My Project

Generated by Doxygen 1.9.4

1 Class Index **1**

1.1 Class List 1

2 Class Documentation **3**

2.1 RobotAPI_class.RobotAPI Class Reference 3

2.1.1 Member Function Documentation 4

2.1.1.1 getJointAngle() 4

2.1.1.2 getJointState() 4

2.1.1.3 getRobotState() 5

2.1.1.4 getTorqueStatus() 5

2.1.1.5 goHome() 5

2.1.1.6 penDown() 6

2.1.1.7 penUp() 6

2.1.1.8 pingTest() 6

2.1.1.9 setGetRobotState() 7

2.1.1.10 setJointAngle() 7

2.1.1.11 setJointState() 7

2.1.1.12 setRobotState() 8

2.1.1.13 setTorqueOFF() 8

2.1.1.14 setTorqueON() 8

Chapter 1

LabRobotsIITK

Python API and Arduino codes for operating the planar 2R and 5-bar robots.

Python libraries needed:

1. PySerialTransfer
   ```
   pip install pySerialTransfer
   ```

Arduino libraries needed:

1. SerialTransfer
2. DynamixelShield
3. Dynamixel2Arduino
4. Servo

TODO:

1. Edit servo movements for the 5-bar robot

Chapter 2

Class Index

2.1 Class List

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

RobotAPI_class.RobotAPI 3

Chapter 3

Class Documentation

3.1 RobotAPI_class.RobotAPI Class Reference

Public Member Functions

- def **__init__** (self, str port, int baud_rate, str robot_type)
- def **__del__** (self)
- def pingTest (self, int motorID)
- def setTorqueON (self, int motorID)
- def setTorqueOFF (self, int motorID)
- def getTorqueStatus (self, int motorID)
- def getJointAngle (self, int motorID, str unit="rad")
- def setJointAngle (self, int motorID, float angle, str unit="rad")
- def penDown (self)
- def penUp (self)
- def getJointState (self, int motorID)
- def setJointState (self, int motorID, list state)
- def getRobotState (self)
- def setRobotState (self, list state)
- def goHome (self)
- def setGetRobotState (self, list state)

Public Attributes

- **link**
- **rob**

Static Public Attributes

- int **ADDR_PC** = 100
- int **ADDR_MEGA** = 200
- int **FC_PING** = 1
- int **FC_TORQUE_ON** = 2
- int **FC_TORQUE_OFF** = 3
- int **FC_TORQUE_STATUS** = 4
- int **FC_READ_ANGLE** = 5

- int **FC_WRITE_ANGLE** = 6
- int **FC_PEN_SERVO** = 7
- int **FC_READ_STATE** = 8
- int **FC_WRITE_STATE** = 9
- int **FC_READ_ROBOT_STATE** = 10
- int **FC_WRITE_ROBOT_STATE** = 11
- int **FC_RW_ROBOT_STATE** = 12
- float **RS485_TO** = 0.2

3.1.1 Member Function Documentation

3.1.1.1 getJointAngle()

```
def RobotAPI_class.RobotAPI.getJointAngle (
             self,
            int motorID,
            str  unit = "rad" )
```

```
Description: This function reads the motor joint angle

Input:
motorID: The ID of the motor
unit: optional input. Default is 'rad'. Use 'deg' for degrees.

Output:
joint angle
-1 if failed to get the joint angle
```

3.1.1.2 getJointState()

```
def RobotAPI_class.RobotAPI.getJointState (
             self,
            int motorID )
```

```
Description: This function reads the motor joint angle and joint velocity

Input:
motorID: The ID of the motor

Output:
[joint angle (rad), joint velocity (rad/s)]
-1 if failed to get the state
```

3.1.1.3 getRobotState()

```
def RobotAPI_class.RobotAPI.getRobotState (
              self )
```

Description: This function reads the robot joint angles and joint velocities.

