tuple of values each time it is called. Each member in the tuple will be logged as a separate column in the CSV file.
- **freq** (*int*) logging frequency in Hz

### clear()

Clear all stored data.

### close()

Remove the object from the Scheduler.

### enabled

Logging enabled. When true data recorded whenever the Scheduler is running.

**Returns** true if logging enabled

Return type bool

### loop()

The control loop

writeToCsv (filename=None, precision=12)

Write all stored values to a CSV file.

#### **Parameters**

- filename (string) the file
- **precision** (*int*) number of decimal places to output (float/complex)

### writeToServer (ip, port=1234)

Write all stored values to a remote log server.

### **Parameters**

- ip (string) IP address of the remote server
- port (int (default 1234)) port of remote server

```
class bstem.log.LoggerBase (func, names, freq)
```

Bases: bstem.control.ControlLoop

Abstract base class for all loggers.

Initialize logger.

### **Parameters**

- **func** (*function returning tuple of values*) function returning a tuple of values each time it is called. Each member in the tuple will be logged as a separate column in the CSV file.
- **freq** (*int*) logging frequency in Hz

close()

Remove logger from the scheduler.

class bstem.log.RemoteLogger(ip, func, names, freq, port=1234)

Bases: bstem.log.LoggerBase

Initialize logger.

### **Parameters**

- ip (string) IP address of the remote server
- **func** (*function returning tuple of values*) function returning a tuple of values each time it is called. Each member in the tuple will be logged as a separate column in the CSV file.

```
• names (tuple(string)) – A tuple of names corresponding to each entry returned by func
                 • freq (int) – logging frequency in Hz
                 • port (int (default 1234)) – port of remote server
     close()
           Close connection to remote server and remove logger from the scheduler.
           The control loop
class bstem.log.VariableClient (ip, port=1235)
     Bases: object
     Client for setting variables remotely.
     Initialize client.
           Parameters
                 • ip (string) – ip address or host name of variable server
                 • port (int (default 1235)) – remote port
     close()
           Close the associated socket.
     set (name, val)
           Set a remote variable.
               Parameters
                    • name (string) – name of the variable as set by VariableServer.register
                    • val - the new value
class bstem.log.VariableServer (freq=2, port=1235)
     Bases: bstem.control.ControlLoop
     Server for setting variables remotely.
     Initialize server.
           Parameters
                 • freq (int (default 2 Hz)) – frequency in Hz
                 • port (int (default 1235)) – port to listen for incoming connections
     close()
           Remove logger from the scheduler.
     loop()
     register(obj, name)
           Register an object with the server.
               Parameters
                    • obj (a python object) – the object
                    • name (string) – object name as it will be referenced on the client
```

# 1.17 bstem.motion\_hardware module

Bases: bstem.video\_encoder.VideoEncoder

Motion specific interface to the Video Encoder

#### **Parameters**

- width (int) width of the images to be passed in must be a multiple of 8
- height (int) height of the images to be passed in must be a multiple of 8
- **bitrate** (*int*) the bitrate used for encoding
- max\_im\_queued (int) the size of the image queue, used by add\_image
- max\_frames\_queued (int) the size of the frame queue, used by get\_motion
- **image\_change\_threshold** (*int*, *default* 5) how much in pixels a region of an image must change by to indicate significant change, and thus allow motion to be computed there.

add\_image (im, block=True, timeout=10)

add an image to have motion computed on

### **Parameters**

- im (ImageWrapper) input image
- **block** (*int*, *default 10*) flag to indicate waiting until there is space in the input queue. If False, images will be dropped when the input queue is full.
- timeout If block==True then timeout is how long to wait while blocking

```
get_motion (block=False, timeout=1, fillin_val=0)
```

Gets the computed motion. The motion is normally available after 4-6 add\_image() calls. Valid motion estimates are between 7 and 248, which correspond to half pixel resolution motion estimates from the previous frame to the current frame. Regions where motion could not be computed will have an "invalid" 0 value.

## **Parameters**

- **block** (*int*, *default 1*) flag to indicate waiting until there is a valid motion measurement.
- timeout If block==True then timeout is how long to wait while blocking

Returns dx, dy, frame\_number, frame, prev\_frame or None,None,None,None,None if no motion is available

dx is the motion in x

dy is the motion in y

frame\_number is the number of the frame being returned, starts at 0 and may not be contiguous because data can get dropped

frame is the reference frame for which motion is computed (and aligned to)

prev\_frame is the frame used to compute motions from, motion vectors are

# start()

must be called before calling add\_image or get\_motion

```
bstem.motion_hardware.video_encoder_speedup_hack()
Initialize the hardware in a "faster" configuration.
```

Run this command before using the hardware for motion processing.

This is a complete hack and has no justification for the parameters except that they work.

The video encoder hardward supports up to 4 simultaneous streams to be compressed. By creating several with different sizes the hardware goes into a faster mode. Use this hack until the kernel driver can be manipulated so that we can always get fast encoding.

# 1.18 bstem.plot module

```
class bstem.plot.Plot
    Bases: object
    Live variable plotting.
    classmethod clear(*args, **kwargs)
    classmethod histogram(*args, **kwargs)
    Live histogram.
```

#### **Parameters**

- **func** (function returning single values OR two-element tuple, (LogServer instance, string OR tuple/list of strings)) function returning a single value each time it is called. Each member in the tuple will be plotted as a separate line OR a tuple with two elements, a LogServer and a string or tuple of strings representing keys to observe.
- title (string) plot title (Default ")
- **x\_label** (*string*) x axis label (Default ")
- **bg\_color** (*string*) plot background color (Default 'k')
- **color** ([int,]) histogram color in [R,G,B,Alpha] format, where each entry ranges from 0-255 (Default [0, 0, 255, 80])
- bins (int) number of bins (Default 20)
- bin\_start (int) min start value, if None will be minimum of recorded values (Default None)
- bin\_end (int) min end value, if None will be maximum of recorded values (Default None)
- windowsize ([int, int]) window size (Default (600, 350))
- hist\_len length of history, if None all values are stored (Default None)
- **is\_observer** (*Boolean*) True if plot's update will be called remotely (e.g. if the plot will registering as a subscriber to a LogServer)

**NOTE:** When argument 'func' is a tuple (instead of a function), the plot will not update itself periodically (the default behavior). Instead, the tuple's first element is interpreted as a LogServer to observe and its second element is interpreted as a string or tuple/list of strings representing which keys on the LogServer should be observed. When a plot's observed key is updated the LogServer will make the necessary calls to update().

Histogram Example:

```
from bstem.plot import Plot
from time import time
from math import sin

def fxn():
    # Line Plot can plot multiple fields
    return sin(time() % 6.2830)

# Add histograms to Scheduler
Plot.histogram(fxn, 'Local Histogram')
Plot.histogram(fxn, bins=40, bin_start=-0.5, bin_end=1.5, windowsize=(800, 700))
Plot.histogram(fxn, 'title', 'x_label', 'w', [255, 0, 0, 50])

# Run Scheduler for 5 seconds
Scheduler.start(5)

# Remove histograms from Scheduler
Plot.clear()
```

### classmethod lineplot (\*args, \*\*kwargs)

Live line plot.

#### **Parameters**

- **func** (function returning single values OR two-element tuple, (LogServer instance, string OR tuple/list of strings)) function returning a single value each time it is called. Each member in the tuple will be plotted as a separate line OR a tuple with two elements, a LogServer and a string or tuple of strings representing keys to observe.
- title (string) plot title (Default ")
- **y\_label** (*string*) y axis label (Default ")
- **y\_units** (*string*) y unit label (Default None)
- **bg\_color** (*string*) plot background color (Default 'k')
- color map ([string, ]) colors for each line (Default ['r', 'g', 'b', 'c', 'm', 'k'])
- linewidth (*float*) line width (Default 2.0)
- timewindow (int) time window of plotted data in seconds (Default 10)
- windowsize ([int, int]) window size (Default (600, 350))

**NOTE:** When argument 'func' is a tuple (instead of a function), the plot will not update itself periodically (the default behavior). Instead, the tuple's first element is interpreted as a LogServer to observe and its second element is interpreted as a string or tuple/list of strings representing which keys on the LogServer should be observed. When a plot's observed key is updated the LogServer will make the necessary calls to update().

## LinePlot Example:

```
from bstem.plot import Plot
from time import time
from math import sin

def fxn():
    # Line Plot can plot multiple fields
    return (sin(time() % 6.2830) + .3, sin(time() % 6.2830), sin(time() % 6.2830) - .3)
```

```
# Add lineplots to Scheduler
Plot.lineplot(fxn, 'Local Line Plot')
Plot.lineplot(fxn, timewindow=20, windowsize=(800, 700))
Plot.lineplot(fxn, 'title', 'y_label', 'y_units', 'w', ['b', 'g', 'r'], 3.0)

# Run Scheduler for 5 seconds
Scheduler.start(5)

# Remove lineplots from Scheduler
Plot.clear()
```

# 1.19 bstem.sensor module

```
class bstem.sensor.Accelerometer
     Bases: bstem.sensor.Sensor
     Linear 3D Accelerometer.
     enabled
           Device enabled.
               Returns True if device is enabled
               Return type Boolean
     value
           Linear acceleration.
               Returns last recorded linear acceleration along all axis (x, y, z) in g's
               Return type tuple (long, long, long)
     value_ts(ts)
           Linear acceleration.
               Returns last recorded linear acceleration along all axis (x, y, z) in g's
               Return type tuple (long, long, long)
     x
           Acceleration along the x axis.
               Returns last recorded linear acceleration along x axis in g's
               Return type long
     У
           Acceleration along the y axis.
               Returns last recorded linear acceleration along y axis in g's
               Return type long
     z
           Acceleration along the z axis.
               Returns last recorded linear acceleration along z axis in g's
               Return type long
```

```
class bstem.sensor.Barometer
     Bases: bstem.sensor.Sensor
     Absolute pressure barometer.
     enabled
          Device enabled.
               Returns True if device is enabled
               Return type Boolean
     pressure
          Absolute pressure.
               Returns last recorded absolute pressure
               Return type long
     value
          Absolute pressure.
               Returns last recorded absolute pressure
               Return type tuple (long)
class bstem.sensor.Gyroscope
     Bases: bstem.sensor.Sensor
     3 axis Gyroscope.
     DPS_2000 = 2000
     DPS_250 = 250
     \mathtt{DPS}\_500 = 500
     dps
          Degrees per Second.
               Returns gyro sensitivity (250/500/2000 dps)
               Return type Boolean
     enabled
          Device enabled.
               Returns True if device is enabled
               Return type Boolean
     value
          Angular rate.
               Returns last recorded angular rate (x, y, z) in rads/s
               Return type tuple (long, long, long)
     value_ts(ts)
          Angular rate.
               Returns last recorded angular rate (x, y, z) in rads/s
               Return type tuple (long, long, long)
     x
          Angular rate along the x axis.
```

```
Returns last recorded angular rate along x axis in rads/s
               Return type long
     У
           Angular rate along the y axis.
               Returns last recorded angular rate along y axis in rads/s
               Return type long
     Z
           Angular rate along the z axis.
               Returns last recorded angular rate along z axis in rads/s
               Return type long
class bstem.sensor.Magnetometer(calibrated=False)
     Bases: bstem.sensor.Sensor
     3D magnetic sensor :param calibrated: returns calibrated readings when True, else returns raw values from
     sensor. :type calibrated: Boolean
     For calibrating magnetometer you can use the script compass_calibration.py in bstem. Calibrated magnetometer
     data is stored in /usr/share/bstem/.compass_calibration.log
     calibration_file_name = '/usr/share/bstem/.compass_calibration.log'
     enabled
           Device enabled.
               Returns True if device is enabled
               Return type Boolean
     raw value
           Orientation.
               Returns last recorded orientation (x, y, z)
               Return type tuple (long, long, long)
     value
           Orientation.
               Returns last recorded orientation (x, y, z)
               Return type tuple (long, long, long)
     x
           Orientation along the x axis.
               Returns last recorded orientation along x axis
               Return type long
     У
           Orientation along the y axis.
               Returns last recorded orientation along y axis
               Return type long
     z
           Orientation along the z axis.
```

**Returns** last recorded orientation along z axis

### Return type long

```
class bstem.sensor.Sensor
```

Bases: object

Abstract base class for all sensors.

timestamp(name)

value()

Return tuple of parameter values for this sensor.

