Deadalus

Generated by Doxygen 1.7.6.1

Tue May 13 2014 00:07:58

Contents

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

Class Documentation

2.1 ActuationValue Class Reference

Public Attributes

- int **motor** [4]

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

- /home/usappz/DeadalusSim/source/dependencies/resources.hh

2.2 AngularVel Class Reference

Public Member Functions

- AngularVel **get_acceleration** (AngularVel angular_vel_current, AngularVel angular_vel_last, double dT)

Public Attributes

- double **motor** [4]

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

- /home/usappz/DeadalusSim/source/dependencies/resources.hh

2.3 gazebo::GazeboIMUPlugin Class Reference

Gazebo Plugin which updates the subscribed messages of the sensors.

`#include <gazebo_imu_plugin.hh>`

Public Member Functions

- GazeboIMUPlugin ()

  Constructor.
- virtual ~GazeboIMUPlugin ()

  Destructor.
- virtual void Load (sensors::SensorPtr _sensor, sdf::ElementPtr _sdf)

  Load the sensor plugin.

Private Member Functions

- virtual void OnUpdate ()

  Callback that recieves the contact sensor's update signal.

Private Attributes

- sensors::ImuSensorPtr parentSensor

  Pointer to the contact sensor.
- event::ConnectionPtr updateConnection

  Connection that maintains a link between the contact sensor's updated signal and the OnUpdate callback.

2.3.1 Detailed Description

Gazebo Plugin which updates the subscribed messages of the sensors.

2.3.2 Member Function Documentation

2.3.2.1 void GazeboIMUPlugin::Load (sensors::SensorPtr *_sensor*, sdf::ElementPtr *_sdf*) [virtual]

Load the sensor plugin.

Parameters

in	*_sensor*	Pointer to the sensor that loaded this plugin.
in	*_sdf*	SDF element that describes the plugin.

The documentation for this class was generated from the following files:

- /home/usappz/DeadalusSim/source/gazebo_plugins/gazebo_imu_plugin.hh
- /home/usappz/DeadalusSim/source/gazebo_plugins/gazebo_imu_plugin.cc

2.4 gazebo::GazeboModelPlugin Class Reference

Gazebo Plugins which contains the main update function of the simulation.

`#include <gazebo_model_plugin.hh>`

Public Member Functions

- void **Load** (physics::ModelPtr _parent, sdf::ElementPtr)
- void callback_imu (ConstIMUPtr &msg)

  Function is called everytime an imu-message is received.
- void OnUpdate (const common::UpdateInfo &)

  Called by the world update start event.

Public Attributes

- QuadRotorDynamics **_myQuadRotorDynamics**
- AngularVel **_rotor_speed**
- Mechanics **_mechanics**

Private Types

- enum **MotorNumber** { **m1**, **m2**, **m3**, **m4**, **front**, **rear**, **right**, **left**, **all** }

Private Attributes

- transport::NodePtr **node**
- transport::SubscriberPtr **imuSubscriber**
- SIL **_sil**
- Sensors **_sensor_vals**
- physics::ModelPtr model

  Pointer to the model.
- physics::LinkPtr motor_1

  Pointer to the motor links.
- physics::LinkPtr **motor_front**
- physics::LinkPtr **motor_2**
- physics::LinkPtr **motor_right**
- physics::LinkPtr **motor_3**
- physics::LinkPtr **motor_rear**
- physics::LinkPtr **motor_4**
- physics::LinkPtr **motor_left**
- physics::LinkPtr frame_center

  Pointer to the center Link.
- physics::LinkPtr motor_prop_1

Pointer to the motor rod links.

- physics::LinkPtr **motor_prop_front**
- physics::LinkPtr **motor_prop_2**
- physics::LinkPtr **motor_prop_right**
- physics::LinkPtr **motor_prop_3**
- physics::LinkPtr **motor_prop_rear**
- physics::LinkPtr **motor_prop_4**
- physics::LinkPtr **motor_prop_left**
- event::ConnectionPtr updateConnection

  Pointer to the update event connection.
- common::Time **_current_time**
- common::Time **_last_time**
- common::Time **_dt**

2.4.1 Detailed Description

Gazebo Plugins which contains the main update function of the simulation.