Input: -NA-

Output:
For 2R robot: JA1, JV1 corresponds to the base motor.
For 5bar robot: JA1, JV1 corresponds to the left motor.
[[JA1, JV1],[JA2, JV2]] units: rad for angle, rad/s for velocity
-1 if failed to get the state

3.1.1.4 getTorqueStatus()

```
def RobotAPI_class.RobotAPI.getTorqueStatus (
              self,
             int motorID )
```

Description: This function checks the torque enabled/disabled status.

Input:
motorID: The ID of the motor to check the torque status

Output:
1 if the motor torque is enabled
0 if the motor torque is disabled
-1 if failed to get the torque status

3.1.1.5 goHome()

```
def RobotAPI_class.RobotAPI.goHome (
              self )
```

Description:
This function moves the robot to the home position.

Input: -NA-

Output: success/failure

3.1.1.6 penDown()

```
def RobotAPI_class.RobotAPI.penDown (
              self )
```

```
Description:
Pen down

Input: -NA-

Output:
No feedback available from the servo motor. Inspect visually.
```

3.1.1.7 penUp()

```
def RobotAPI_class.RobotAPI.penUp (
              self )
```

```
Description:
Pen up (lifted in air)

Input: -NA-

Output:
No feedback available from the servo motor. Inspect visually.
```

3.1.1.8 pingTest()

```
def RobotAPI_class.RobotAPI.pingTest (
              self,
             int motorID )
```

```
Description: This function sends a ping command
to the specified motor and waits for a response.

Input:
motorID: The ID of the motor to ping

Output:
True if the motor responds to the ping command
False if the motor does not respond to the ping command
```

3.1.1.9 setGetRobotState()

```
def RobotAPI_class.RobotAPI.setGetRobotState (
              self,
             list state )
```

```
Description:
This function sets the robot joint angles and joint velocities synchronously.
And then reads the instantaneous robot joint angles and joint velocities.

Input:
state: [[JA1, JV1],[JA2, JV2]] units: rad for angle, rad/s for velocity
For 2R robot: JA1, JV1 corresponds to the base motor.
For 5bar robot: JA1, JV1 corresponds to the left motor.

Output:
For 2R robot: JA1, JV1 corresponds to the base motor.
For 5bar robot: JA1, JV1 corresponds to the left motor.
[[JA1, JV1],[JA2, JV2]] units: rad for angle, rad/s for velocity
-1 if failed to get the state
```

3.1.1.10 setJointAngle()

```
def RobotAPI_class.RobotAPI.setJointAngle (
              self,
             int motorID,
             float angle,
             str  unit = "rad" )
```

```
Description:
This function sets the motor joint angle.
The motor torque is enabled automatically.

Input:
motorID: The ID of the motor
angle: The angle
unit: optional input. Default is 'rad'. Use 'deg' for degrees.

Output:
success/failure
```

3.1.1.11 setJointState()

```
def RobotAPI_class.RobotAPI.setJointState (
              self,
             int motorID,
             list state )
```

```
Description:
This function sets the motor joint angle and joint velocity.
The motor torque is enabled automatically.

Input:
motorID: The ID of the motor
state: [joint angle (rad), joint velocity (rad/s)]

Output:
success/failure
```

3.1.1.12 setRobotState()

```
def RobotAPI_class.RobotAPI.setRobotState (
              self,
             list state )
```

```
Description:
This function sets the robot joint angles and joint velocities synchronously.
The motor torques are enabled automatically.

Input:
state: [[JA1, JV1],[JA2, JV2]] units: rad for angle, rad/s for velocity

Output:
success/failure
```

3.1.1.13 setTorqueOFF()

```
def RobotAPI_class.RobotAPI.setTorqueOFF (
              self,
             int motorID )
```

```
Description: This function disables the motor control
of the specified motor and waits for a response.

Input:
motorID: The ID of the motor to disable torque

Output:
True if the torque is disabled
False if failed to disable the torque
```

3.1.1.14 setTorqueON()

```
def RobotAPI_class.RobotAPI.setTorqueON (
              self,
             int motorID )
```

```
Description: This function enables the motor control
of the specified motor and waits for a response.

Input:
motorID: The ID of the motor to enable torque

Output:
True if the torque is enabled
False if failed to enabled the torque
```

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

- RobotAPI_class.py