## 1.20 bstem.sonar module

MaxSonar code running on better using the analog input for sampling the range data. PWM mode is not supported currently. The sonar need to be connected to one of the better analog pins.

Filter types: MEDIAN, MODE, NONE (last sample only)

### Tested on:

1. http://www.maxbotix.com/Ultrasonic\_Sensors/MB1200.htm

Currently supported sonar class is XL. Will be extended to other types shortly.

Bases: object

MaxSonarArray triggering mechanism using GPIO pins of Bstem. Only supports all sonars of same type and same supply voltage.

TODO: Allow sonar class as an array of different type and different voltage

specify the pins to which sonar analog signal is connected. It is recommended to connect the gpio[4] pin to the sonar RX pin to start the convertion process.

### distance

returns a list of filtered distance for the sonar array

### get\_last\_sample()

Returns a list of last sample from the sonar array

```
get_samples()
```

returns all the recent samples collected from the sonar array. Each element of the list is a buffer or history of current reading. Useful for user specific custom functions.

### raw\_value

Samples and returns the raw value from sonar adc without doing any cms conversion

# 1.21 bstem.tracker module

```
class bstem.tracker.CAMShiftTracker (initial_bb, img_curr, histSize=[16, 16])
```

Tracker Based on Color Histogram and CAMShift. Initialize the tracker by specifying a bounding box on the initial image. This will be the target.

The tracker uses the HSV colorspace and a 2D histogram is constructed for the target. One needs to specify the number of bins in the two axes of this histogram.

Initialize a tracker. :param initial\_bb: a list [x,y,w,h] where x and y is the coordinate of the upper-left corner of the bounding box, w and h is the width and height. :type initial\_bb: list :param img\_curr: the initial image in BGR :type img\_curr: Numpy array :param histSize: the number of bins in the histogram :type histSize: list, default to [16,16]

Parameters img\_curr (Numpy array) - new image in BGR

# 1.22 bstem.video module

This class provides streaming video from the bettern to a web page via the Html5 mjpeg video tag (currently supported in Chrome and Firefox). Initialize the video object with a callback function that generates a single frame each time it is called and returns it as a numpy array e.g. to send a video stream from the right camera:

See the video\_basic example for complete python and html to stream a single camera.

Initialize streaming video.

### **Parameters**

- callback (numpy array) callback to retrieve each video frame
- width (int (default 640)) width of video frame
- height (int (default 480)) height of video frame

### next()

Return next encoded video frome.

```
setup()
```

Start streaming video

### shutdown()

Stop streaming video

#### streaming

Return true if currently generating and encoding frames.

# 1.23 bstem.video\_encoder module

```
 \begin{array}{c} \textbf{class} \, \texttt{bstem.video\_encoder.VideoEncoder} \, (\textit{width=}1280, \quad \textit{height=}720, \quad \textit{bitrate=}3000000, \\ \textit{max\_im\_queued=}1, \quad \textit{max\_frames\_queued=}1, \quad \textit{cap-ture\_type='COMPRESSED'}, \textit{fps=}1, \textit{grayscale=}False) \\ \textbf{Bases: object} \end{array}
```

This class provides convenient access to the Video Encoder. It is able to take aribrary images and return compressed, uncompressed or motion data. It is currently most used for motion information but could be used for streaming compressed data.

#### **Parameters**

- width (int) width of the images to be passed in must be a multiple of 16
- height (int) height of the images to be passed in must be a multiple of 16
- bitrate (int) the bitrate used for encoding
- max\_im\_queued (int) the size of the image queue, used by add\_image
- max\_frames\_queued (int) the size of the frame queue, used by get\_frame
- capture\_type if "FILE" the compressed data will be written to an .mp4 file.

Otherwise, this indicates the format of data that will be available when calling get\_frame() :type capture\_type: string: "FILE", "COMPRESSED", "DECOMPRESSED" OR "MOTION"

```
add_image (im, block=True, timeout=10)
```

im is the image to be encoded. It must be an instance of ImageWrapper block indicates if add\_image should wait until there is space in the image queue timeout indicates how long to wait if block is True, in seconds and 0 means indefinitely

### atexit()

helper method to handle python shutdown with extra threads correctly

```
get_frame (block=False, timeout=1)
```

Gets a frame. block indicates if get\_frame should wait until there is a frame in the frame queue timeout indicates how long to wait if block is True, in seconds and 0 means indefinitely

returns frame as a numpy array, timestamp or None, None if no frame is available

```
set_filename (filename)
```

#### start()

must be called before calling add\_image or get\_frame

### stop()

should be called when done processing for now, but may start up again

# 1.24 Module contents

# PYTHON MODULE INDEX

b	9
bstem,43	bstem.tutorials.example_3c_camera_config,
bstem.analog_filter, 12	9
bstem.battery_check, 12	bstem.tutorials.example_4_scheduler,9
bstem.camera, 13	bstem.tutorials.example_5a_local_logging,
bstem.control, 15	9
bstem.decorator, 16	<pre>bstem.tutorials.example_5b_remote_logging_client_s</pre>
bstem.device, 17	10
bstem.disparity_hardware, 25	<pre>bstem.tutorials.example_6a_local_plotting,</pre>
bstem.display_thread, 26	10
bstem.exception, 27	<pre>bstem.tutorials.example_6b_remote_plot_client_serv</pre>
bstem.fpga, 27	10
bstem.hokuyo, 28	<pre>bstem.tutorials.example_7_parameter_tuning,</pre>
bstem.image_wrapper, 28	10
bstem.infrared, 30	bstem.tutorials.logging_client, 10
bstem.log, 30	bstem.tutorials.logging_server,10
bstem.motion_hardware, 35	bstem.tutorials.remoteplot_client,10
bstem.platform, 5	bstem.tutorials.remoteplot_server,11
bstem.platform.ad_cord,1	bstem.utils, 12
bstem.platform.bstem_platform,2	bstem.utils.analyze_sonar_dataset,11
bstem.platform.lg_cord,3	bstem.utils.compass_calibration, 11
bstem.platform.rc_cord,4	bstem.utils.demo_ppm_out,11
bstem.plot, 36	bstem.utils.get_min_max_ppm,11
bstem.sensor,38	bstem.utils.make_sonar_dataset, 12
bstem.sonar,41	bstem.utils.ppm_repeater, 12
bstem.test,7	bstem.video,42
bstem.test.cord_info,5	bstem.video_encoder,43
$bstem.test.test\_bstem, 6$	
bstem.test.test_control,6	
bstem.test.test_image_wrapper,6	
bstem.test.test_plot,6	
bstem.test.test_tracker,7	
bstem.tracker,41	
bstem.tutorials, 11	
bstem.tutorials.demo_sonar_ir,7	
bstem.tutorials.example_10_tracking,8	
$\verb bstem.tutorials.example_11_disparity, 8 \\$	
<pre>bstem.tutorials.example_1_hello_world,</pre>	
8	
$\verb bstem.tutorials.example_2_sensors, 8 \\$	
<pre>bstem.tutorials.example_3a_camera,8</pre>	
bstem.tutorials.example_3b_video_record	ling,

46 Python Module Index

A avg_exec_time (bstem.control.ControlLoop attribute),	, 15
abs_gain (bstem.device.SmartServo attribute), 25	
abs_position (ostem.device.SmartServo attribute), 23	
AbsEncoder (class in bstem.device), 17 Barometer (class in bstem.sensor), 38	
Acceler (class in bstem.tutorials.example_4_scheduler), 9 battery (bstem.platform.ad_cord.AdCord attribute), 2	
Accelerometer (class in bstem.sensor), 38 battery (bstem.platform.lg_cord.LgCord attribute), 4	
ADC_DEV_BATTERY (bstem.device.AdcAdCord at- battery (bstem.platform.rc_cord.RcCord attribute), 5	
tribute), 17 BGR (bstem.image_wrapper.ImageWrapper attribute),	
ADC_DEV_BATTERY (bstem.device.AdcRcCord attribute), 18  Blinker (class in bstem.tutorials.example_4_scheduler brightness (bstem.camera.Camera attribute), 13	:), 9
ADC_DEV_MOTOR_0 (bstem.device.AdcAdCord at- brightness (bstem.device.Led attribute), 20	
tribute), 17 Bstem (class in bstem.platform.bstem_platform), 2	
ADC_DEV_MOTOR_0 (bstem.device.AdcLgCord at- bstem (module), 43	
tribute), 18 bstem.analog_filter (module), 12	
ADC_DEV_MOTOR_1 (bstem.device.AdcAdCord at- bstem.battery_check (module), 12	
tribute), 17 bstem.camera (module), 13	
ADC_DEV_MOTOR_2 (bstem.device.AdcAdCord at bstem.control (module), 15	
tribute), 17 bstem.decorator (module), 16	
ADC_DEV_MOTOR_3 (bstem.device.AdcAdCord at bstem.device (module), 17	
tribute), 17 bstem.disparity_hardware (module), 25	
ADC_DS (bstem.platform.rc_cord.RcCord attribute), 5 bstem.display_thread (module), 26	
AdcAdCord (class in bstem.device), 17 bstem.exception (module), 27	
AdcLgCord (class in bstem.device), 17 bstem.fpga (module), 27	
AdCord (class in bstem.platform.ad_cord), 1 bstem.hokuyo (module), 28	
AdcRcCord (class in bstem.device), 18 bstem.image_wrapper (module), 28	
add() (bstem.control.Scheduler class method), 16 bstem.infrared (module), 30	
add_image() (bstem.motion_hardware.MotionHardware bstem.log (module), 30	
method), 35 bstem.motion_hardware (module), 35	
add_image() (bstem.video_encoder.VideoEncoder bstem.platform (module), 5	
method), 43 bstem.platform.ad_cord (module), 1	
add_images() (bstem.disparity_hardware.DisparityHardwarebstem.platform.bstem_platform (module), 2	
method), 26 bstem.platform.lg_cord (module), 3	
aligned_crop() (bstem.disparity_hardware.DisparityHardwarestem.platform.rc_cord (module), 4	
static method), 26 bstem.plot (module), 36	
AnalogFilter (class in bstem.analog_filter), 12 bstem.sensor (module), 38 atexit() (bstem.camera.Camera method), 13 bstem.sonar (module), 41	
atexit() (bstem.camera.Camera method), 13 bstem.sonar (module), 41 atexit() (bstem.video_encoder.VideoEncoder method), 43 bstem.test (module), 7	
atexit() (bstein.video_encoder.videoEncoder inethod), 43 bstein.test (inoduce), 7 atexit_handler() (bstein.video_encoder.videoEncoder inethod), 43 bstein.test (inoduce), 5	
method), 26 bstem.test.test_bstem (module), 6	
auto_exposure (bstem.camera.Camera attribute), 13  bstem.test_est_est_(module), 6  bstem.test_test_control (module), 6	
auto_exposure (ostern.camera.camera attribute), 13 bstem.test.test_image_wrapper (module), 6	
avg_error (bstem.control.ControlLoop attribute), 15 bstem.test.test_plot (module), 6	