This is the Gazebo Plugin which updates the simulation. It applies the Thrust forces and Yaw Torque to the frame, subscribes the sensor values and hands these and the simulation time, which has passed since the last step to the software in the loop (SIL)

2.4.2 Member Function Documentation

2.4.2.1 void GazeboModelPlugin::callback_imu (ConstIMUPtr & *msg*)

Function is called everytime an imu-message is received.

The function

2.4.2.2 void GazeboModelPlugin::OnUpdate (const common::UpdateInfo &)

Called by the world update start event.

Every time the simulation is updated the sofware in the loop is updated by the current sensor values and the time which has passed since the last step. Further morethe desired actuation values are aquired by the software in the software in the loop (SIL) instance. These are handed to the instance of QuadRotorDynamics. This calculates the corresponding rotor speeds by its pt1_element function. After checking whether these do exceed the maximal angular velocity and the maximal angular acceleleration of the rotors by check_limits(), it calculates the Thrust forces and Yaw Torque. Now the Thrust forces and Yaw Torque can be applied to the frame.

The documentation for this class was generated from the following files:

- /home/usappz/DeadalusSim/source/gazebo_plugins/gazebo_model_plugin.hh
- /home/usappz/DeadalusSim/source/gazebo_plugins/gazebo_model_plugin.cc

2.5 GPS Class Reference

Public Attributes

- double **altitude**
- double **latitude_deg**
- double **longitude_deg**

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

- /home/usappz/DeadalusSim/source/dependencies/resources.hh

2.6 IMU Class Reference

Public Member Functions

- void **set_acc** (double a_x, double a_y, double a_z)
- void **set_acc** (gazebo::msgs::Vector3d acc)
- void **set_gyro** (double g_x, double g_y, double g_z)
- void **set_gyro** (gazebo::msgs::Vector3d gyro)
- void **set** (gazebo::msgs::Vector3d acc, gazebo::msgs::Vector3d gyro)

Public Attributes

- double **acc_x**
- double **acc_y**
- double **acc_z**
- double **gyro_x**
- double **gyro_y**
- double **gyro_z**

Private Attributes

- gazebo::msgs::Vector3d **acc_vec3**
- gazebo::msgs::Vector3d **gyro_vec3**

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

- /home/usappz/DeadalusSim/source/dependencies/resources.hh

2.7 MAG Class Reference

Public Member Functions

- void **set** (double m_x, double m_y, double m_z)
- void **set** (gazebo::msgs::Vector3d mag)

Public Attributes

- double **mag_x**
- double **mag_y**
- double **mag_z**

Private Attributes

- gazebo::msgs::Vector3d **mag_vec3**

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

- /home/usappz/DeadalusSim/source/dependencies/resources.hh

2.8 Mechanics Class Reference

Public Attributes

- Thrust **thrust**
- Torque **torque**

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

- /home/usappz/DeadalusSim/source/dependencies/resources.hh

2.9 QuadRotorDynamics Class Reference

Class for the calculation of the Quadrotor Dynamics.

`#include <quadrotor_dynamics.hh>`

Public Member Functions

- QuadRotorDynamics (double k_t=1.0, double k_m=1.0, double moment_arm=0.-025, double K=1.0, double C=0.0, double T=0.0, double dead_time=0.0)

  Constructor. constructor with default values.

- virtual ~QuadRotorDynamics ()

  Destructor.
- Mechanics update (ActuationValue actuation_value, double deltaT)

  Updates the Thrusts and Torques.
- void activate_limit_constant (bool active)

  Activate check_limits() for maximal rotor speed.
- void activate_limit_transient (bool active)

  Activate check_limits() for maximal rotor acceleration.
- void set_limit_const (double rotor_speed_max)

  Set maximal rotor speed for all motors for check_limits()
- void set_limit_transient (double rotor_acceleration_max)

  Set maximal rotor acceleration for all motors for check_limits()
- void set_k_pt1 (double K)

  Set functions for pt1 element.
- void set_t_pt1 (double T)

  Set function for time constant of pt1_element()
- void set_c_pt1 (double C)

