
pyJD Documentation

Release 0.1

BrutusTT

Aug 20, 2017

Contents

1	Installation	3
2	Usage	5
3	Modules	7
4	License	11

pyJD is a collection of [Yarp](#) Modules to control the [EZ-Robot JD Humanoid Robot](#).

CHAPTER 1

Installation

1. Install the dependencies:

You need Yarp installed with the python bindings. For more details see [full instructions](#).

Example: OSX using [Homebrew](#)

```
brew tap homebrew/x11  
brew install --with-python yarp
```

2. Download the source code:

```
git clone https://github.com/BrutusTT/pyJD
```

3. build and install:

```
cd pyJD  
python setup.py install
```


CHAPTER 2

Usage

Each module can be started standalone using the following command line.

```
<ModuleName> [--ip <IP Address>] [--port <Port>] [--name <Name Prefix>]
```

Parameters:

```
[ ]           - denotes optional parameter
<ModuleName> - can be one of the following: - JModule
                                              - JDVideo
                                              - JDFollowFace
<IP Address> - default is 192.168.1.1
<Port>       - default is 23
<Name Prefix> - if a name is given it will be used as a prefix for the port names
                e.g.: --name test results in /test/JModule/rpc
```

Example:

```
$ JModule --name MyRobot
```


JDModule

The JDModule can be used to set JD's joints to specified positions.

`/JDModule/rpc`

- **Type:** RPC
- **Message Format:** `set pos <joint_id> <position>`
- **Parameter**
 - `<joint_id>`: integer [0 - 23]
 - `<position>`: integer [0 - 180]
- **Response**
 - `ack` - command processed
 - `nack` - error occurred
- **Message Example**
 - `set pos 0 13` - JD looks to the left.

`/JDModule/lookAt:i`

- **Type:** Buffered Port
- **Message Format:** `<near-far> <left-right> <down-up>`
- **Parameter**
 - `<near-far>` float - distance in meters [0.1, -inf] ***Be aware: negative distance!***

- <left-right> float - distance in meters [-inf, inf]
- <down-up> float - distance in meters [-inf, inf]

- **Message Example**

- -1.0 0.5 0.0 - Looks to the left (Fixation Point: 1m in front + 50cm to the left side)

`/JDModule/pointLeft:i`

- **Type:** Buffered Port

- **Message Format:** <near-far> <left-right> <down-up>

- **Parameter**

- <near-far> float - distance in meters [inf, -inf] ***Be aware: negative distance!***
 - <left-right> float - distance in meters [-inf, inf]
 - <down-up> float - distance in meters [-inf, inf]

- **Message Example**

- -1.0 0.5 0.0 - Points to the left (Point: 1m in front + 50cm to the left side)

`/JDModule/pointRight:i`

- **Type:** Buffered Port

- **Message Format:** <near-far> <left-right> <down-up>

- **Parameter**

- <near-far> float - distance in meters [inf, -inf] ***Be aware: negative distance!***
 - <left-right> float - distance in meters [-inf, inf]
 - <down-up> float - distance in meters [-inf, inf]

- **Message Example**

- -1.0 0.5 0.0 - Points to the left (Point: 1m in front + 50cm to the left side)

JDVideo

This module is used to receive a video stream from the JD robot.

`/JDVideo/img:o`

- **Type:** Output

- **Message Format:** RGB Image [320 x 200]

JDFollowFace

This module is used to calculate the head joint positions provided that it receives a face position. The output port of this module should be connected to the [JDLookAtModule](#).

/JDFollowFace/face

- **Type:** Input
- **Message Format:** <face_count> ((<x0> <y0>) (<x1> <y1>) ...)
- **Parameter**
 - <face_count>: integer - number of faces in the list
 - <xn>: integer - x-center of the n-th face
 - <yn>: integer - y-center of the n-th face

/JDFollowFace/position

- **Type:** Output
- **Message Format:** <near_far> <left_right> <down_up>
- **Parameter**
 - <near_far>: double
 - <left_right>: double
 - <down_up>: double

Happy hacking!

CHAPTER 4

License

See COPYING for licensing info.