bstem.test.test_tracker (module), 7 bstem.tracker (module), 41	clear() (bstem.log.Logger method), 33 clear() (bstem.log.LogServer method), 31
bstem.tutorials (module), 11	clear() (bstem.plot.Plot class method), 36
bstem.tutorials.demo_sonar_ir (module), 7	close() (bstem.hokuyo.HokuyoDriver method), 28
bstem.tutorials.example_10_tracking (module), 8	close() (bstem.log.FileLogger method), 30
bstem.tutorials.example_11_disparity (module), 8	close() (bstem.log.Logger method), 33
bstem.tutorials.example_1_hello_world (module), 8	close() (bstem.log.LoggerBase method), 33
bstem.tutorials.example_2_sensors (module), 8	close() (bstem.log.LogServer method), 31
bstem.tutorials.example_3a_camera (module), 8	close() (bstem.log.RemoteLogger method), 34
bstem.tutorials.example_3b_video_recording (module), 9	close() (bstem.log.VariableClient method), 34
bstem.tutorials.example_3c_camera_config (module), 9	close() (bstem.log. VariableServer method), 34
bstem.tutorials.example_4_scheduler (module), 9	CLTestLoop (class in bstem.test.test_control), 6
bstem.tutorials.example_5a_local_logging (module), 9	Clunker (class in bstem.tutorials.example_4_scheduler),
bstem.tutorials.example_5b_remote_logging_client_server	
(module), 10	collect_data() (in module
bstem.tutorials.example_6a_local_plotting (module), 10	bstem.tutorials.example_5a_local_logging), 9
bstem.tutorials.example_6b_remote_plot_client_server (module), 10	collect_data() (in module bstem.tutorials.logging_client),
bstem.tutorials.example_7_parameter_tuning (module),	color (bstem.device.MulticolorLed attribute), 21
10	CompassCalibration (class in
bstem.tutorials.logging_client (module), 10	bstem.utils.compass_calibration), 11
bstem.tutorials.logging_server (module), 10	contrast (bstem.camera.Camera attribute), 13
bstem.tutorials.remoteplot_client (module), 10	control_vars_reset() (bstem.control.ControlLoop
bstem.tutorials.remoteplot_server (module), 11	method), 15
bstem.utils (module), 12	ControlLoop (class in bstem.control), 15
bstem.utils.analyze_sonar_dataset (module), 11	convert_to_clicks() (bstem.device.Encoder method), 19
bstem.utils.compass_calibration (module), 11 bstem.utils.demo_ppm_out (module), 11	convert_to_clicks() (bstem.device.QuadEncoder method), 23
bstem.utils.get_min_max_ppm (module), 11	convert_to_rads() (bstem.device.Encoder method), 19
bstem.utils.make_sonar_dataset (module), 12	convert_to_rads() (bstem.device.QuadEncoder method),
bstem.utils.ppm_repeater (module), 12	23
bstem.video (module), 42	copy() (bstem.log.LogServer method), 31
bstem.video_encoder (module), 43	crop() (bstem.image_wrapper.ImageWrapper method),
bstem_alternate_sensor_examples() (in module	29
bstem.tutorials.example_2_sensors), 8	CsvOutput (class in bstem.log), 30
bstem_sensor_examples() (in module	
bstem.tutorials.example_2_sensors), 8	D
	default_factory (bstem.log.LogServer attribute), 31
C	demo_range_sensor() (in module
calc_velocity() (bstem.device.Encoder method), 18	bstem.tutorials.demo_sonar_ir), 7
calc_velocity() (bstem.device.QuadEncoder method), 23	DeviceNotEnabled, 27
calibrate() (bstem.utils.compass_calibration.CompassCalib	
class method), 11	direction (bstem.device.Gpio attribute), 19
calibration_file_name (bstem.sensor.Magnetometer at-	disable() (bstem.device.Motor method), 20
tribute), 40	DisparityHardware (class in bstem.disparity_hardware),
Camera (class in bstem.camera), 13	25
camera_config_example() (in module	DisplayThread (class in bstem.display_thread), 26
bstem.tutorials.example_3c_camera_config), 9	distance (bstem.analog_filter.AnalogFilter attribute), 12
camera_example() (in module	distance (bstem.sonar.MaxSonarArray attribute), 41
bstem.tutorials.example_3a_camera), 8	distance() (bstem.hokuyo.HokuyoDriver method), 28
CAMShiftTracker (class in bstem.tracker), 41	do_battery_check() (in module bstem.battery_check), 12
central_auto_exposure (bstem.camera.Camera attribute),	downscale() (bstem.image_wrapper.ImageWrapper method), 29
cleanup() (bstem.camera.Camera method), 13	dps (bstem.sensor.Gyroscope attribute), 39
	-r - (

DPS_2000 (bstem.sensor.Gyroscope attribute), 39 DPS_250 (bstem.sensor.Gyroscope attribute), 39 DPS_500 (bstem.sensor.Gyroscope attribute), 39	get_min_max_ppm_values() (in module bstem.utils.get_min_max_ppm), 11 get_motion() (bstem.motion_hardware.MotionHardware
E	method), 35 get_pulse_width() (bstem.device.PPMOutput method),
enable_motors (bstem.platform.ad_cord.AdCord attribute), 2	21 get_samples() (bstem.analog_filter.AnalogFilter method),
enable_motors (bstem.platform.lg_cord.LgCord attribute), 4	12 get_samples() (bstem.sonar.MaxSonarArray method), 41
enable_pwm_out (bstem.platform.ad_cord.AdCord attribute), 2	Gpio (class in bstem.device), 19 Gyroscope (class in bstem.sensor), 39
enable_pwm_out (bstem.platform.rc_cord.RcCord attribute), 5	Н
enable_servo_power (bstem.platform.rc_cord.RcCord attribute), 5	hardware_disparity_example() (in module bstem.tutorials.example_11_disparity), 8
enabled (bstem.log.Logger attribute), 33 enabled (bstem.sensor.Accelerometer attribute), 38	has_YUV_data() (bstem.image_wrapper.ImageWrapper method), 29
enabled (bstem.sensor.Barometer attribute), 39 enabled (bstem.sensor.Gyroscope attribute), 39	height (bstem.image_wrapper.ImageWrapper attribute),
enabled (bstem.sensor.Magnetometer attribute), 40 Encoder (class in bstem.device), 18 end_recording() (bstem.camera.Camera method), 13	histogram() (bstem.plot.Plot class method), 36 hit_return() (in module bstem.utils.get_min_max_ppm), 11
error (bstem.control.ControlLoop attribute), 15 exec_time (bstem.control.ControlLoop attribute), 15	HokuyoDriver (class in bstem.hokuyo), 28 hue (bstem.camera.Camera attribute), 14
F	1
fade_to_black (bstem.camera.Camera attribute), 14 FileLogger (class in bstem.log), 30 filelogger_example() (in module bstem.tutorials.example_5a_local_logging), 9 find_file() (in module bstem.test.cord_info), 5 Fpga (class in bstem.fpga), 27 freq (bstem.control.ControlLoop attribute), 15	id (bstem.device.PowerGauge attribute), 21 ImageWrapper (class in bstem.image_wrapper), 28 items() (bstem.log.LogServer method), 31 iter (bstem.control.ControlLoop attribute), 15 iteritems() (bstem.log.LogServer method), 31 iterkeys() (bstem.log.LogServer method), 31 itervalues() (bstem.log.LogServer method), 31
G	K
gain (bstem.camera.Camera attribute), 14 gain (bstem.device.AbsEncoder attribute), 17	keys() (bstem.log.LogServer method), 31
generate_sample_video() (in module bstem.test.test_tracker), 7	L
generate_shutdown_message() (in module bstem.battery_check), 12 generate_warning_message() (in module	Led (class in bstem.device), 19 led_example() (in module bstem.tutorials.example_1_hello_world),
bstem.battery_check), 12 get() (bstem.log.LogServer method), 31	LgCord (class in bstem.platform.lg_cord), 3
get_bstem_cord() (in module bstem.test.cord_info), 5 get_cord_property() (in module bstem.test.cord_info), 5	LgServo (class in bstem.device), 20 lineplot() (bstem.plot.Plot class method), 37 Logger (class in bstem.log), 32
get_disparity() (bstem.disparity_hardware.DisparityHardw method), 26	are logger_writeToCsv_example() (in module
get_frame() (bstem.video_encoder.VideoEncoder method), 43	bstem.tutorials.example_5a_local_logging), 9 LoggerBase (class in bstem.log), 33 LogServer (class in bstem log), 31
get_last_sample() (bstem.analog_filter.AnalogFilter method), 12	LogServer (class in bstem.log), 31 loop() (bstem.control.ControlLoop method), 15 loop() (bstem.log.FileLogger method), 31
get_last_sample() (bstem.sonar.MaxSonarArray method), 41	loop() (bstem.log.FrieLogger method), 33

