Gripper Interface

Generated by Doxygen 1.8.8

Tue Feb 24 2015 09:54:00

Contents

Chapter 1

Main Page

An Interface (C++ Linux Library - libgripper.a) to communicate with the Robotiq adaptive 3 Fingers gripper.
This interface is a communication layer that are built on top of the Modbus TCP Protocol.
It is advised that the Robotiq Instruction Manual be consulted before using this interface.
A showcase (demo.cpp) and a template (template.cpp) is available.

Created on: Jan 30, 2015
dragojub.surdilovic@ipk.fraunhofer.de
henry.chiu@ipk.fraunhofer.de

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Class Documentation

4.1 Gripper::Finger Class Reference

Individual Data of fingers and scissor.

```
#include <Gripper.h>
```

Public Member Functions

- void setPosition (int position)

  Sets position (range: fully open=0 to fully closed=255) for finger.
- void setSpeed (int speed)

  Sets speed (range: min=0 to max=255) for finger.
- void setForce (int force)

  Sets force (range: min=0 to max=255) for finger.
- void clear (void)

  Sets position, speed and force back to 0 for finger.
- bool isMoving (void) const

  Returns true if finger is in motion.
- int getPositionReq (void) const

  Returns position requested (Echo) of finger.
- int getPosition (void) const

  Returns actual position of finger.
- int getCurrent (void) const

  Returns electric current consumption (0.1∗getForce() in mA) of finger.

Friends

- class Gripper

4.1.1 Detailed Description

Individual Data of fingers and scissor.

Definition at line 58 of file Gripper.h.

4.1.2 Member Function Documentation

4.1.2.1 void Finger::clear (void)

Sets position, speed and force back to 0 for finger.

Definition at line 39 of file Gripper.cpp.

4.1.2.2 int Finger::getCurrent (void) const

Returns electric current consumption (0.1∗getForce() in mA) of finger.

Definition at line 68 of file Gripper.cpp.

4.1.2.3 int Finger::getPosition (void) const

Returns actual position of finger.

Definition at line 61 of file Gripper.cpp.

4.1.2.4 int Finger::getPositionReq (void) const

Returns position requested (Echo) of finger.

Definition at line 55 of file Gripper.cpp.

4.1.2.5 bool Finger::isMoving (void) const

Returns true if finger is in motion.

Definition at line 46 of file Gripper.cpp.

4.1.2.6 void Finger::setForce (int *force*)

Sets force (range: min=0 to max=255) for finger.

Definition at line 33 of file Gripper.cpp.

4.1.2.7 void Finger::setPosition (int *position*)

Sets position (range: fully open=0 to fully closed=255) for finger.

Definition at line 23 of file Gripper.cpp.

4.1.2.8 void Finger::setSpeed (int *speed*)

Sets speed (range: min=0 to max=255) for finger.

Definition at line 28 of file Gripper.cpp.

4.1.3 Friends And Related Function Documentation

4.1.3.1 friend class Gripper [friend]

Definition at line 69 of file Gripper.h.

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

- Gripper.h
- Gripper.cpp

4.2 Gripper Class Reference

Interface to communicate with gripper.

```
#include <Gripper.h>
```

Classes

- class Finger

  Individual Data of fingers and scissor.

Public Types

- enum Mode { Basic, Pinch, Wide, Scissor }

  gripper operation Mode
- enum Sync { SendOnly, ReadOnly, Dual }

  Synchronisation Channel.
- enum Fault { NoFault, ActionDelay, Minor, Major }

  Fault Status.

Public Member Functions

- Gripper ()

  Constructs a Gripper Object.
- virtual ∼Gripper ()

  Guarantees that the gripper is deactivated and disconnected.
- void connect (const char ∗ip, int port)

  Establishes a TCP Connection to the gripper.
- void disconnect (void)

  Disconnects from the gripper.
- void activate (Mode mode=Basic)

  Activates gripper in Mode mode.
- void setIndividualCtrl (bool rICF, bool rICS=false)

  Enables/disables individual control of fingers/scissor.
- void deactivate (void)

  Deactivates gripper.
- void emergencyRelease (void)

  Automatic Release routine.
- void synchronise (Sync Channel=Dual)

  Synchronises data between Interface and gripper according to the specified channel Sync.
- void setPosition (int position)

  Sets position (range: fully open=0 to fully closed=255) for gripper.
- void setSpeed (int speed)

  Sets speed (range: min=0 to max=255) for gripper.
- void setForce (int force)

  Sets force (range: min=0 to max=255) for gripper.
- void clear (void)

Sets position, speed and force back to 0 for all fingers and scissor.

- void go (bool flag)

  Go2 requested position (flag=true) or Stop (flag=false)
- bool isConnected (void) const

  Returns true if gripper is connected.
- bool isActivated (void) const

  Returns true if gripper has been activated in the corresponding mode and ready for command.
- bool isMoving (void) const

  Returns true if gripper is in motion towards requested position.
- Gripper::Mode getMode (void) const

  Returns gripper operation Mode mode.
- int getPositionReq (void) const

  Returns position requested (Echo) of gripper.
- int getPosition (void) const

  Returns actual position of gripper.
- int getCurrent (void) const

  Returns electric current consumption (0.1∗getForce() in mA) of gripper.
- Gripper::Fault getFaultStatus (void) const

  Returns Fault Status.
- string getFaultMsg (void) const

  Returns fault message.

