# NOTE interface to KUKA robot

* This note contains a LITTLE documentation related to the interface between the KUKA robot and an external control. This information could be subject to change…

Date: 13/03/2016

Ole Madsen (om@m-tech.aau.dk)

Contents

[NOTE interface to KUKA robot 1](#_Toc445654191)

[1. KUKA Robot-server: Network setup 2](#_Toc445654192)

[2. Commands (from extern computer to KUKA) 2](#_Toc445654193)

[2.1 Commands to the KUKA server 3](#_Toc445654194)

[3. Use. 5](#_Toc445654195)

[3.1. Start-up: 5](#_Toc445654196)

[3.2. MATLAB 5](#_Toc445654197)

# KUKA Robot-server: Network setup

Communication with the KUKA robot is done over TCP/IP using a simple request/response structure. The KUKA robot acts as server in this control. You must connect to the robot controller via the given IP and port. The ip addresses are:

Wall-E 192.168.1.200

Jarviz 192.168.1.210

KUKA server

Ip = 192.168.1.210

port = 54601

Extern computer

# Commands (from extern computer to KUKA)

The communication is based on transmitting a byte array. On the robot side, the byte array is decoded into the appropriate variables. On the PC side, the array is constructed as a character array (or string). Each message is terminated by a carriage return and line feed [carriage return][line feed].

A number of commands have been defined between the KUKA robot and the extern computer.

The general syntax of a command is

<CommandType> <Data> <carriage return> <line feed>

where

**Commandtype** is an integer characterizing the command send to the robot server

**Data** are data related to the message (e.g positional data related to a move command. Note that in some cases there are not data, e.g. open/close gripper.

The robot gives the following replies to the commands:

<commandType>;started; (immediately after receiving the command)

<commandType>;ended; (after successful execution of the commend)

<commandType>;ERROR,<description of error>

In case of an unknown command the server will return nothing.

## 2.1 Commands to the KUKA server

|  |  |
| --- | --- |
| **1 <double> <double> <double> <double> <double> <double> <double>** | |
| **data:** | J1, J2, J3, J4, J5, J6, velocity  Configuration of robot in degrees of each axis |
| **Description:** | Moves robot to the given joint configuration |
| **Response** | MJ;started; (when message received and movements started)  MJ;ended; |
| **Example:** | 1 0.0 0.0 90.0 54.0 0.0 0.0 1.0 |
| **Matlab:** | moveJoint(r,0.0, 0.0, 90.0, 54.0, 0.0, 0.0, 1.0) |

|  |  |
| --- | --- |
| **2 <double> <double> <double> <double> <double> <double> <double>** | |
| **data:** | X,Y,Z,R, P, Y, velocity  Location of TOOL relatively to WORLD coordinates |
| **Description:** | Moves and rotates robot end-effector (TOOL) to the desired location and orientation using PTP-move. |
| **Response** | MC;started; (when message received and movements started)  MC;ended; |
| **Example:** | 2 400 0.0 200.0 54.0 0.0 0.0 1.0 |
| **Matlab:** | moveCart(r,400,0.0,200.0,54.0,0.0,0.0,1.0) |

|  |  |
| --- | --- |
| **3 <double> <double> <double> <double> <double> <double> <double>** | |
| **data:** | X,Y,Z,R, P, Y, velocity  Location of TOOL relatively to WORLD coordinates |
| **Description:** | Moves and rotates robot end-effector (TOOL) to the desired location and orientation using LIN-move. |
| **Response** | ML;started; (when message received and movements started)  ML;ended; |
| **Example:** | 3 400 0.0 200.0 54.0 0.0 0.0 1.0 |
| **Matlab:** | moveLinear(r,400,0.0,200.0,54.0,0.0,0.0,1.0) |

|  |  |
| --- | --- |
| **4 <double> <double> <double> <double> <double> <double> <double>** | |
| **data:** | J1, J2, J3, J4, J5, J6, velocity  Change in degrees of each axis. |
| **Description:** | Relative joint (PTP) move. The joints are moved the given degrees according to their current position |
| **Response** | JR;started; (when message received and movements started)  JR;ended; |
| **Example:** | 4 0.0 3.0 6.0 0.0 0.0 0.0 1.0 |
| **Matlab:** | moveRelativeJoint(r,0.0,3.0,6.0,0.0,0.0,0.0,1.0) |

|  |  |
| --- | --- |
| **5 <double> <double> <double> <double> <double> <double> <double>** | |
| **data:** | X,Y,Z,R, P, Y, velocity  Relative movement of TCP in each axis/rotation |
| **Description:** | Relative linear move and rotation. The robot end-effector (TOOL0) is translated the dX,dY,dZ and rotation is changed: dR, dP, dY. |
| **Response** | LR;started; (when message received and movements started)  LR;ended; |
| **Example:** | 5 400 0.0 0 0.0 0.0 0.0 1.0 |
| **Matlab:** | moveRelativeLinear(r,400,0.0,0.00.0,0.0,0.0,1.0) |

|  |  |
| --- | --- |
| **6** | |
| **Data:** | None |
| **Description:** | Open the gripper |
| **Example:** | 6 |
| **Matlab** | openGrapper(r) |

|  |  |
| --- | --- |
| **7** | |
| **Data:** | None |
| **Description:** | Closes the gripper |
| **Example:** | 7 |
| **Matlab** | closeGrapper(r) |

|  |  |
| --- | --- |
| **8** | |
| **data:** | none |
| **Description:** | Returns the current joint configuration of the robot (in degrees) |
| **Response** | <double><double><double><double><double><double> |
| **Example:** | 8 |
| **Matlab:** | [] = getJoints(r) |

|  |  |
| --- | --- |
| **9** | |
| **data:** | none |
| **Description:** | Returns the current Cartesian position of the TCP (in mm and degrees) |
| **Response** | <double><double><double><double><double><double> |
| **Example:** | 9 |
| **Matlab:** | [] = getPosition(r) |

# Use.

## Start-up:

Start the program: “KukaServer” in the “ethernet" folder on the KUKA robot.

## 3.2. MATLAB

A matlab Class: KukaRobotConnector has been made (Appendix B). Put this into the MatLab path

A simple MATLAB program could look as:

r = RobotConnector; - connects to the robot

openGrapper(r) - open gripper

moveRelativeLinear (r, 0,0,-100,0,0,0,100);

closeGrapper(r);

moveJoint (r,100,100,100,0,0,0,100);

For further documentation of the matlab script, please refer to the documentation and guide in the top of the matlab file itself.