method, 9 loop() (bstem.tutorials.example_4_scheduler.Blinker) method), 9 loop() (bstem.tutorials.example_4_scheduler.Clunker) method), 9  M M Magnetometer (class in bstem.sensor), 40 main() (in module bstem.tutorials.example_5a_local_logging_server), 10 main() (in module bstem.tutorials.example_5a_local_logging_server), 10 main() (in module bstem.tutorials.example_5a_local_logging_elient_method), 27 pulse_width (bstem.device.Pervo attribute), 24 main() (in module bstem.tutorials.example_5a_local_logging_elient_method), 27 pulse_width (bstem.device.Pervo attribute), 24 main() (in module bstem.tutorials.example_5a_local_plotting), 10 main() (in module bstem.tutorials.example_6a_local_plotting), 10 main() (in module bstem.tutorials.example_7_parameter_turnler), 21 max_cell_voltage (bstem.device.PowerGauge attribute), 23 mone (bstem.platform.re_cord.RcCord attribute), 23 mane (bstem.platform.re_cord.RcCord attribute), 24 mane (bstem.platform.re_cord.RcCord attribute), 25 pususe() (bstem.camera.Camera method), 14 period (bstem.device.AbsEncoder attribute), 17 popticard() (bstem.device.AbsEncoder attribute), 19 popticard() (bstem.device.AbsEncoder attribute), 23 position (bstem.device.CabsEncoder attribute), 23 position (bstem.device.Encoder attribute), 24 position (bstem.device.Encoder attribute), 29 position (bstem.device.Encoder attribute), 29 position (bstem.device.Encoder attribute), 29 position (bstem.device.E	loop() (b	stem.log.RemoteLogger method), 34 stem.log.VariableServer method), 34 stem.test.test_control.CLTestLoop method), 6 (bstem.tutorials.example_4_scheduler.Acceler	position_clicks (bstem.device.QuadEncoder attribute), 23 position_clicks (bstem.device.SmartServo attribute), 25 PowerGauge (class in bstem.device), 21 PPM_DS (bstem.platform.rc_cord.RcCord attribute), 5
method), 9 (bstem.tutorials.example_4_scheduler.Clunker method), 9  Magnetometer (class in bstem.sensor), 40 main() (in module bstem.tutorials.example_5a_local_logismigle_width() (bstem.device_PMOutput method), 27 pulse_width (bstem.device_PMOutput method), 21 pulse_width (bstem.device.PMOutput pulse_width (bstem.device.PMOutput method), 21 pulse_width			
loop() (bstem.tutorials.example_4.scheduler.Clunker method), 9  Magnetometer (class in bstem.sensor), 40 main() (in module bstem.tutorials.example_5a_local_logisin_gwise_width) (bstem.device.EgServo attribute), 24 main() (in module bstem.tutorials.example_5a_local_logisin_gwise_width) (bstem.device.Re attribute), 24 main() (in module bstem.tutorials.example_5a_local_logisin_gwise_width) (bstem.device.PwPMOutput method), 21 pul_window_queue() (bstem.display_thread.DisplayThr	loop()		*
bstem.utorials.logging_server), 10  Magnetometer (class in bstem.sensor), 40 main() (in module bstem.hokuyo), 28 main() (in module bstem.hokuyo), 28 main() (in module bstem.tutorials.example_5a_local_loggin_gulse_width (bstem.device.Re attribute), 24 main() (in module bstem.tutorials.example_5a_local_loggin_gulse_width() (bstem.device.PMOutput method), 21 put_window_queue() (bstem.display_thread.DisplayThread main() (in module bstem.tutorials.example_6a_local_plotting), 10 main() (in module bstem.tutorials.example_6b_remote_plot_QuliadEncoder)(class in bstem.device.Servo attribute), 24 main() (in module bstem.tutorials.example_6b_remote_plot_QuliadEncoder)(class in bstem.device), 23 10 main() (in module bstem.tutorials.example_7_parameter_turng), 10 main() (in module bstem.tutorials.example_7_parameter_turng), 10 main() (in module bstem.tutorials.example_7_parameter_turng), 10 main() (in module bstem.sensonar), 41 MaxSonar (class in bstem.sonar), 41 MaxSonar (class in bstem.device), 20 MulticolorLed (class in bstem.device), 20 Folto (class in bstem.platform.ra_cord.AdCord attribute), 2 name (bstem.platform.ad_cord.AdCord attribute), 2 name (bstem.cample_form.ra_cord.ReCord attribute), 3 name (bstem.platform.ad_cord.AdCord attribute), 2 pause() (bstem.cample_form.ra_cord.ReCord attribute), 3 remoilog_ger_client() (in module bstem.tutorials.example_form.ra_cord.Record attribute), 3 remoilog_			
Magnetometer (class in bstem.sensor), 40 main() (in module bstem.hokuyo), 28 main() (in module bstem.tutorials.example_5a_local_logging_main() (in module bstem.tutorials.example_5b_remote_logging_client_matching(), 27	loop()		
Magnetometer (class in bstem.sensor), 40 main() (in module bstem.hokuyo), 28 main() (in module bstem.tutorials.example_5a_local_loggingylise_width() (bstem.device.PMOutput method), 21 pulse_width (bstem.device.Powro attribute), 24 main() (in module bstem.tutorials.example_5b_remote_logging_client_webed), 27 pwm_frequency (bstem.device.Powro attribute), 24 main() (in module bstem.tutorials.example_6a_local_plotting), 10 main() (in module bstem.tutorials.example_6b_remote_plot_QlitatEseotex)(class in bstem.device), 23 10 main() (in module bstem.tutorials.example_7_parameter_turneg), 10 max_cell_voltage (bstem.device.PowerGauge attribute), 21 MaxSonar (class in bstem.sonar), 41 MaxSonar (class in bstem.device), 20 MulticolorLed (class in bstem.device), 20 P Pause() (bstem.camera.Camera method), 42 read() (bstem.platform.re_cord.ReCord attribute), 5 next() (bstem.camera.Camera method), 42 read() (bstem.device.Phaplatform.re_cord, 4 read() (bstem.device), 20 RC_MULTI_DS (bstem.platform.re_cord), 4 read() (bstem.platform.re_cord, 4 read() (bstem.device), 20 pulse_width (bstem.device.Lgscerve attribute), 20 read (class in bstem.device), 20 formain() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10 position (bstem.device.AbsEncoder attribute), 17 position (bstem.device.Becoder attribute), 20 position (bstem.device.Gevo attribute), 20 position (bstem.device.Serve attribute), 20 position (bstem.device.Serve attribute), 20 position (bstem.device.Serve attribute), 25		,,,.	
Magnetometer (class in bstem.sensor), 40 main() (in module bstem.tutorials.example_5a_local_loggin_gulse_width (bstem.device.PMOutput method), 21 pul.window_queue() (bstem.display_thread.DisplayThread main() (in module bstem.tutorials.example_5b_remote_loggin_gclient_width() (bstem.device.PMOutput method), 21 pul.window_queue() (bstem.display_thread.DisplayThread main() (in module bstem.tutorials.example_6a_local_plotting). 10 main() (in module bstem.tutorials.example_6b_remote_plot_QliadEscover)(class in bstem.device), 23 10 main() (in module bstem.tutorials.example_7_parameter_tum_g), 10 max_cell_voltage (bstem.device.PowerGauge attribute), 21 MaxSonar (class in bstem.sonar), 41 MaxSonar (class in bstem.sonar), 41 MotionHardware (class in bstem.sonar), 41 MotionHardware (class in bstem.device), 20 N name (bstem.platform.ad_cord.AdCord attribute), 2 name (bstem.platform.ad_cord.AdCord attribute), 5 next() (bstem.video.Video method), 42 pause() (bstem.camera.Camera method), 44 period (bstem.control.ControlLoop attribute), 16 Plot_class in bstem.plot), 36 plot_local_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_raind() (in module bstem.test.test_plot), 7 pop() (bstem.campera.Camera method), 31 position (bstem.device.Poserver method), 31 position (bstem.device.Encoder attribute), 23 position (bstem.device.Camera ttribute), 23 position (bstem.device.Servo attribute), 23 position (bstem.device.Servo attribute), 23 position (bstem.device.Camera ttribute), 23 position (bstem.device.Servo attribute), 24 pulse_width (bstem.device.Servo attribute), 24 pulse_widt	M		
main() (in module bstem.hutorials.example_5a_local_logging_yllse_width() (bstem.device.Prvo attribute), 24 main() (in module bstem.tutorials.example_5a_local_logging_glien_glylse_width() (bstem.device.PrwOtuput method), 21 put_window_queue() (bstem.display_thread.DisplayThread main() (in module bstem.tutorials.example_6a_local_plotting), 10 max(cell_voltage (bstem.device.PowerGauge attribute), 21 max(Sonar (class in bstem.sonar), 41 MaxSonar (class in bstem.sonar), 41 MaxSonar (class in bstem.sonar), 41 MaxSonar/Array (class in bstem.motion_hardware), 35 Motor (class in bstem.sonar), 41 MotionHardware (class in bstem.motion_hardware), 35 Motor (class in bstem.device), 20 MulticolorLed (class in bstem.device), 20 MulticolorLed (class in bstem.device), 20 N name (bstem.platform.rc_cord.AcCord attribute), 5 next() (bstem.outrol.Contro	Magneto	meter (class in hetem sensor) 40	
main() (in module bstem.tutorials.example_5a_local_logging_usle_width() (bstem.device.PPMOutput method), 21 put_window_queue() (bstem.device.play_thread_Display_Thread main() (in module bstem.tutorials.example_5b_remote_logging_client_Method), 27 pwm_frequency (bstem.device.Servo attribute), 24 main() (in module bstem.tutorials.example_6a_local_plotting), 10 main() (in module bstem.tutorials.example_7_parameter_turn_ng), 10 main() (in module bstem.tutorials.example_7_parameter_turn_ng), 10 main() (in module bstem.tutorials.example_7_parameter_turn_ng), 10 max_cell_voltage (bstem.device.PowerGauge attribute), 2 14 MaxSonar (class in bstem.sonar), 41 MaxSonarArray (class in bstem.sonar), 41 MotionHardware (class in bstem.motion_hardware), 35 Motor (class in bstem.device), 20 N name (bstem.platform.ad_cord.AdCord attribute), 5 max() (bstem.platform.rc_cord.ReCord attribute), 5 mext() (bstem.camera.Camera method), 42 P  P  P  Record (class in bstem.platform.rc_cord.keCord attribute), 5 mext() (bstem.camera.Camera method), 14 period (bstem.control.ControlLoop attribute), 5 mext() (bstem.camera.Camera method), 14 period (bstem.control.ControlLoop attribute), 5 mext() (bstem.camera.Camera method), 14 period (bstem.control.ControlLoop attribute), 5 mext() (bstem.control.ControlLoop attribute), 10 plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rainbow_example() (in module bstem.tutorials.remoteplot_client() (in module bstem.tutorials.remotep	_		
put_window_queue() (bstem.display_thread.DisplayThread main() (in module bstem.tutorials.example_6a_local_plottine_10	main() (i	n module betem tutorials example 5a local loggir	
main() (in module bstem.tutorials.example_6a_local_plotting).  main() (in module bstem.tutorials.example_6a_local_plotting).  main() (in module bstem.tutorials.example_6b_remote_plot_QliadExecter) (class in bstem.device), 23  10  main() (in module bstem.tutorials.example_6b_remote_plot_QliadExecter) (class in bstem.device), 23  10  main() (in module bstem.tutorials.example_6b_remote_plot_QliadExecter) (class in bstem.device), 23  10  main() (in module bstem.tutorials.example_6b_remote_plot_QliadExecter) (class in bstem.device), 23  10  main() (in module bstem.tutorials.example_6b_remote_plot_QliadExecter) (class in bstem.device), 23  max_cell_voltage (bstem.device, PowerGauge attribute), 2  max_cell_voltage (bstem.device, PowerGauge attribute), 23  max_cell_voltage (bstem.device, PowerGauge attribute), 24  max_cell_voltage (bstem.device, PowerGauge attribute), 25  max_cell_voltage (bstem.device, PowerGauge attribute), 24  max_cell_voltage (bstem.device, PowerGauge attribute), 25  mand() (in module bstem.test.test_plot), 7  max_cell_voltage (bstem.device, PowerGauge attribute), 24  main() (in module bstem.test.test_plot), 7  max_cell_voltage (bstem.device, PowerGauge attribute), 24  main() (in module bstem.test.test_plot), 7  max_cell_voltage (bstem.device, PowerGauge attribute), 24  main() (in module bstem.test.test_plot, 7  max_value (bstem.sensor.Magnetometer attribute), 14  Rc (class in bstem.device, 24  RC_DS (bstem.platform.re_cord.ReCord attribute), 5  RC_MULTLDS (bstem.platform.re_cord, 4  RC_DS (bstem.log, VariableServer method), 31  remaingly (bstem.control.ControlLoop method), 16  RemoteLogger (class in bstem.tutorials.logging_server), 10  remoteplot_server() (in module bstem.tutorials.remoteplot_server), 10  remoteplot_server() (in m	mam() (n	o	put window queue() (bstem.display thread.DisplayThread