Public Attributes

- class Gripper::Finger a
- class Gripper::Finger b
- class Gripper::Finger c
- class Gripper::Finger s

  objects of individual fingers and scissor

4.2.1 Detailed Description

Interface to communicate with gripper.

Definition at line 29 of file Gripper.h.

4.2.2 Member Enumeration Documentation

4.2.2.1 enum Gripper::Fault

Fault Status.

NoFault/ActionDelay (led off), Minor (led turns red), Major (led blinking red)

Enumerator

NoFault

ActionDelay

Minor

Major

Definition at line 51 of file Gripper.h.

4.2.2.2 enum Gripper::Mode

gripper operation Mode

Enumerator

- ***Basic***
- ***Pinch***
- ***Wide***
- ***Scissor***

Definition at line 44 of file Gripper.h.

4.2.2.3 enum Gripper::Sync

Synchronisation Channel.

Enumerator

- ***SendOnly***
- ***ReadOnly***
- ***Dual***

Definition at line 46 of file Gripper.h.

4.2.3 Constructor & Destructor Documentation

4.2.3.1 Gripper::Gripper ( )

Constructs a Gripper Object.

Definition at line 76 of file Gripper.cpp.

4.2.3.2 Gripper::∼Gripper ( ) `[virtual]`

Guarantees that the gripper is deactivated and disconnected.

Definition at line 88 of file Gripper.cpp.

4.2.4 Member Function Documentation

4.2.4.1 void Gripper::activate (Mode *mode* = Basic)

Activates gripper in Mode mode.

Definition at line 179 of file Gripper.cpp.

4.2.4.2 void Gripper::clear (void)

Sets position, speed and force back to 0 for all fingers and scissor.

clear() is implemented in Gripper::deactivate()

Definition at line 250 of file Gripper.cpp.

4.2.4.3 void Gripper::connect (const char ∗ *ip,* int *port*)

Establishes a TCP Connection to the gripper.

Definition at line 149 of file Gripper.cpp.

4.2.4.4 void Gripper::deactivate (void)

Deactivates gripper.

Definition at line 267 of file Gripper.cpp.

4.2.4.5 void Gripper::disconnect (void)

Disconnects from the gripper.

Definition at line 165 of file Gripper.cpp.

4.2.4.6 void Gripper::emergencyRelease (void)

Automatic Release routine.

usage: to disengage gripper after an emergency stop of robot
not to be used under normal operating conditions

Definition at line 276 of file Gripper.cpp.

4.2.4.7 int Gripper::getCurrent (void) const

Returns electric current consumption (0.1∗getForce() in mA) of gripper.

Definition at line 339 of file Gripper.cpp.

4.2.4.8 string Gripper::getFaultMsg (void) const

Returns fault message.

Definition at line 388 of file Gripper.cpp.

4.2.4.9 Gripper::Fault Gripper::getFaultStatus (void) const

Returns Fault Status.

Definition at line 370 of file Gripper.cpp.

4.2.4.10 Gripper::Mode Gripper::getMode (void) const

Returns gripper operation Mode mode.

Definition at line 303 of file Gripper.cpp.

4.2.4.11 int Gripper::getPosition (void) const

Returns actual position of gripper.

Definition at line 333 of file Gripper.cpp.

4.2.4.12 int Gripper::getPositionReq (void) const

Returns position requested (Echo) of gripper.

Definition at line 327 of file Gripper.cpp.

4.2.4.13 void Gripper::go (bool *flag*)

Go2 requested position (flag=true) or Stop (flag=false)

go(false) is implemented in Gripper::activate(Mode), Gripper::deactivate()
and Gripper::emergencyRelease()

Definition at line 261 of file Gripper.cpp.

4.2.4.14 bool Gripper::isActivated (void) const

Returns true if gripper has been activated in the corresponding mode and ready for command.

Definition at line 345 of file Gripper.cpp.

4.2.4.15 bool Gripper::isConnected (void) const

Returns true if gripper is connected.

Definition at line 174 of file Gripper.cpp.

4.2.4.16 bool Gripper::isMoving (void) const

Returns true if gripper is in motion towards requested position.

Definition at line 357 of file Gripper.cpp.

4.2.4.17 void Gripper::setForce (int *force*)

Sets force (range: min=0 to max=255) for gripper.

Definition at line 244 of file Gripper.cpp.

4.2.4.18 void Gripper::setIndividualCtrl (bool *rICF,* bool *rICS* = `false`)

Enables/disables individual control of fingers/scissor.

Parameters

bool	rICF: Individual control of fingers
bool	rICS: Individual control of scissor caution: position request, speed and force for ALL fingers will be initialised, resepctively to the current position, speed and force of gripper. option rICF is discarded by gripper if mode is set to Scissor and option rICS is set to false mode is discarded by gripper if option rICS is set to true

Definition at line 207 of file Gripper.cpp.

