Helicopter Rig Project

Generated by Doxygen 1.8.13

Contents

| 1 Module Index | | | 2 |
|----------------|-------------|--------------------------------------|----|
| | 1.1 | Modules | 2 |
| 2 | Data | a Structure Index | 2 |
| | 2.1 | Data Structures | 2 |
| 3 | Mod | dule Documentation | 2 |
| | 3.1 | ButtonsApi | 2 |
| | | 3.1.1 Detailed Description | 3 |
| | | 3.1.2 Enumeration Type Documentation | 3 |
| | | 3.1.3 Function Documentation | 3 |
| | 3.2 | FlightController | 5 |
| | | 3.2.1 Detailed Description | 5 |
| | | 3.2.2 Function Documentation | 5 |
| | 3.3 | HeightApi | 7 |
| | | 3.3.1 Detailed Description | 7 |
| | | 3.3.2 Function Documentation | 7 |
| | 3.4 | HeightController | 8 |
| | | 3.4.1 Detailed Description | 8 |
| | | 3.4.2 Function Documentation | 8 |
| | 3.5 | PidController | 11 |
| | | 3.5.1 Detailed Description | 11 |
| | | 3.5.2 Function Documentation | 11 |
| | 3.6 | PwmOutput | 13 |
| | | 3.6.1 Detailed Description | 13 |
| | | 3.6.2 Enumeration Type Documentation | 13 |
| | | 3.6.3 Function Documentation | 13 |
| | 3.7 | SwitchApi | 16 |
| | | 3.7.1 Detailed Description | 16 |
| | | 3.7.2 Enumeration Type Documentation | 16 |
| | | 3.7.3 Function Documentation | 16 |
| | 3.8 | YawApi | 18 |
| | | 3.8.1 Detailed Description | 18 |
| | | 3.8.2 Function Documentation | 18 |
| | 3.9 | YawController | 20 |
| | J. J | 3.9.1 Detailed Description | 20 |
| | | 3.9.2 Function Documentation | 20 |
| | | College Carolina Doddination | _0 |

| 4 Data Structure Documentation | 2 | 23 |
|---|--------------|------------|
| 4.1 PidState Struct Reference | 2 | 23 |
| 4.1.1 Detailed Description | 2 | 23 |
| Index | 2 | <u>?</u> 5 |
| | | |
| 1 Module Index | | |
| 1.1 Modules | | |
| Here is a list of all modules: | | |
| ButtonsApi | | 2 |
| FlightController | | 5 |
| HeightApi | | 7 |
| PidController | 1 | 1 |
| HeightController | | 8 |
| YawController | 2 | 20 |
| PwmOutput | 1 | 3 |
| SwitchApi | 1 | 6 |
| YawApi | 1 | 8 |
| 2 Data Structure Index | | |
| 2.1 Data Structures | | |
| Here are the data structures with brief descriptions: | | |
| PidState A structure to accumulate the error and store the previous error for use by the pid c | controller 2 | !3 |
| 3 Module Documentation | | |
| 3.1 ButtonsApi | | |
| A module to operate the buttons. | | |

3.1 ButtonsApi 3

Enumerations

```
enum Button {
    BTN_UP, BTN_DOWN, BTN_LEFT, BTN_RIGHT,
    NUM_BUTTONS }
```

The buttons.

Functions

• void ButtonsInit (void)

Initialise the buttons.

• void UpdateButtons ()

Update all of the buttons and their state.

• uint8_t NumPushes (uint8_t button_name)

Gets the number of pushes for a given button and resets the push count.

void ResetPushes (void)

Reset the push count for all buttons.

3.1.1 Detailed Description

A module to operate the buttons.

3.1.2 Enumeration Type Documentation

3.1.2.1 Button

```
enum Button
```

The buttons.

Enumerator

| BTN_UP | The UP button. |
|-------------|------------------------------|
| BTN_DOWN | The DOWN button. |
| BTN_LEFT | The LEFT button. |
| BTN_RIGHT | The RIGHT button. |
| NUM_BUTTONS | The total number of buttons. |

3.1.3 Function Documentation

3.1.3.1 NumPushes()

Gets the number of pushes for a given button and resets the push count.