main() (in module bstem.tutorials.example_6a_local_plotting).  main() (in module bstem.tutorials.example_6b_remote_plot_QliadEneroter)(class in bstem.device), 23  10  main() (in module bstem.tutorials.example_fob_remote_plot_QliadEneroter)(class in bstem.device), 23  10  main() (in module bstem.tutorials.example_fob_remote_plot_QliadEneroter)(class in bstem.device), 23  10  max_cell_voltage (bstem.device.PowerGauge attribute), 21  MaxSonar (class in bstem.sonar), 41  MaxSonarArray (class in bstem.sonar), 41  MaxSonarArray (class in bstem.sonar), 41  MaxSonarArray (class in bstem.motion_hardware), 35  Motor (class in bstem.device), 20  MulticolorLed (class in bstem.device), 20  P  mame (bstem.platform.a_cord.AdCord attribute), 5  next() (bstem.platform.rc_cord.ReCord attribute), 5  next() (bstem.platform.rc_cord.ReCord attribute), 5  next() (bstem.camera.Camera method), 42  P  pause() (bstem.camera.Camera method), 14  period (bstem.control.Controll.cop attribute), 16  Plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_rand() (in module bstem.test.test_plot), 7  raw_value (bstem.analog_fliter.Analog_friiter attribute), 40  raw_value (bstem.analog_fliter.Analog_friiter attribute), 40  raw_value (bstem.analog_fliter.Analog_friiter attribute), 40  raw_value (bstem.sonar.MaxSonarArray attribute), 41  Rc (class in bstem.device.), 24  RC_DK (bstem.platform.rc_cord.ReCord attribute), 5  RC_MULTI_DS (bstem.platform.rc_cord., 4  read() (bstem.log_LogServer method), 31  register() (bstem.log_Logging_client), 10  plot_rainbow_example() (in module bstem.test.test_plot), 7  remotelo.ger_client() (in module bstem.test.test_plot), 7  remotelo.ger_closs in bstem.device.log method), 16  remotelo.ger_client() (in module bstem.test.device.log method), 16  remotelo.ger_client() (in module bstem.test.device.log method), 16  remotelo.ger_client() (in module bstem.test.devi	main() (ii	n module betem tutorials example 5b, remote logo	
main() (in module bstem.tutorials.example_6b_remote_plot_QliadE_neoter)(class in bstem.device), 23  main() (in module bstem.tutorials.example_7_parameter_tunRg), 10  max_cell_voltage (bstem.device.PowerGauge attribute), 21  MaxSonar (class in bstem.sonar), 41  MaxSonar/Array (class in bstem.sonar), 41  MaxSonar/Array (class in bstem.motion_hardware), 35  Motor (class in bstem.device), 20  MulticolorLed (class in bstem.device), 20  P  mame (bstem.platform.rc_cord.AcCord attribute), 2  name (bstem.platform.rc_cord.AcCord attribute), 2  name (bstem.platform.rc_cord.AcCord attribute), 2  name (bstem.platform.rc_cord.AcCord attribute), 2  name (bstem.camera.Camera method), 42  P  pause() (bstem.camera.Camera method), 14  period (bstem.control.ControlLoop attribute), 16  Plot (class in bstem.plot), 36  plot_rand() (in module bstem.test.test_plot), 7  pop() (bstem.control.Scheduler class in bstem.device.AbsEncoder attribute), 17  pop() (bstem.device.AbsEncoder attribute), 17  position (bstem.device.Encoder attribute), 23  position (bstem.device.SmartServo attribute), 23  position (bstem.device.SmartServo attribute), 23  position (bstem.device.SmartServo attribute), 23  position (bstem.medevice.SmartServo attribute), 23  position (bstem.device.SmartServo attribute), 25	mam() (n		pwm_frequency (bstem.device.Servo attribute), 24
main() (in module bstem.tutorials.example_7_parameter_turng), 10  max_cell_voltage (bstem.device.PowerGauge attribute), 21  MaxSonar (class in bstem.sonar), 41  MaxSonarArray (class in bstem.sonar), 41  MotionHardware (class in bstem.motion_hardware), 35  Motor (class in bstem.device), 20  MulticolorLed (class in bstem.device), 20  N  name (bstem.platform.ad_cord.AdCord attribute), 2  name (bstem.platform.ad_cord.AdCord attribute), 5  next() (bstem.platform.ad_cord.AdCord attribute), 5  next() (bstem.oplatform.ac_cord.ReCord attribute), 5  next() (bstem.oplatform.ac_cord.accord attribute), 20  pause() (bstem.camera.Camera method), 42  P  pause() (bstem.camera.Camera method), 14  period (bstem.control.ControlLoop attribute), 16  Plot (class in bstem.plot), 36  plot_oal_example() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_rainbow_example() (in module bstem.tutorials.remoteplot_client), 10  plot_rainbow_example() (in module bstem.tutorials.remoteplot_server), 11  popt() (bstem.log.LogServer method), 31  popitem() (bstem.device.Becoder attribute), 17  poposition (bstem.device.Becoder attribute), 29  position (bstem.device.Servo attribute), 24  position (bstem.device.Servo attribute), 25  RCGM(LTI_DS (bstem.platform.rc_cord.ReCord attribute), 5  RCC_MULTI_DS (bstem.platform.rc_cord.ReCord attribute), 5  RCC_MULTI_DS (bstem.platform.rc_cord.ReCord attribute), 5  RCC_MULTI_DS (bstem.control.ControlLoop method), 31  register() (bstem.log.VariableServer method), 31  register() (bstem.log.VariableServer method), 31  remition() (bstem.log.VariableServer method), 31  remotelogger_client() (in module bstem.tutorials.logging_client), 10  remoteplo_client() (in module bstem.tutorials.logging_server), 10  remoteplo_server() (in module bstem.tutorials.remoteplot_client), 10  remoteplo_server() (in module bstem.tutorials.remoteplot_server), 11  remove() (bstem.log.VariableServer class method), 16  remoteplo_server() (in module bstem.tutorials.remoteplot_server), 11  remove() (bstem.log.LogServer method), 20	main() (i		<b>Q</b>
main() (in module bstem.tutorials.example_7_parameter_unng),  10  max_cell_voltage (bstem.device.PowerGauge attribute), 21  MaxSonar (class in bstem.sonar), 41  MaxSonarArray (class in bstem.sonar), 41  MotionHardware (class in bstem.motion_hardware), 35  Motor (class in bstem.device), 20  MulticolorLed (class in bstem.device), 20  MulticolorLed (class in bstem.device), 20  MulticolorLed (class in bstem.device), 20  N  name (bstem.platform.ac_cord.AdCord attribute), 5  next() (bstem.platform.rc_cord.RcCord attribute), 5  next() (bstem.camera.Camera method), 42  P  pause() (bstem.camera.Camera method), 14  period (bstem.control.ControlLoop attribute), 16  Plot (class in bstem.plot), 36  plot_jocal_example() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_rando() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_rando() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_rando() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_mindo() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_mindo() (in module bstem.tutorials.example_form.rc_cord.RcCord attribute), 24  position (bstem.device.Bncoder attribute), 17  popition (bstem.device.Bncoder attribute), 29  position (bstem.device.Servo attribute), 24  position (bstem.device.Servo attribute), 25  RCOrd (class in bstem.test.test_plot), 7  RcCord (class in bstem.platform.rc_cord.RcCord attribute), 5  RcCord (class in bstem.platform.rc_cord.RcCord attribute), 34  register() (bstem.log.LogServer method), 31  register() (bstem.log.LogServer method), 31  register() (bstem.log.logging_client), 10  remotelogger_client() (in module bstem.tutorials.remoteplot_client), 10  remotelogger_server() (in module bstem.tutorials.remoteplot_server), 11  remotelogger_client() (in module bstem.tutorials.remoteplot_server), 11  remove() (bstem.device.Motor method), 20  reset() (bstem.platform.rc_cord.RcCord method), 2  reset() (bstem.platform.rc_cord.RcCord attribute), 29  reset() (bstem.platform.rc_ord.RcCord attribute)	main() (in		QliadEscover)(class in bstem.device), 23
max_cell_voltage (bstem.device.PowerGauge attribute), 21  MaxSonar (class in bstem.sonar), 41  MaxSonar (class in bstem.sonar), 41  MotionHardware (class in bstem.motion_hardware), 35  Motor (class in bstem.device), 20  MulticolorLed (class in bstem.device), 20  P  RC_DS (bstem.platform.rc_cord.ReCord attribute), 5  RC_MULTI_DS (bstem.platform.rc_cord), 4  read() (bstem.log_LogServer method), 31  register() (bstem.log_VariableServer method), 31  register() (bstem.log_VariableServer method), 31  remoteloger_client() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_rand() (in module bstem.test.test_plot), 7  popit (bstem.log_LogServer method), 31  popitem() (bstem.log_LogServer method), 31  popitem() (bstem.log_LogServer method), 31  popition (bstem.device.AbsEncoder attribute), 17  position (bstem.device.Encoder attribute), 24  position (bstem.device.Servo attribute), 25  RC_MULTI_DS (bstem.platform.rc_cord.ReCord attribute), 5  RC_MULTI_DS (bstem.platform.rc_cord), 4  read() (bstem.log_LogServer method), 31  register() (bstem.log_VariableServer method), 31  register() (bstem.log_VariableServer method), 16  RemoteLogger (class in bstem.log), 33  remotelogger_client() (in module bstem.tutorials.remoteplot_client), 10  remoteplot_server() (in module bstem.tutorials.remoteplot_client), 10  remoteplot_server() (in module bstem.tutorials.remoteplot_server), 11  reset() (bstem.log_LogServer method), 20  reset() (bstem.log_LogServer method), 20  reset() (bstem.log_LogServer method), 32  reset() (bstem.log_LogServer method), 32  reset() (bstem.log_LogServer method), 32  reset() (bstem.log_LogServer method), 32  reset() (bstem	main() (i		n <b>R</b> g),
max_cell_voltage (bstem.device.PowerGauge attribute), 2  MaxSonar (class in bstem.sonar), 41  MaxSonarArray (class in bstem.sonar), 41  MotionHardware (class in bstem.motion_hardware), 35  Motor (class in bstem.device), 20  MulticolorLed (class in bstem.device), 20  N  name (bstem.platform.ad_cord.AdCord attribute), 2  name (bstem.platform.cord.RcCord attribute), 5  next() (bstem.video.Video method), 42  P  pause() (bstem.camera.Camera method), 14  period (bstem.control.ControlLoop attribute), 16  Plot (class in bstem.plot), 36  plot_local_example() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_rand() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_rand() (in module bstem.log.LogServer method), 31  popitem() (bstem.log.LogServer method), 31  popition (bstem.device.AbsEncoder attribute), 17  position (bstem.device.LgServo attribute), 24  position (bstem.device.Servo attribute), 25  RC_MULTI_DS (bstem.platform.rc_cord.RcCord attribute), 5  RC_MULTI_DS (bstem.platform.rc_cord.RcCord attribute), 5  RC_MULTI_DS (bstem.platform.rc_cord.RcCord attribute), 5  RC_MULTI_DS (bstem.platform.rc_cord, RcCord attribute), 5  RC_MULT_DS (bstem.platform.rc_cord, RcCord attribute), 10  remaining() (bstem.control.ControlLoop method), 31  remaining() (bstem.log.LogServer method), 31  remotelogger_client() (in module bstem.tutorials.logging_client), 10  remotelogger_client() (in module bstem.tutorials.remoteplot_client), 10  remoteplot_server() (in module bstem.tutorials.remoteplot_client), 10  remoteplot_server() (in module bstem.test.ftest_plot), 7  remotelogger client() (in module bstem.test.fte			random() (in module bstem.test.test_plot), 7
MaxSonar (class in bstem.sonar), 41  MaxSonarArray (class in bstem.sonar), 41  MotionHardware (class in bstem.motion_hardware), 35  Motor (class in bstem.device), 20  MulticolorLed (class in bstem.device), 20  N  N  name (bstem.platform.ad_cord.AdCord attribute), 2 name (bstem.platform.rc_cord.RcCord attribute), 5  next() (bstem.video.Video method), 42  P  pause() (bstem.camera.Camera method), 14  period (bstem.control.ControlLoop attribute), 16  Plot (class in bstem.plot), 36  plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_raind() (in module bstem.test.test_plot), 7  pop) (bstem.log.LogServer method), 31  popition (bstem.device.AbsEncoder attribute), 17  position (bstem.device.LegServo attribute), 24  position (bstem.device.Servo attribute), 25  Motior (class in bstem.motion_hardware), 35  RC_MULTI_DS (bstem.platform.rc_cord.RcCord attribute), 5  RCCOd (class in bstem.platform.rc_cord), 4  read() (bstem.log.LogServer method), 34  remaining() (bstem.control.ControlLoop method), 16  RemoteLogger (class in bstem.log), 33  remotelogger_client() (in module bstem.tutorials.remoteplot_client), 10  remotelogger_server() (in module bstem.tutorials.remoteplot_server), 10  remotelogger_server() (in module bstem.tutorials.remoteplot_server), 11  remove() (bstem.control.Scheduler class method), 16  remotelogger.client() (in module bstem.tutorials.remoteplot_server), 10  remotelogger_client() (in module bstem.tutorials.remoteplot_client), 10  remotelogger.client() (in module bstem.tutorials.remoteplot_client), 20  reset() (bstem.platform.rc_cord.RcCord method), 20  reset()	max_cell		raw_value (bstem.analog_filter.AnalogFilter attribute), 12