  Set function for constant of pt1_element()
- void set_dead_time_pt1 (double dead_time)

  Set function for dead time of pt1_element()
- Mechanics getMechanics ()
- Thrust getThrust ()
- Torque getTorque ()
- void setErrorType (int error_type)

  Set error type for disturbance()

Private Member Functions

- void check_limits ()

  checks whether angular velocity and angular acceleration exceed the corresponding maximum values
- void pt1_element ()

  First order differential equation for desired angular velocities.
- void aerodynamics ()

  Thrust and Drag equation calculated by the momentum theory.
- void roll_pitch_torque ()

  Equations for Roll and Pitch Torque resulting from Thrusts.
- void disturbance ()

  Adds Error value to the Thrusts.

Private Attributes

- double _error_index

  Needed by disturbance(), for error calculation.
- AngularVel _rotor_speed

  Current rotor speeds.
- AngularVel _rotor_speed_prev

  Previous rotor speeds.
- AngularVel _rotor_acceleration

  Current angular velocity of the rotors.
- ActuationValue _actuation_value
- Mechanics _mechanics

  Mechanics containing Thrust and Torque on the frame.
- double _dT
- double _k_t

  Aerodynamics Constants.
- double **_k_m**
- double _moment_arm

  Distance between motor axis and center of frame.
- int **_error_type**
- double _K

  Proportional part of pt1_element()
- double _C

  Added Constant of pt1_element()
- double _T

  Damping time constant of pt1_element()
- double _dead_time

  Dead Time of pt1_element() (not supported yet)
- double _rotor_speed_max

  Maximal rotor speed for check_limits()
- double _rotor_acceleration_max

  Maximal acceleratin for ceck_limits()
- bool _limit_const_act

  Activate check_limits() for maximal angular velocity for motors.
- bool _limit_transient_act

  Activate check_limits() for maximal antular acceleration of motors.

2.9.1 Detailed Description

Class for the calculation of the Quadrotor Dynamics.

This class calculates the Torques and Forces on the quadrotor. Since gazebo does not support aerodynamics, this class calculates the Thrusts and the Yaw Torques resulting from the Rotor Speeds.

2.9.2 Member Function Documentation

2.9.2.1 void QuadRotorDynamics::aerodynamics ( ) [private]

Thrust and Drag equation calculated by the momentum theory.

The Thrusts and Draq torques are calculated by momentum theory. The measurements from qihich the constants are derived are found in scripts/measurements of the DeadalusSim directory. The scripts fitting the curves are found in the DeadalusSim scripts.

2.9.2.2 void QuadRotorDynamics::check_limits ( ) [private]

checks whether angular velocity and angular acceleration exceed the corresponding maximum values

Limit for maximum angular velocity is set by set_limit_const() and limits for angular acceleration are set by set_limit_transient() if these maximum values are activated by the corresponding functions activate_limit_cons(bool) and activate_limit_transient, the angular velocities are set to the maximum vlaues if thy exceed them.

2.9.2.3 void QuadRotorDynamics::disturbance ( ) [private]

Adds Error value to the Thrusts.

Since in reality the Thrusts of the motors are considered to be influenced by the environment a error value can be added. If an error type, different to NO_ERROR is set by SetErrorType an error is added to the calculated thrusts.

2.9.2.4 Mechanics QuadRotorDynamics::getMechanics ( )

Returns

Mechanics containing Thrust and Torque acting on the frame

2.9.2.5 Thrust QuadRotorDynamics::getThrust ( )

Returns

Thrust acting on the frame

2.9.2.6 Torque QuadRotorDynamics::getTorque ( )

Returns

Torque acting on the frame

2.9.2.7 void QuadRotorDynamics::pt1_element ( ) [private]

First order differential equation for desired angular velocities.

The motors are considered to be delayed by their inertia. Thus, corresponding to the actuation value set by the SIL the angular velocity is described by a first order differential equation. The input value of the PT1 element is the actuation value, the output value of the PT1 element is the desired angular velocity

2.9.2.8 void QuadRotorDynamics::roll_pitch_torque ( ) [private]

Equations for Roll and Pitch Torque resulting from Thrusts.