Parameters

| button_name | One of BTN_UP, BTN_DOWN, BTN_LEFT, or BTN_RIGHT. |
|-------------|--|
|-------------|--|

Returns

The number of pushes for the given button since this function was last called.

3.2 FlightController 5

3.2 FlightController

Module containing functions to manager switches between flight states.

Functions

· void FlightControllerInit (void)

Initialise the flight controller module.

void UpdateFlightMode ()

A task to be called periodically that manages switches between flight states.

const char * GetFlightMode (void)

Get a string representation of the current flight mode.

void PriorityTaskInit (void)

Initialise the priority task sequencer.

void PriorityTaskEnable (void)

Enable the priority task sequencer.

void PriorityTaskDisable (void)

Disable the priority task sequencer.

3.2.1 Detailed Description

Module containing functions to manager switches between flight states.

Additionally the module contains a timer, which has been set up to run critical tasks. These include the pid controllers and the updating of the height sensor.

3.2.2 Function Documentation

3.2.2.1 GetFlightMode()

Get a string representation of the current flight mode.

Returns

a string representing the flight mode

3.2.2.2 PriorityTaskInit()

```
void PriorityTaskInit (
     void )
```

Initialise the priority task sequencer.

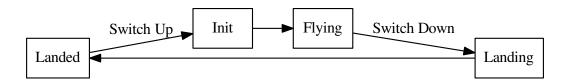
This timer handles updating of the pid controllers and height.

3.2.2.3 UpdateFlightMode()

```
void UpdateFlightMode ( )
```

A task to be called periodically that manages switches between flight states.

Below is a state machine representing the transistions between flight states.



3.3 HeightApi 7

3.3 HeightApi

Module to acquire current height via the height sensor and trigger a zero-height reading.

Functions

• int32_t GetHeight (void)

Get the current height.

int32_t GetHeightPercentage (void)

Get the current height as a percentage.

void UpdateHeight ()

Trigger the current height reading to be updated.

void HeightManagerInit (void)

Initialise the height sensor peripherals and ports.

void ZeroHeightTrigger (void)

Trigger a zero height reading to be used as a reference for subsequent height readings.

3.3.1 Detailed Description

Module to acquire current height via the height sensor and trigger a zero-height reading.

Provides helper methods to get the current height.

3.3.2 Function Documentation

3.3.2.1 GetHeight()

```
int32_t GetHeight (
     void )
```

Get the current height.

Retrieve the sensor reading after it has been offset the zeroed sensor reading.

Returns

The height.

3.3.2.2 GetHeightPercentage()

Get the current height as a percentage.

Returns

The height as a percentage.

3.3.2.3 UpdateHeight()

```
void UpdateHeight ( )
```

Trigger the current height reading to be updated.

No longer used in favour of direct triggering via timer timeout flag.

3.4 HeightController

Pid controller for the main rotor.

Collaboration diagram for HeightController:



Functions

void SetTargetHeight (uint32_t height)

Set the target height (%).

uint32_t GetTargetHeight (void)

Get the target height (%).

• void HeightControllerInit (void)

Initialise the height controller.

void PreloadHeightController (int32_t control, int32_t error)

Preload the integral component of the pid contoller so the Main rotor starts of with control power.

void UpdateHeightController (uint32_t delta_t)

Update the height controller pid loop.

• void TuneProportionalMainRotor (double gain)

Use at own risk.

3.4.1 Detailed Description

Pid controller for the main rotor.

3.4.2 Function Documentation

3.4.2.1 GetTargetHeight()

Get the target height (%).

Returns

The target height(%).

3.4 HeightController 9

3.4.2.2 PreloadHeightController()

 $Preload \ the \ integral \ component \ of \ the \ pid \ contoller \ so \ the \ Main \ rotor \ starts \ of \ with \ \verb|control| \ power.$

This was to improves rise time of the helicopter by boosting the Main rotor.

Parameters

| control | The immediate control power desired by the Main rotor |
|---------|---|
| error | The absolute difference between current height and target height. |

3.4.2.3 SetTargetHeight()

Set the target height (%).

Parameters

| he target height (%). | height |
|-----------------------|--------|
|-----------------------|--------|

3.4.2.4 TuneProportionalMainRotor()

```
\begin{tabular}{ll} {\tt void TuneProportionalMainRotor (} \\ & {\tt double } \end{tabular} \label{table pain ()} \\ \end{tabular}
```

Use at own risk.