MaxSonarArray (class in bstem.sonar), 41 MotionHardware (class in bstem.motion_hardware), 35 Motor (class in bstem.device), 20 MulticolorLed (class in bstem.device), 20  N name (bstem.platform.ad_cord.AdCord attribute), 2 name (bstem.platform.rc_cord.RcCord attribute), 5 next() (bstem.platform.rc_cord.RcCord attribute), 5 next() (bstem.platform.rc_cord.RcCord attribute), 5 next() (bstem.control.ControlLoop attribute), 16 Polt (class in bstem.plot), 36 plot_ocal_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rand() (in module bstem.test.test_plot), 7 pop() (bstem.log.LogServer method), 31 position (bstem.device.AbsEncoder attribute), 29 position (bstem.device.Egroa ttribute), 24 position (bstem.device.Servo attribute), 24 position (bstem.device.Servo attribute), 25  McClass in bstem.device,), 24 RC_DS (bstem.platform.rc_cord.RcCord attribute), 5 RC_MULTI_DS (bstem.platform.rc_cord.NeCord attribute), 5 RC_MULTI_DS (bstem.platform.rc_cord.NeCord attribute), 5 RC_MULTI_DS (bstem.platform.rc_cord.NeCord attribute), 3 read() (bstem.log.LogServer method), 31 register() (bstem.log.LogServer method), 31 register() (bstem.log.LogServer method), 31 register() (bstem.log.LogServer method), 31 register() (bstem.log.LogIntribute), 20 remotelogger_client() (in module bstem.tutorials.logging_client), 10 remotelogger_client() (in module bstem.tutorials.remoteplot_client), 10 remotelogger_client() (in module bstem.tutorials.remoteplot_client), 10 remotelogger_client() (bstem.l		<b>-</b> -	raw_value (bstem.sensor.Magnetometer attribute), 40
MotionHardware (class in bstem.motion_hardware), 35 Motor (class in bstem.device), 20 MulticolorLed (class in bstem.device), 20 N name (bstem.platform.ad_cord.AdCord attribute), 2 name (bstem.platform.c_cord.RcCord attribute), 5 next() (bstem.platform.c_cord.RcCord attribute), 5 next() (bstem.video.Video method), 42  P  pause() (bstem.camera.Camera method), 14 period (bstem.control.ControlLoop attribute), 16 Plot (class in bstem.plot), 36 plot_local_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rand() (in module bstem.test.test_plot), 7 pop() (bstem.log.LogServer method), 31 popition (bstem.device.AbsEncoder attribute), 29 position (bstem.device.Servo attribute), 24 position (bstem.device.Servo attribute), 24 position (bstem.device.SmartServo attribute), 25  MC_DS (bstem.platform.rc_cord.RcCord attribute), 5 RC_MULTI_DS (bstem.platform.rc_cord.RcCord attribute), 5 RCCMULTI_DS (bstem.platform.rc_cord.RcCord attribute), 5 RCCMULTI_DS (bstem.platform.rc_cord.RcCord attribute), 3 read() (bstem.platform.rc_cord.NcCord method), 31 register() (bstem.log.LogServer method), 31 remotelogger_client() (in module bstem.tutorials.logging_client), 10 remotelogger_server() (in module bstem.tutorials.remoteplot_client) (in module bstem.tutorials.remoteplot_client), 10 remotelogger_server() (in module bstem.tutorials.remoteplot_client), 10 remotelogger_server() (in module bstem.tutorials.remoteplot_client), 10 remotelogger_client() (in module bstem.tutorials.logging_server), 10 remotelogger_client() (in module bstem.tutorials.remoteplot_client), 10 remotelogger_client() (in module bstem.tutorials.logging_server), 10 remotelogger_client() (in module bstem.tutorials.logging_client), 10 remotelogger_cl			raw_value (bstem.sonar.MaxSonarArray attribute), 41
Motor (class in bstem.device), 20  MulticolorLed (class in bstem.device), 20  N  name (bstem.platform.ad_cord.AdCord attribute), 2 name (bstem.platform.rc_cord.RcCord attribute), 5 next() (bstem.video.Video method), 42  P  pause() (bstem.camera.Camera method), 14 period (bstem.control.ControlLoop attribute), 16 Plot (class in bstem.plot), 36 Plot (class in bstem.plot), 36 plot_local_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rand() (in module bstem.test.test_plot), 7 pop() (bstem.log.LogServer method), 31 popitem() (bstem.device.AbsEncoder attribute), 17 position (bstem.device.Encoder attribute), 23 position (bstem.device.Servo attribute), 24 position (bstem.device.Servo attribute), 24 position (bstem.device.Servo attribute), 24 position (bstem.device.SmartServo attribute), 25  RC_MULTI_DSS (bstem.platform.rc_cord.RcCord attribute), 5 RCCord (class in bstem.platform.rc_cord), 4 read() (bstem.platform.rc_cord), 4 read() (bstem.log.LogServer method), 34 remaining() (bstem.control.ControlLoop method), 16 RemoteLogger (class in bstem.log.NariableServer method), 16 RemoteLogger (class in bstem.log.), 33 remotelogger_client() (in module bstem.tutorials.logging_server), 10 remoteplot_elient() (in module bstem.tutorials.remoteplot_elient), 10 remoteplot_server() (in module bstem.tutorials.remoteplot_server), 11 remove() (bstem.control.Scheduler class method), 14 reset() (bstem.platform.rc_cord.RcCord method), 2 reset() (bstem.platform.rc_cord.RcCord method), 2 reset() (bstem.platform.rc_cord.RcCord method), 2 reset() (bstem.platform.rc_cord.RcCord method), 32 remotelogger_client() (in module bstem.tutorials.logging_server), 10 remoteplot_server() (in module bstem.tutorials.remoteplot_client), 10 remoteplot_server() (in module bstem.tutorials.remoteplot_client), 10 remoteplot_server() (in module bstem.tutorials.remoteplot_client), 10 remoteplot_server() (in server), 11 remove() (bstem.control.Scheduler class me			Rc (class in bstem.device), 24
MulticolorLed (class in bstem.device), 20  N  N  name (bstem.platform.ad_cord.AdCord attribute), 2 name (bstem.platform.rc_cord.ReCord attribute), 5 next() (bstem.video.Video method), 42  P  pause() (bstem.camera.Camera method), 14 period (bstem.control.ControlLoop attribute), 16 Plot (class in bstem.plot), 36 plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_raind() (in module bstem.test.test_plot), 7 pop() (bstem.log.LogServer method), 31 popitem() (bstem.log.LogServer method), 31 popition (bstem.device.LgServo attribute), 19 position (bstem.device.Servo attribute), 24 position (bstem.device.Servo attribute), 24 position (bstem.device.SmartServo attribute), 25  RCCord (class in bstem.platform.rc_cord), 4 read() (bstem.log.LogServer method), 31 register() (bstem.log.VariableServer method), 34 remaining() (bstem.control.ControlLoop method), 16 RemoteLogger_client() (in module bstem.tutorials.logging_client), 10 remoteplot_client() (in module bstem.tutorials.remoteplot_client), 10 remoteplot_client() (in module bstem.tutorials.remoteplot_client), 10 remoteplot_server() (in semoteplot_server), 11 remove() (bstem.control.Scheduler class method), 14 reset() (bstem.platform.ac_cord.AdCord method), 20 reset() (bstem.platform.rc_cord.Record method), 20 reset() (bstem.platform.rc_cord.Record method), 20			
RCCord (class in bstem.platform.rc_cord), 4 read() (bstem.log.LogServer method), 31 register() (bstem.log.VariableServer method), 34 remaining() (bstem.log.VariableServer method), 34 remaining() (bstem.log.VariableServer method), 34 remaining() (bstem.log.VariableServer method), 34 remaining() (bstem.log.VariableServer method), 36 remoteLogger (class in bstem.log), 33 remotelogger_client() (in module bstem.tutorials.cample_of_alocal_plotting), 10 plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_raind() (in module bstem.test.test_plot), 7 pop() (bstem.log.LogServer method), 31 popitem() (bstem.log.LogServer method), 31 popitem() (bstem.device.AbsEncoder attribute), 17 position (bstem.device.Encoder attribute), 29 position (bstem.device.Servo attribute), 24 position (bstem.device.Servo attribute), 25  RCCord (class in bstem.platform.rc_cord), 4 read() (bstem.log.LogServer method), 31 register() (bstem.log.VariableServer method), 34 remaining() (bstem.log.VariableServer method), 36 remoteLogger_client() (in module bstem.tutorials.logging_client), 10 remoteplot_client() (in module bstem.tutorials.remoteplot_client), 10 remoteplot_server() (in module bstem.tutorials.remoteplot_server), 11 remotelogger_server() (in module bstem.tutorials.remoteplot_client), 10 remoteplot_server() (in semoteplot_server), 11 register() (bstem.log.VariableServer method), 20 remoteplot_server() (in semoteplot_server), 10 remoteplot_server() (in semoteplot_server), 10 remoteplot_server() (in semoteplot_server), 10 remotep			RC_MULTI_DS (bstem.platform.rc_cord.RcCord at-
read() (bstem.log.LogServer method), 31 register() (bstem.log.LogServer method), 34 remaining() (bstem.control.ControlLoop method), 16 RemoteLogger (class in bstem.log), 33 remotelogger_client() (in module bstem.tutorials.logging_client), 10 remotelogger_server() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rand() (in module bstem.test.test_plot), 7 popition (bstem.log.LogServer method), 31 register() (bstem.log.LogServer method), 34 remaining() (bstem.control.ControlLoop method), 16 RemoteLogger (class in bstem.log), 33 remotelogger_client() (in module bstem.tutorials.logging_client), 10 remoteplot_client() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_raind() (in module bstem.test.test_plot), 7 popition (bstem.log.LogServer method), 31 register() (bstem.log.VariableServer method), 34 remaining() (bstem.control.ControlLoop method), 16 RemoteLogger (class in bstem.log), 33 remotelogger_client() (in module bstem.tutorials.logging_server), 10 remoteplot_client() (in module bstem.tutorials.example_ba_local_plotting), 10 remoteplot_client() (in module bstem.tutorials.exemoteplot_client), 10 remoteplot_server() (in module bstem.tutorials.remoteplot_server() (in module bstem.tutorials.exemoteplot_server() (in semoteplot_server() (in semoteplot_server() (in semoteplot_server() (in semoteplot_server() (in semoteplot_server() (in semoteplot_server() (semoteplot_server() (semoteplot_server() (semoteplot_server() (setem.log.LogServer method), 14 reset() (bstem.dog. LogServer method), 20 reset() (bst	Multicol	orLed (class in bstem.device), 20	
read() (bstem.log.LogServer method), 31 register() (bstem.log.LogServer method), 34 remaining() (bstem.log.LogServer method), 34 remaining() (bstem.control.ControlLoop method), 16 RemoteLogger (class in bstem.log), 33 remotelogger_client() (in module bstem.tutorials.logging_client), 10 remotelogger_server() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rand() (in module bstem.test.test_plot), 7 pop() (bstem.log.LogServer method), 31 position (bstem.device.AbsEncoder attribute), 17 position (bstem.device.CugadEncoder attribute), 24 position (bstem.device.Servo attribute), 24 position (bstem.device.SmartServo attribute), 25  read() (bstem.log.LogServer method), 34 register() (bstem.log.LogServer method), 34 remoteLogger_client() (in module bstem.tutorials.logging_server), 10 remotelogger_server() (in module bstem.tutorials.remoteplot_client), 10 remotelogger_server() (in module bstem.tutorials.remoteplot_client), 10 remotelogger_server() (in module bstem.tutorials.remoteplot_client), 10 remotelogger_client() (in module bstem.tutorials.logging_server), 10 remotelogger_server() (in module bstem.tutorials.logging_server), 10 remotelogger_client() (bstem.control.Server), 11 remoteplo_client() (bstem.device.Module class method), 16 remotelogger_client() (bstem.device.Module class method), 20 reset() (bstem.device.Module clas	N		
name (bstem.platform.rc_cord.RcCord attribute), 5 next() (bstem.video.Video method), 42  P  RemoteLogger (class in bstem.log), 33  remotelogger_client() (in module bstem.tutorials.logging_client), 10  remotelogger_server() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_raind() (in module bstem.test.test_plot), 7  pop() (bstem.log.LogServer method), 31  popition (bstem.device.AbsEncoder attribute), 19  position (bstem.device.Encoder attribute), 24  position (bstem.device.Servo attribute), 24  position (bstem.device.SmartServo attribute), 25  remaining() (bstem.control.ControlLoop method), 16  RemoteLogger (class in bstem.log), 33  remotelogger_client() (in module bstem.tutorials.logging_server), 10  remotelogger_server() (in module bstem.tutorials.remoteplot_client), 10  remotelogger_server() (in module bstem.tutorials.remoteplot_server) (in module bstem.tutorials.remoteplot_server), 11  remove() (bstem.control.ControlLoop method), 16  remoteLogger (class in bstem.log), 33  remotelogger_client() (in module bstem.tutorials.logging_server), 10  remotelogger_server() (in module bstem.tutorials.remoteplot_client() (in module bstem.tutorials.remoteplot_server() (in module bstem.tutorials.premoteplot_server() (in module bstem.tutorials.premoteplot_server() (in module bstem.tutorials.premoteplot_server() (bstem.control.Scheduler class method), 16  remotelogger_server() (in bstem.device.Mosomethod) (bstem.device.Mosomethod), 20  remotelogger_server() (in bstem.device.Mosomethod), 20  remoteplot_server() (in bstem.device.Mosomethod), 20  remoteplot_server() (bstem.server() (bstem.log.Scheduler class method), 16  remoteplot_server() (bstem.device.Mosomethod), 20  reset() (bstem.platform.device.		1.6	
