More about Motors

The Control Class

The `Motor` class uses PID control to accurately track your commanded target angles. Similarly, the `DriveBase` class uses two of such controllers: one to control the heading and one to control the traveled distance.

You can change the control settings through the following attributes, which are instances of the `Control` class given below.:

- `Motor.control`
- `DriveBase.heading_control`
- `DriveBase.distance_control`

You can only change the settings while the controller is stopped. For example, you can set the settings at the beginning of your program. Alternatively, first call `stop()` to make your `Motor` or `DriveBase` stop, and then change the settings.

class **Control**

Class to interact with PID controller and settings.

scale

Scaling factor between the controlled integer variable and the physical output. For example, for a single motor this is the number of encoder pulses per degree of rotation.

Status

done()

Checks if an ongoing command or maneuver is done.

Returns: `True` if the command is done, `False` if not.

Return type: bool

stalled()

Checks if the controller is currently stalled.

A controller is stalled when it cannot reach the target speed or position, even with the maximum actuation signal.

Returns: `True` if the controller is stalled, `False` if not.

Return type: bool

Settings

`limits`(*speed*, *acceleration*, *actuation*)

Configures the maximum speed, acceleration, and actuation.

If no arguments are given, this will return the current values.

Parameters:

- **speed** (rotational speed: deg/s or speed: mm/s) – Maximum speed. All speed commands will be capped to this value.
- **acceleration** (rotational acceleration: deg/s/s or linear acceleration: mm/s/s) – Maximum acceleration.
- **actuation** (percentage: %) – Maximum actuation as percentage of absolute maximum.

`pid`(*kp*, *ki*, *kd*, *integral_range*, *integral_rate*, *feed_forward*)

Gets or sets the PID values for position and speed control.

If no arguments are given, this will return the current values.

Parameters:

- **kp** (*int*) – Proportional position (or integral speed) control constant.
- **ki** (*int*) – Integral position control constant.
- **kd** (*int*) – Derivative position (or proportional speed) control constant.
- **integral_range** (angle: deg or distance: mm) – Region around the target angle or distance, in which integral control errors are accumulated.
- **integral_rate** (rotational speed: deg/s or speed: mm/s) – Maximum rate at which the error integral is allowed to grow.
- **feed_forward** (percentage: %) – This adds a feed forward signal to the PID feedback signal, in the direction of the speed reference. This value is expressed as a percentage of the absolute maximum duty cycle.

`target_tolerances`(*speed*, *position*)

Gets or sets the tolerances that say when a maneuver is done.

If no arguments are given, this will return the current values.

Parameters:

- **speed** (rotational speed: deg/s or speed: mm/s) – Allowed deviation from zero speed before motion is considered complete.
- **position** (angle: deg or distance: mm) – Allowed deviation from the target before motion is considered complete.

`stall_tolerances`(*speed*, *time*)

Gets or sets stalling tolerances.

If no arguments are given, this will return the current values.

Parameters:

- **speed** (rotational speed: deg/s or speed: mm/s) – If the controller cannot reach this speed for some `time` even with maximum actuation, it is stalled.
- **time** (time: ms) – How long the controller has to be below this minimum `speed` before we say it is stalled.