Parameters

| gain | Proportial gain. |
|------|--------------------|
| 94 | i roportiai gaiiii |

3.4.2.5 UpdateHeightController()

```
void UpdateHeightController ( \mbox{uint32\_t} \ \ \mbox{delta\_t} \ )
```

Update the height controller pid loop.

Parameters

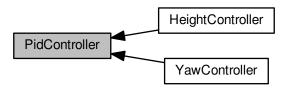
| delta⊷ | The update period of the height controller. |
|--------|---|
| _t | |

3.5 PidController 11

3.5 PidController

Generic pid controller module.

Collaboration diagram for PidController:



Modules

HeightController

Pid controller for the main rotor.

YawController

Pid controller for the tail rotor.

Data Structures

struct PidState

A structure to accumulate the error and store the previous error for use by the pid controller.

Functions

• void PidInit (PidState *state)

Initialise the pid controller.

void PreloadPid (PidState *state, int32_t integral_preload)

Preload the integral component of the pid state with a postitve or negative error to reduce integration time.

int32_t UpdatePid (PidState *state, int32_t error, uint32_t delta_t, double proportional_gain, double integral
 — gain, double derivative_gain)

Update the pid controller loop.

3.5.1 Detailed Description

Generic pid controller module.

3.5.2 Function Documentation

3.5.2.1 PidInit()

Initialise the pid controller.

Parameters

| state The pid e | rror state. |
|-----------------|-------------|
|-----------------|-------------|

See also

PidState

3.5.2.2 PreloadPid()

Preload the integral component of the pid state with a postitve or negative error to reduce integration time.

Parameters

| state | The pid error state. |
|------------------|----------------------|
| integral_preload | The preload error. |

3.5.2.3 UpdatePid()

Update the pid controller loop.

Parameters

| state | The pid error state. |
|-------------------|--|
| error | The current error. |
| delta_t | The update period of the pid controller. |
| proportional_gain | The proportional gain constant. |
| integral_gain | The integral gain constant. |
| derivative_gain | The derivative gain constant. |

Returns

3.6 PwmOutput 13

3.6 PwmOutput

PWM module to handle the power output for the main and tail rotors.

Enumerations

enum PwmOutput { MAIN_ROTOR, TAIL_ROTOR }

An enumeration for determining which PWM output to configure.

Functions

· void PwmInit ()

Initialise both of the pwm outputs.

void SetPwmDutyCycle (uint8_t pwm_output, uint32_t duty_cycle)

Set the duty cycle of the PWM output in the range 2-98%.

uint32_t GetPwmDutyCycle (uint8_t pwm_output)

Get the current PWM duty cycle for the given PWM output.

void PwmDisable (uint8_t pwm_output)

Disable the given PWM output.

void PwmEnable (uint8_t pwm_output)

Enable the given PWM output.

3.6.1 Detailed Description

PWM module to handle the power output for the main and tail rotors.

3.6.2 Enumeration Type Documentation

3.6.2.1 PwmOutput

enum PwmOutput

An enumeration for determining which PWM output to configure.

Enumerator

| MAIN_ROTOR | The main rotor PWM output. |
|------------|----------------------------|
| TAIL_ROTOR | The tail rotor PWM output. |

3.6.3 Function Documentation

3.6.3.1 GetPwmDutyCycle()

Get the current PWM duty cycle for the given PWM output.

Parameters

```
pwm_output The PWM output.
```

Returns

The current PWM duty cycle as percentage.

3.6.3.2 PwmDisable()

Disable the given PWM output.

Parameters

|--|

3.6.3.3 PwmEnable()

Enable the given PWM output.

Parameters

| pwm_output | The PWM output to configure. |
|------------|------------------------------|

3.6.3.4 SetPwmDutyCycle()

Set the duty cycle of the PWM output in the range 2-98%.

3.6 PwmOutput 15

Parameters

| pwm_output | The PWM output to configure. | |
|------------|---|--|
| duty_cycle | The desired duty cycle, in the range 2-98%. | |

3.7 SwitchApi

A module to operate the mode switch.