The Roll and Pitch Torques which are resulting from the Thrust forces and the the distance between the motors and the center of the frame (_moment_arm) are calculated here. The yaw torque is calculated in aerodynamics already, since it is derived from momentum theory.

2.9.2.9 void QuadRotorDynamics::set_k_pt1 (double *K*)

Set functions for pt1 element.

Set function for K of pt1_element()

2.9.2.10 void QuadRotorDynamics::setErrorType (int *error_type*)

Set error type for disturbance()

Parameters

error_type	error type of disturbance() (NO_ERROR/GAUSSIAN_ERROR/SINE_-ERROR)

2.9.2.11 Mechanics QuadRotorDynamics::update (ActuationValue *actuation_value*, double *deltaT*)

Updates the Thrusts and Torques.

This function updates the Mechanics containing the Thrust forces and Torque on the frame resulting from the current rotor speeds. It is intended to be called in every simulation step.

Parameters

actuation_-value	value which is set by the software in the loop (SIL)
deltaT	is the time which passed since the last simulation step

Returns

the updated Mechanics of the quadrocopter which includes the Thrust and Torque acting on the frame

2.9.3 Member Data Documentation

2.9.3.1 ActuationValue QuadRotorDynamics::_actuation_value [private]

Actuation values set by the control algorithms in the software in the loop (SIL) which correspond to desired rotor speeds.

2.9.3.2 double QuadRotorDynamics::_dT [private]

Simulation time passed since the last simulation step.

The documentation for this class was generated from the following files:

- /home/usappz/DeadalusSim/source/quadrotor_dynamics/quadrotor_dynamics.hh
- /home/usappz/DeadalusSim/source/quadrotor_dynamics/quadrotor_dynamics.cc

2.10 Sensors Class Reference

Public Attributes

- IMU **imu**
- MAG **mag**
- GPS **gps**

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

- /home/usappz/DeadalusSim/source/dependencies/resources.hh

2.11 SIL Class Reference

Class containing the software in the loop (SIL)

```
#include <sil.hh>
```

Public Member Functions

- SIL ()

  Constructor.

- virtual ~SIL ()

  Destructor.
- void test_sequence ()

  Test Sequence containing test stimuli for the simulation.
- ActuationValue get_actuation_value ()

  get functio for _actuation_value
- void update (double dT, Sensors mySensorsVals)

  update function of the software in the loop (SIL)

Private Attributes

- ActuationValue _actuation_value
- int _test_sequence_iterator

  iterator needed by the test_sequence()

2.11.1 Detailed Description

Class containing the software in the loop (SIL)

The member update of this class contains the software which is intended to be tested by the DeadalusSim

2.11.2 Member Function Documentation

2.11.2.1 void SIL::update (double *dT*, Sensors *mySensorsVals*)

update function of the software in the loop (SIL)

This function is intended to be called every iteration cycle of the simulation and should be placed in the update function of the gazebo plugin GazeboModelPugin().

Parameters

dT	time which passed since the last iteration step
all	current values of the Sensors containing acceleration and gyro (IMU), magnetic field (MAG) and GPS

2.11.3 Member Data Documentation

2.11.3.1 ActuationValue SIL::_actuation_value [private]

actuation value which corresponds to a desired rotor speed

The documentation for this class was generated from the following files:

- /home/usappz/DeadalusSim/source/sil/sil.hh

- /home/usappz/DeadalusSim/source/sil/sil.cc

2.12 Thrust Class Reference

Public Member Functions

- gazebo::math::Vector3 **vec3** (int nmb_motor)
- gazebo::math::Vector3 **sum_vec3** ()

Public Attributes

- double **motor** [4]

Private Attributes

- gazebo::math::Vector3 **motor_vec3** [4]

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

- /home/usappz/DeadalusSim/source/dependencies/resources.hh

2.13 Torque Class Reference

Public Member Functions

- gazebo::math::Vector3 **vec3** ()
- void **set_torque** (double r, double p, double y)

Public Attributes

- double **roll**
- double **pitch**
- double **yaw**

Private Attributes

- gazebo::math::Vector3 **torque_vec3**

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

- /home/usappz/DeadalusSim/source/dependencies/resources.hh