P RemoteLogger (class in bstem.log), 33 remotellogger_client() (in module bstem.tutorials.logging_client), 10 remotellogger_client() (in module bstem.tutorials.logging_client), 10 remotellogger_client() (in module bstem.tutorials.logging_client), 10 remotellogger_client() (in module bstem.tutorials.logging_server), 10 remotellogger_server() (in module bstem.tutorials.example_6a_local_plotting), 10 remotellogger_server() (in module bstem.tutorials.example_6a_local_plotting), 10 remotellogger_server() (in module bstem.tutorials.remoteplot_client(), 10 remotellogger_server() (in module bstem.tutorials.remoteplot_client), 10 remotellogger_server() (in module bstem.tutorials.remoteplot_client(), 10 remotellogger_server() (in module bstem.tutorials.remoteplot_client), 10 remotellogger_server() (in module bstem.tutorials.remoteplot_client), 10 remotellogger_server() (in module bstem.tutorials.remoteplot_client() (in module bstem.tutorials.remoteplot_server() (in module bstem.tutorials.remoteplot_client(), 10 remotellogger_client() (in module bstem.tutorials.logging_server), 10 remotellogger_client() (in module bstem.tutorials.logging_server() (in module bstem.tutorials.remoteplot_client() (in module bstem.tutorials.remoteplot_client() (in module bstem.tutorials.remoteplot_server() (in module bstem.tutorials.remoteplot_client() (in module bstem.tutorials.remoteplot_client() (in module bstem.tutorials.remoteplot_server() (in module bstem.tutorials.remoteplot_server() (in module bstem.tutorials.remoteplot_client() (in module bstem.tutorials.remoteplot_client() (in module bstem.tutorials.remoteplot_server() (in module bstem.tutorials.remoteplot_server() (in module bstem.tutorials.remoteplot_client() (in module bstem.tutorials.remoteplot_server() (bstem.platform.cluentary (in module bstem.tutorials.remoteplot_server() (bstem.gutorials.remoteplot_server() (in module bstem.tutorials.remoteplot_server() (bstem.butorials.remoteplot_server() (bstem.platform.dutorials.remoteplot_server() (bstem.gutorials.remoteplot_server() (bst			
remotelogger_client() (in module bstem.tutorials.logging_client), 10  Plot (class in bstem.plot), 36  Plot_local_example() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_raind() (in module bstem.test.test_plot), 7  pop() (bstem.log.LogServer method), 31  popitem() (bstem.log.LogServer method), 31  position (bstem.device.Encoder attribute), 19  position (bstem.device.QuadEncoder attribute), 24  position (bstem.device.Servo attribute), 24  position (bstem.device.SmartServo attribute), 25  remotelogger_client() (in module bstem.tutorials.logging_server), 10  remotelogger_server() (in module bstem.tutorials.logging_server), 10  remotelogger_server() (in module bstem.tutorials.remoteplot_client), 10  remotelogger_server() (in module bstem.tutorials.logging_server), 10  remoteloger_server() (in module bstem.tutorials.logging_server), 10  remoteloger_server() (in module bstem.tutorials.logging_server), 10  remotelog_server() (in semoteplot_server() (bstem.tutorials.remoteplot_server() (bstem.tutorials.remoteplot_server() (bstem.tutorials.remoteplot_server() (bstem.tutorials.remoteplot_server() (bstem.tutorials.remoteplot_server() (bstem.toging_sevarper() (bstem.togi		•	
bstem.tutorials.logging_client), 10  pause() (bstem.camera.Camera method), 14  period (bstem.control.ControlLoop attribute), 16  Plot (class in bstem.plot), 36  plot_local_example() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10  plot_rand() (in module bstem.test.test_plot), 7  pop() (bstem.log.LogServer method), 31  popitem() (bstem.log.LogServer method), 31  position (bstem.device.AbsEncoder attribute), 19  position (bstem.device.QuadEncoder attribute), 20  position (bstem.device.Servo attribute), 24  position (bstem.device.Servo attribute), 24  position (bstem.device.SmartServo attribute), 25  Bottm.tutorials.logging_client), 10  remotelogger_server() (in module bstem.tutorials.remoteplot_client), 10  remoteplot_client() (in module bstem.tutorials.remoteplot_server() (in module bstem.tutorials.remoteplot_server), 11  remove() (bstem.control.Scheduler class method), 16  remove_adaptation() (bstem.camera.Camera method), 14  reset() (bstem.device.Motor method), 20  reset() (bstem.platform.ad_cord.AdCord method), 2  reset() (bstem.platform.rc_cord.RcCord method), 5  reset_subscriptions() (bstem.log.LogServer method), 32  resize() (bstem.image_wrapper.ImageWrapper method), 29  RGB (bstem.image_wrapper.ImageWrapper attribute), 29	next() (b	stem.video.Video method), 42	
pause() (bstem.camera.Camera method), 14 period (bstem.control.ControlLoop attribute), 16 Plot (class in bstem.plot), 36 plot_local_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rand() (in module bstem.test.test_plot), 7 pop() (bstem.log.LogServer method), 31 popitem() (bstem.device.AbsEncoder attribute), 17 position (bstem.device.LgServo attribute), 20 position (bstem.device.QuadEncoder attribute), 24 position (bstem.device.SmartServo attribute), 24 position (bstem.device.SmartServo attribute), 25  remotelogger_server() (in module bstem.tutorials.logging_ctient), 10 remotelogger_server() (in module bstem.tutorials.remoteplot_client), 10 remoteplot_server() (in module bstem.tutorials.remoteplot_server() (bstem.control.Scheduler class method), 16 remove_adaptation() (bstem.camera.Camera method), 14 reset() (bstem.platform.ad_cord.AdCord method), 2 reset() (bstem.platform.rc_cord.RcCord method), 5 reset_subscriptions() (bstem.log.LogServer method), 32 resize() (bstem.image_wrapper.ImageWrapper method), 29  RGB (bstem.image_wrapper.ImageWrapper attribute), 29	P		
period (bstem.control.ControlLoop attribute), 16 Plot (class in bstem.plot), 36	-		
Plot (class in bstem.plot), 36			ee =
plot_local_example() (in module bstem.tutorials.example_6a_local_plotting), 10 remoteplot_server() (in module bstem.tutorials.remoteplot_server), 11 remove() (bstem.tutorials.remoteplot_server), 11 remove() (bstem.control.Scheduler class method), 16 remove_adaptation() (bstem.camera.Camera method), 14 reset() (bstem.device.Motor method), 20 reset() (bstem.platform.ad_cord.AdCord method), 2 reset() (bstem.platform.re_cord.RcCord method), 4 reset() (bstem.platform.re_cord.RcCord method), 5 reset_subscriptions() (bstem.log.LogServer method), 32 resize() (bstem.device.DuadEncoder attribute), 23 position (bstem.device.Servo attribute), 24 position (bstem.device.Servo attribute), 25 RGB (bstem.image_wrapper.ImageWrapper attribute), 29	_		
bstem.tutorials.example_6a_local_plotting), 10 plot_rainbow_example() (in module     bstem.tutorials.example_6a_local_plotting), 10 plot_raind() (in module bstem.test.test_plot), 7 pop() (bstem.log.LogServer method), 31 popitem() (bstem.log.LogServer method), 31 position (bstem.device.AbsEncoder attribute), 17 position (bstem.device.LgServo attribute), 20 position (bstem.device.QuadEncoder attribute), 23 position (bstem.device.Servo attribute), 24 position (bstem.device.Servo attribute), 25  Betting to module bstem.tutorials.remoteplot_server() (in module bstem.tutorials.remoteplot_server), 11 remove() (bstem.control.Scheduler class method), 16 remove_adaptation() (bstem.camera.Camera method), 20 reset() (bstem.platform.ad_cord.AdCord method), 2 reset() (bstem.platform.rc_cord.RcCord method), 5 reset_subscriptions() (bstem.log.LogServer method), 32 resize() (bstem.image_wrapper.ImageWrapper method), 29  RGB (bstem.image_wrapper.ImageWrapper attribute), 29			
plot_rainbow_example() (in module bstem.tutorials.example_6a_local_plotting), 10 plot_rand() (in module bstem.test.test_plot), 7 pop() (bstem.log.LogServer method), 31 position (bstem.device.AbsEncoder attribute), 17 position (bstem.device.LgServo attribute), 20 position (bstem.device.QuadEncoder attribute), 24 position (bstem.device.SmartServo attribute), 25 position (bstem.device.SmartServo attribute), 26 position (bstem.device.SmartServo attribute), 27 position (bstem.de	plot_loca	•	* · · · · · · · · · · · · · · · · · · ·
bstem.tutorials.example_6a_local_plotting), 10 plot_rand() (in module bstem.test.test_plot), 7 pop() (bstem.log.LogServer method), 31 popitem() (bstem.log.LogServer method), 31 position (bstem.device.AbsEncoder attribute), 19 position (bstem.device.LgServo attribute), 20 position (bstem.device.QuadEncoder attribute), 23 position (bstem.device.Servo attribute), 24 position (bstem.device.Servo attribute), 25 position (bstem.device.SmartServo attribute), 25			• - "
plot_rand() (in module bstem.test.test_plot), 7 pop() (bstem.log.LogServer method), 31 popitem() (bstem.log.LogServer method), 31 position (bstem.device.AbsEncoder attribute), 17 position (bstem.device.LgServo attribute), 20 position (bstem.device.QuadEncoder attribute), 23 position (bstem.device.Servo attribute), 24 position (bstem.device.Servo attribute), 25  position (bstem.device.SmartServo attribute), 25  remove_adaptation() (bstem.camera.Camera method), 14 reset() (bstem.device.Motor method), 20 reset() (bstem.platform.ad_cord.AdCord method), 4 reset() (bstem.platform.rc_cord.RcCord method), 5 reset_subscriptions() (bstem.log.LogServer method), 32 resize() (bstem.image_wrapper.ImageWrapper method), 29  RGB (bstem.image_wrapper.ImageWrapper attribute), 29	plot_rain		•
pop() (bstem.log.LogServer method), 31 reset() (bstem.device.Motor method), 20 popitem() (bstem.log.LogServer method), 31 reset() (bstem.platform.ad_cord.AdCord method), 2 position (bstem.device.AbsEncoder attribute), 17 position (bstem.device.Encoder attribute), 19 position (bstem.device.LgServo attribute), 20 position (bstem.device.QuadEncoder attribute), 23 position (bstem.device.Servo attribute), 24 position (bstem.device.Servo attribute), 25 RGB (bstem.image_wrapper.ImageWrapper attribute), 29	1.	1 – – 1 C	
popitem() (bstem.log.LogServer method), 31	-	• •	
position (bstem.device.AbsEncoder attribute), 17 position (bstem.device.Encoder attribute), 19 position (bstem.device.LgServo attribute), 20 position (bstem.device.QuadEncoder attribute), 23 position (bstem.device.Servo attribute), 24 position (bstem.device.SmartServo attribute), 25  reset() (bstem.platform.rc_cord.RcCord method), 5 reset_subscriptions() (bstem.log.LogServer method), 32 resize() (bstem.image_wrapper.ImageWrapper method), 29  RGB (bstem.image_wrapper.ImageWrapper attribute), 29			
position (bstem.device.Encoder attribute), 19 position (bstem.device.LgServo attribute), 20 position (bstem.device.QuadEncoder attribute), 23 position (bstem.device.Servo attribute), 24 position (bstem.device.Servo attribute), 24 position (bstem.device.SmartServo attribute), 25  RGB (bstem.image_wrapper.ImageWrapper attribute), 29			
position (bstem.device.LgServo attribute), 20 reset_subscriptions() (bstem.log.LogServer method), 32 position (bstem.device.QuadEncoder attribute), 23 position (bstem.device.Servo attribute), 24 position (bstem.device.SmartServo attribute), 25 resize() (bstem.image_wrapper.ImageWrapper method), 29 RGB (bstem.image_wrapper.ImageWrapper attribute), 29	-		
position (bstem.device.QuadEncoder attribute), 23 position (bstem.device.Servo attribute), 24 position (bstem.device.SmartServo attribute), 25  RGB (bstem.image_wrapper.ImageWrapper attribute), 29  RGB (bstem.image_wrapper.ImageWrapper attribute), 29	-		
position (bstem.device.Servo attribute), 24 position (bstem.device.SmartServo attribute), 25 RGB (bstem.image_wrapper.ImageWrapper attribute), 29	_		
position (bstem.device.SmartServo attribute), 25 RGB (bstem.image_wrapper.ImageWrapper attribute), 29	_		
	-		
	-		(ostem.mage_wrapper.magewrapper aurioute), 29