Enumerations

enum SwitchState { SWITCH_DOWN, SWITCH_UP }

An enumeration of the states of the slider switch.

Functions

void SwitchInit (void)

Initialise the switch.

· void UpdateSwitch ()

Update the switch state.

uint8_t GetSwitchEvent (void)

Get the lastest switch event.

3.7.1 Detailed Description

A module to operate the mode switch.

3.7.2 Enumeration Type Documentation

3.7.2.1 SwitchState

```
enum SwitchState
```

An enumeration of the states of the slider switch.

Enumerator

| SWITCH_DOWN | The switch is in the downwards position. |
|-------------|--|
| SWITCH_UP | The switch is in the upwards position. |

3.7.3 Function Documentation

3.7.3.1 GetSwitchEvent()

Get the lastest switch event.

3.7 SwitchApi

DOWN or UP slide of the switch.

3.8 YawApi

Functions

void YawDetectionInit (void)

Initialises the yaw manager.

int32_t GetYaw (void)

Get the current yaw.

• int32_t GetYawDegrees (void)

Get the current yaw.

int32_t GetClosestYawRef (int32_t current_yaw)

Helper function to return the closest yaw such that the helicopter is facing towards the camera.

void YawRefTrigger (void)

Triggers an interrupt to fire when the refernce yaw has been found.

· bool YawRefFound (void)

Check if the reference yaw has been found.

- 3.8.1 Detailed Description
- 3.8.2 Function Documentation

3.8.2.1 GetClosestYawRef()

Helper function to return the closest yaw such that the helicopter is facing towards the camera.

Parameters

```
current_yaw the current yaw
```

Returns

the closest reference yaw

3.8.2.2 GetYaw()

```
int32_t GetYaw (
     void )
```

Get the current yaw.

Returns

the yaw (notches)

3.8 YawApi 19

3.8.2.3 GetYawDegrees()

```
int32_t GetYawDegrees ( void )
```

Get the current yaw.

Returns

the yaw (degrees)

3.8.2.4 YawRefFound()

```
bool YawRefFound (
     void )
```

Check if the reference yaw has been found.

Returns

true if the yaw reference has been found else false

3.9 YawController

Pid controller for the tail rotor.

Collaboration diagram for YawController:



Functions

• int32_t GetTargetYawDegrees (void)

Get the target yaw in degrees.

void SetTargetYawDegrees (int32_t yaw)

Set the desired target yaw (degrees).

int32_t GetTargetYaw (void)

Get the target yaw.

void SetTargetYaw (int32_t yaw)

Set the desired target yaw.

void YawControllerInit (void)

Initialise the yaw controller.

• void PreloadYawController (int32_t control, int32_t error)

Preload the integral component of the pid contoller so the Tail rotor starts of with control power.

void UpdateYawController (uint32_t delta_t)

Update the yaw controller pid loop.

• void TuneProportionalTailRotor (double gain)

Use at own risk.

3.9.1 Detailed Description

Pid controller for the tail rotor.

3.9.2 Function Documentation

3.9.2.1 GetTargetYaw()

```
int32_t GetTargetYaw (
     void )
```

Get the target yaw.

Returns

the target yaw.

3.9 YawController 21

3.9.2.2 GetTargetYawDegrees()

Get the target yaw in degrees.

Returns

The target yaw in degrees.

3.9.2.3 PreloadYawController()

Preload the integral component of the pid contoller so the Tail rotor starts of with control power.

Can be used in conjuction with PreloadHeightController to improve the rise time of the helicopter.

Parameters

| control | The immediate control power desired by the Tail rotor |
|---------|---|
| error | The absolute difference between current yaw and target yaw. |

3.9.2.4 SetTargetYaw()

Set the desired target yaw.

Parameters

| yaw | The desired target yaw. |
|-----|-------------------------|
| yaw | The desired target yaw. |

3.9.2.5 SetTargetYawDegrees()

Set the desired target yaw (degrees).

Parameters

| yaw | The desired target yaw (degrees). |
|-----|-----------------------------------|
|-----|-----------------------------------|

3.9.2.6 TuneProportionalTailRotor()

```
void TuneProportionalTailRotor ( \mbox{double $\it gain $\it )}
```

Use at own risk.