4.2.4.19 void Gripper::setPosition (int *position*)

Sets position (range: fully open=0 to fully closed=255) for gripper.

Definition at line 232 of file Gripper.cpp.

4.2.4.20 void Gripper::setSpeed (int *speed*)

Sets speed (range: min=0 to max=255) for gripper.

Definition at line 238 of file Gripper.cpp.

4.2.4.21 void Gripper::synchronise (Sync *Channel* = Dual)

Synchronises data between Interface and gripper according to the specified channel Sync.

data are NOT exchanged with the gripper until this method is called
this method does nothing until the gripper is connected.
usage: to be implemented in a real time communication thread
method can also be called in a one shot manner (i.e. threadfree/call at will)
maximum calling frequency: 1 millisecond

Definition at line 282 of file Gripper.cpp.

4.2.5 Member Data Documentation

4.2.5.1 class Gripper::Finger Gripper::a

4.2.5.2 class Gripper::Finger Gripper::b

4.2.5.3 class Gripper::Finger Gripper::c

4.2.5.4 class Gripper::Finger Gripper::s

objects of individual fingers and scissor

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

- Gripper.h
- Gripper.cpp

Chapter 5

File Documentation

5.1 demo.cpp File Reference

```
#include "Gripper.h"
#include <pthread.h>
```

Macros

- #define sec 1000000

  sec in microseconds
- #define ms 1000

  ms in microseconds

Enumerations

- enum Thread { Com, Fault, Com, Fault }

  thread purposes

Functions

- pthread_t launch (Gripper ∗, Thread)

  thread launcher
- void terminate (pthread_t)

  thread terminator
- int main ()
- void ∗ com (void ∗gripper)
- void ∗ fault (void ∗gripper)

5.1.1 Macro Definition Documentation

5.1.1.1 #define ms 1000

ms in microseconds

Definition at line 10 of file demo.cpp.

5.1.1.2 #define sec 1000000

sec in microseconds

Definition at line 9 of file demo.cpp.

5.1.2 Enumeration Type Documentation

5.1.2.1 enum Thread

thread purposes

Enumerator

- ***Com***
- ***Fault***
- ***Com***
- ***Fault***

Definition at line 13 of file demo.cpp.

5.1.3 Function Documentation

5.1.3.1 void∗ com (void ∗ *gripper*)

Definition at line 158 of file demo.cpp.

5.1.3.2 void∗ fault (void ∗ *gripper*)

Definition at line 169 of file demo.cpp.

5.1.3.3 pthread_t launch (Gripper ∗ *gripper,* Thread *type*)

thread launcher

Definition at line 183 of file demo.cpp.

5.1.3.4 int main ( )

<start communication

<start fault monitoring

Connect

Activation

Mode Change

<set mode chaning speed

Decelerate full close

Advanced feature

Restore and Deactivate

Definition at line 17 of file demo.cpp.

5.1.3.5 void terminate (pthread_t *id*)

thread terminator

Definition at line 194 of file demo.cpp.

5.2 Gripper.cpp File Reference

```
#include "Gripper.h"
```

5.3 Gripper.h File Reference

```
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>
#include <errno.h>
#include <bitset>
#include "modbus/modbus.h"
#include <iostream>
```

Classes

- class Gripper

  Interface to communicate with gripper.
- class Gripper::Finger

  Individual Data of fingers and scissor.

5.4 template.cpp File Reference

```
#include "Gripper.h"
#include <pthread.h>
```

Macros

- #define sec 1000000

  sec in microseconds
- #define ms 1000

  ms in microseconds

Enumerations

- enum Thread { Com, Fault, Com, Fault }

  thread purposes

Functions

- pthread_t launch (Gripper ∗, Thread)

    thread launcher
- void terminate (pthread_t)

    thread terminator
- int main ()
- void ∗ com (void ∗gripper)
- void ∗ fault (void ∗gripper)

5.4.1 Macro Definition Documentation

5.4.1.1 #define ms 1000

ms in microseconds

Definition at line 10 of file template.cpp.

5.4.1.2 #define sec 1000000

sec in microseconds

Definition at line 9 of file template.cpp.

5.4.2 Enumeration Type Documentation

5.4.2.1 enum Thread

thread purposes

Enumerator

Com

Fault

Com

Fault

Definition at line 13 of file template.cpp.

5.4.3 Function Documentation

5.4.3.1 void∗ com (void ∗ *gripper*)

Definition at line 86 of file template.cpp.

5.4.3.2 void∗ fault (void ∗ *gripper*)

Definition at line 97 of file template.cpp.

5.4.3.3 pthread_t launch (Gripper ∗ *gripper,* Thread *type*)

thread launcher

Definition at line 108 of file template.cpp.

5.4.3.4 int main ( )

<start communication

<start fault monitoring

Connect

Activation

Deactivate

Definition at line 23 of file template.cpp.

5.4.3.5 void terminate (pthread_t *id*)

thread terminator

Definition at line 119 of file template.cpp.

Index