run() (bstem.tutorials.example_10_tracking.TrackerApp method), 8	stop() (bstem.disparity_hardware.DisparityHardware method), 26
run_daemon() (in module bstem.battery_check), 12	stop() (bstem.hokuyo.HokuyoDriver method), 28
S	stop() (bstem.video_encoder.VideoEncoder method), 43 streaming (bstem.video.Video attribute), 42
saturation (bstem.camera.Camera attribute), 14	subscribe() (bstem.log.LogServer method), 32
Scheduler (class in bstem.control), 16	suspend() (bstem.control.Scheduler class method), 16
secondary_status (bstem.device.PowerGauge attribute), 22	T
Sensor (class in bstem.sensor), 41 sensor_info() (bstem.hokuyo.HokuyoDriver method), 28 sensor_perens() (bstem.hokuyo.HokuyoDriver method)	temperature (bstem.device.PowerGauge attribute), 22 test_color_conversions() (in module bstem.test.test_image_wrapper), 6
sensor_params() (bstem.hokuyo.HokuyoDriver method), 28	test_crop() (in module bstem.test_image_wrapper), 6
Servo (class in bstem.device), 24	test_histogram() (bstem.test.test_plot.TestPlot method), 6
SERVO_DS (bstem.platform.ad_cord.AdCord attribute), 2	test_histogram_remote() (bstem.test.test_plot.TestPlot method), 6
SERVO_DS (bstem.platform.rc_cord.RcCord attribute),	test_histogram_remote_passive()
5	(bstem.test_plot.TestPlot method), 7
set() (bstem.log.VariableClient method), 34	test_lineplot() (bstem.test.test_plot.TestPlot method), 7
set_filename() (bstem.video_encoder.VideoEncoder method), 43	test_lineplot_remote() (bstem.test.test_plot.TestPlot method), 7
set_position_minmax() (bstem.device.Servo method), 24	test_lineplot_remote_passive()
setdefault() (bstem.log.LogServer method), 32	(bstem.test_plot.TestPlot method), 7
setup() (bstem.video.Video method), 42	test_multiple_control_loops()
sfx (bstem.camera.Camera attribute), 14	(bstem.test.test_control.TestControlLoop
SharpInfrared2D120X (class in bstem.infrared), 30	method), 6
SharpInfrared2Y0A21 (class in bstem.infrared), 30	test_reshape_and_types() (in module
sharpness (bstem.camera.Camera attribute), 14	bstem.test_image_wrapper), 6
show_disparity() (in module	test_single_control_loop()
bstem.tutorials.example_11_disparity), 8	$(bstem.test.test\_control.TestControlLoop\\$
show_video() (in module	method), 6
bstem.tutorials.example_3c_camera_config), 9	test_tracker() (bstem.test.test_tracker.TestTracker
shutdown() (bstem.video.Video method), 42	method), 7
simple_fxn() (in module bstem.test.test_plot), 7	test_version() (bstem.test.test_bstem.TestBstem method),
sin_tuple_fxn() (in module bstem.test.test_plot), 7	6
singletonmethod() (in module bstem.decorator), 16	test_YUV_forms() (in module
SmartServo (class in bstem.device), 25	bstem.test.test_image_wrapper), 6
snapshot() (bstem.camera.Camera method), 14	TestBstem (class in bstem.test.test_bstem), 6
software_disparity_example() (in module	TestControlLoop (class in bstem.test.test_control), 6
bstem.tutorials.example_11_disparity), 8	TestPlot (class in bstem.test_plot), 6
speed (bstem.device.Motor attribute), 20	TestTracker (class in bstem.test.test_tracker), 7
speed (bstem.device.SmartServo attribute), 25	timed_loop_example() (in module bstem.tutorials.example_4_scheduler), 9
SPI_1 (bstem.fpga.Fpga attribute), 27 SPI_10 (bstem.fpga.Fpga attribute), 27	timestamp() (bstem.sensor.Sensor method), 41
spi_send() (bstem.fpga.Fpga method), 27	to_temp_range() (bstem.device.PowerGauge static
spi_send_mult() (bstem.fpga.Fpga method), 27	method), 22
start() (bstem.control.Scheduler class method), 16	to_voltage() (bstem.device.PowerGauge static method),
start() (bstem.disparity_hardware.DisparityHardware	22
method), 26	TrackerApp (class in bstem.tutorials.example_10_tracking)
start() (bstem.hokuyo.HokuyoDriver method), 28	8
start() (bstem.motion_hardware.MotionHardware	
method), 35	U
start() (bstem.video_encoder.VideoEncoder method), 43	unsubscribe() (bstem.log.LogServer method), 32
start_recording() (bstem.camera.Camera method), 14	update() (bstem.control.ControlLoop method), 16

```
Y
update() (bstem.log.LogServer method), 32
update() (bstem.tracker.CAMShiftTracker method), 42
                                                         Y (bstem.image_wrapper.ImageWrapper attribute), 29
UV (bstem.image wrapper.ImageWrapper attribute), 29
                                                         y (bstem.sensor.Accelerometer attribute), 38
                                                         y (bstem.sensor.Gyroscope attribute), 40
V
                                                         y (bstem.sensor.Magnetometer attribute), 40
validate crop()
                                               module
                             (in
                                                         YUV (bstem.image wrapper.ImageWrapper attribute),
         bstem.test.test_image_wrapper), 6
value (bstem.device.Gpio attribute), 19
                                                         Z
value (bstem.sensor.Accelerometer attribute), 38
value (bstem.sensor.Barometer attribute), 39
                                                         z (bstem.sensor.Accelerometer attribute), 38
value (bstem.sensor.Gyroscope attribute), 39
                                                         z (bstem.sensor.Gyroscope attribute), 40
value (bstem.sensor.Magnetometer attribute), 40
                                                         z (bstem.sensor.Magnetometer attribute), 40
value() (bstem.device.AdcAdCord method), 17
value() (bstem.device.AdcLgCord method), 18
value() (bstem.device.AdcRcCord method), 18
value() (bstem.sensor.Sensor method), 41
value ts() (bstem.sensor.Accelerometer method), 38
value ts() (bstem.sensor.Gyroscope method), 39
values() (bstem.log.LogServer method), 32
variable_server_example()
                                               module
         bstem.tutorials.example_7_parameter_tuning),
VariableClient (class in bstem.log), 34
VariableServer (class in bstem.log), 34
velocity (bstem.device.Encoder attribute), 19
velocity (bstem.device.QuadEncoder attribute), 23
velocity (bstem.device.SmartServo attribute), 25
version() (bstem.hokuyo.HokuyoDriver method), 28
Video (class in bstem.video), 42
video_encoder_speedup_hack()
                                     (in
                                               module
         bstem.motion_hardware), 35
video_recording_example()
                                               module
         bstem.tutorials.example_3b_video_recording),
VideoEncoder (class in bstem.video encoder), 43
viewitems() (bstem.log.LogServer method), 32
viewkeys() (bstem.log.LogServer method), 32
viewvalues() (bstem.log.LogServer method), 32
voltage (bstem.device.PowerGauge attribute), 23
W
waitKey_thread() (bstem.display_thread.DisplayThread
         method), 27
width (bstem.image_wrapper.ImageWrapper attribute),
write_images() (bstem.camera.Camera method), 15
writeToCsv() (bstem.log.Logger method), 33
writeToCsv() (bstem.log.LogServer method), 32
writeToServer() (bstem.log.Logger method), 33
X
x (bstem.sensor.Accelerometer attribute), 38
x (bstem.sensor.Gyroscope attribute), 39
x (bstem.sensor.Magnetometer attribute), 40
```