Parameters

```
gain Proportial gain.
```

3.9.2.7 UpdateYawController()

```
void UpdateYawController ( \mbox{uint32\_t} \ \ delta\_t \ )
```

Update the yaw controller pid loop.

Parameters

| delta⊷ | The update period of the yaw controller. |
|--------|--|
| t | |

4 Data Structure Documentation

4.1 PidState Struct Reference

A structure to accumulate the error and store the previous error for use by the pid controller.

```
#include <pid.h>
```

Data Fields

• int32_t error_previous

The previous error.

• int32_t error_integrated

The accumulated error.

4.1.1 Detailed Description

A structure to accumulate the error and store the previous error for use by the pid controller.

The documentation for this struct was generated from the following file:

• src/pid.h

Index

| Button | HeightController, 8 |
|-------------------------------|---------------------------|
| ButtonsApi, 3 | PreloadPid |
| ButtonsApi, 2 | PidController, 12 |
| Button, 3 | PreloadYawController |
| NumPushes, 3 | YawController, 21 |
| | PriorityTaskInit |
| FlightController, 5 | FlightController, 5 |
| GetFlightMode, 5 | PwmDisable |
| PriorityTaskInit, 5 | PwmOutput, 14 |
| UpdateFlightMode, 5 | PwmEnable |
| , | PwmOutput, 14 |
| GetClosestYawRef | PwmOutput, 13 |
| YawApi, 18 | GetPwmDutyCycle, 13 |
| GetFlightMode | PwmDisable, 14 |
| FlightController, 5 | PwmEnable, 14 |
| GetHeight | PwmOutput, 13 |
| HeightApi, 7 | SetPwmDutyCycle, 14 |
| GetHeightPercentage | octi wiiibatyoyolo, 14 |
| HeightApi, 7 | SetPwmDutyCycle |
| GetPwmDutyCycle | PwmOutput, 14 |
| PwmOutput, 13 | SetTargetHeight |
| GetSwitchEvent | HeightController, 10 |
| SwitchApi, 16 | SetTargetYaw |
| GetTargetHeight | YawController, 21 |
| HeightController, 8 | SetTargetYawDegrees |
| GetTargetYaw | YawController, 21 |
| YawController, 20 | SwitchApi, 16 |
| GetTargetYawDegrees | GetSwitchEvent, 16 |
| YawController, 20 | SwitchState, 16 |
| GetYaw | SwitchState, 10 |
| YawApi, 18 | SwitchApi, 16 |
| GetYawDegrees | Switchapi, 10 |
| YawApi, 18 | TuneProportionalMainRotor |
| Tawλρi, To | HeightController, 10 |
| HeightApi, 7 | TuneProportionalTailRotor |
| GetHeight, 7 | YawController, 22 |
| GetHeightPercentage, 7 | rawcontroller, ZZ |
| UpdateHeight, 7 | UpdateFlightMode |
| HeightController, 8 | FlightController, 5 |
| GetTargetHeight, 8 | UpdateHeight |
| PreloadHeightController, 8 | HeightApi, 7 |
| SetTargetHeight, 10 | UpdateHeightController |
| TuneProportionalMainRotor, 10 | HeightController, 10 |
| UpdateHeightController, 10 | UpdatePid |
| opuater leight controller, 10 | PidController, 12 |
| NumPushes | UpdateYawController |
| ButtonsApi, 3 | YawController, 22 |
| Buttonish (pi, o | rawoontroner, 22 |
| PidController, 11 | YawApi, 18 |
| PidInit, 11 | GetClosestYawRef, 18 |
| PreloadPid, 12 | GetYaw, 18 |
| UpdatePid, 12 | GetYawDegrees, 18 |
| PidInit | YawRefFound, 19 |
| PidController, 11 | YawController, 20 |
| PidState, 23 | GetTargetYaw, 20 |
| PreloadHeightController | GetTargetYawDegrees, 20 |
| 1 Total Total Total Total | Gottargottawbogroco, 20 |

26 INDEX

PreloadYawController, 21
SetTargetYaw, 21
SetTargetYawDegrees, 21
TuneProportionalTailRotor, 22
UpdateYawController, 22
YawRefFound
YawApi, 19