KUKA CONTROL TOOLBOX

For use with MATLAB®

3.0

User's guide

Version 3.0

June 2010

KUKA Control Toolbox

F.Chinello, S.Scheggi, F.Morbidi, D.Prattichizzo

{chinello,scheggi,morbidi,prattichizzo}@dii.unisi.it

Siena Robotics and Systems Lab (SIRSlab)©

Dept. of Information Engineering, University of Siena,

Italy September,2009.

Overview

The Kuka Control Toolbox (KCT) is a collection of MATLAB functions develo- ped at the University of Siena, for motion control of KUKA robot manipulators. The toolbox, which is compatible with all 6 DOF small and low payload KUKA robots that use the Eth.RSIXML, runs on a remote computer connected with the KUKA controller via TCP/IP. KCT includes more than 40 functions, spanning operations such as forward and inverse kinematics computation, point-to-point joint and Cartesian control, trajectory generation, graphical display, 3-D anima- tion and diagnostics. Applicative examples show the flexibility and reliability of KCT, and its easy integration with other MATLAB toolboxes for complex vision and telepresence tasks.

References

The Kuka Control Toolbox: motion control of KUKA robot manipulators with MATLAB, F. Chinello, S. Scheggi, F. Morbidi, D. Prattichizzo, submitted to IEEE Robot. Autom. Mag., December 2009.

System Requirements

- KUKA System Software (KSS) 5.4 or higher
- KUKA Robot Sensor Interface XML (RSIXML) 1.1.0 or higher
- KUKA real-time network card

Recommended MATLAB® 7.0.1 or higher.

Installation

To correctly install the Kuka Control Toolbox:

1. Copy the file kctserver.exe (contained in kct/kctrsi/) in the KUKA Robot Controller (KRC).
2. Copy the file kctrsiclient.src (contained in kct/kctrsi/) inside the specific program folder of the KUKA System Software (KSS) in the KRC (e.g. C:/KRC/ROBOTER/.../R1/Program/).
3. In MATLAB set the folder KCTv2p0 as current directory or add the folder KCTv2p0 to the MATLAB programs path.

Initialization

The information relative to the KUKA robots supported by KCT is stored in the MATLAB file `kctrobotdata.mat` and can be accessed by typing:

```
kctrobot();
```

To initialize a particular robot model, it is sufficient to write,

```
kctinit('KR3');
```

where the argument is a string containing the name of the selected robot (e.g., KR3, KR5r650, KR5r850, . . . , etc.) as specified in `kctrobotdata.mat`.

To insert a new robot model type,

```
kctlinks = [335, 75, 365, 90, 405, 80];
kctinsertrobot('KR5sixxr850', kctlinks);
```

where the first argument is the name the user gives to the robot and `kctlinks` is a vector containing the length of the links (millimeters) of the robot as in Fig 1

To remove a particular robot model from the robots' list, for example the KR5sixxr8590 model, it's sufficient to write,

```
kctdeleterobot('KR5sixxr850');
```

The information relative to the KUKA robots supported by KCT is stored in the MATLAB file `kctrobotdata.mat` and can be accessed by typing:

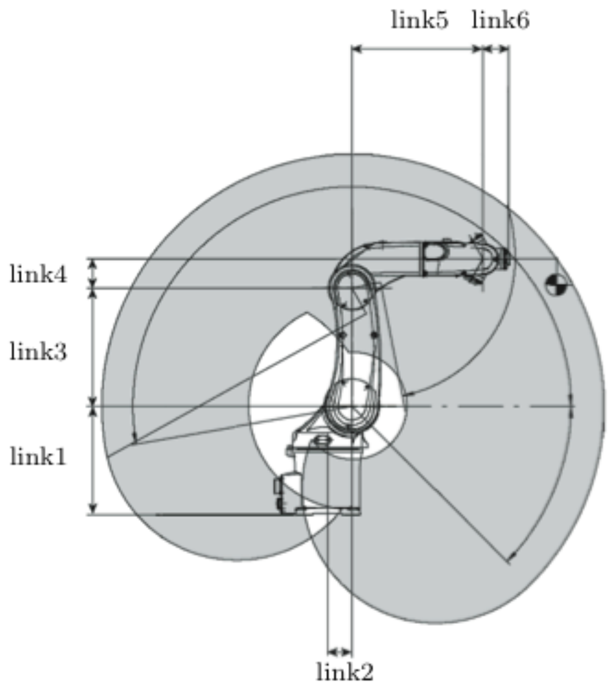


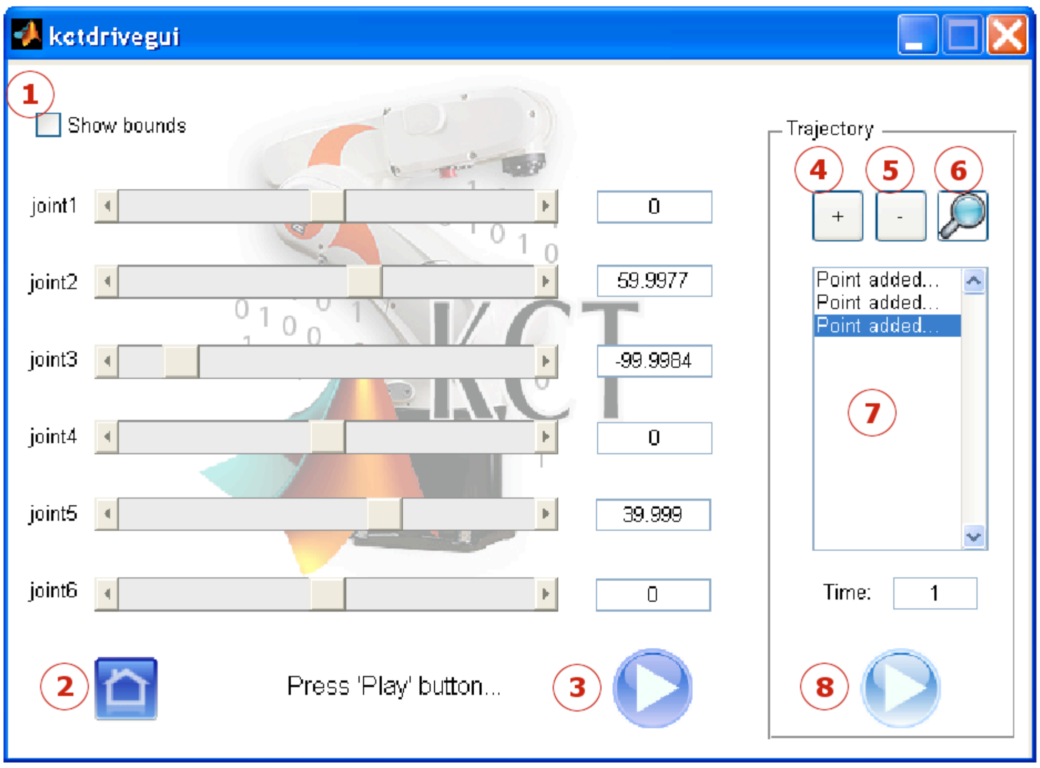
Figure 1: Workspace and links of robot KR5sixxr850.

After the initialization step, the TCP/IP communication between KCT and kctserver must be established. The main steps necessary to initialize the network communication are listed below (in what follows, we assume the reader has some familiarity with KRC and RSI operation):

1. In the KUKA control panel (teach pendant), put the mode selector on AUT (automatic execution).
2. Select `kctrsiclient.src` on KSS.
3. Start `kctserver.exe`.
4. In the MATLAB workspace start the KCT/IP connection with kctserver by typing, `t = kctclient('192.168.1.0');` where `t` is the handle to the network communication and 192.168.1.0 is the IP address of the KRC real-time network card.
5. Start `kctrsiclient.src` by pressing the run button on the KUKA control panel and keep pushing until the `ST_SKIPSENS` line is reached.
6. Check whether the communication is established. If not, return to step 2.

To terminate the TCP/IP communication is sufficient to type:

```
kctcloseclient();
```



Graphical User Interface

A GUI (Graphical User Interface) can be loaded by typing,

```
    kctdrivegui();
```

It's possible to set the values of the robot joints by moving the sliders of the graphical interface. The robot configuration and animation is shown in the apposite window. By clicking on ①, it's possible to enable or disable the visualization of the robot bounds. By clicking on ②, it's possible to move the robot to the configuration shown in the 3-D animation window. To drive the robot back to the starting position just press button ② and then press button ③ to start the motion.

The trajectory control panel on the right hand side of the GUI allows to visually plan the 3-D trajectory of the robot. By clicking on ④, the current values of the joints are stored and the robot configuration is added in the trajectory list ⑦. By clicking on ⑤ it's possible to remove the selected configuration from the robot trajectory list. Button ⑥ allows to visualize in the 3-D animation window the selected robot configuration. Finally ⑧ starts the robot trajectory motion.

Demos

The script `kctdemo.m` shows the steps necessary to correctly initialize and control the robot manipulator (type kctdemo in the MATLAB command window to execute the script).

After the robot initialization step,

```
kctinit(kctmodel);
```

and the network communication initialization,

```
kctclient(addr,0.1);
```

the Motion control functions can be used. In particular,

```
[robotstate,warn] = kctsetjoint([0, 0, 45, 0, -45, 0]’);
```

moves the robot from the current to a desired configuration defined by the joint angles vector `[0 0 45 0 -45 0]’`.

To visualize the time history of the joint angles of the robots, one should type:

```
kctdispdyn(robotstate);
```

Finally, to move the robot along a given trajectory, define the following 5 × 6 matrix whose rows are a sequence of Cartesian frames:

```
P = [450 225 100   0  90   0;
     450 225  550 10  60 -10;
     450 -225 550  0  90   0;
     450 -225 100  0  90  20;
     450  225 100     90   0];
```

and then, type :

```
kctpathxyz(P,30,1);
```

In order to simplify robot’s motion control, a Graphical User Interface is available in KCT:

```
kctdrivegui();
```

List of function

Initialization	
kctrobot	Show the list of supported robots
kctfindrobot	Search the specified robot in the models’ list
kctinsertrobot	Add the specified robot to the models’ list
kctdeleterobot	Remove the robot from the models’ list
kctinit	Load the parameters of the selected robot
kctsetbound	Set the workspace bounds
kctgetbound	Visualize the workspace bounds
kctchecksystem	Check MATLAB’s version and whether the Instrument Control Toolbox is installed

Networking	
kctsettcpip	Set the TCP/IP communication modality
kctgettcpip	Return the TCP/IP communication modality
kctclient	Initialize the client
kctcloseclient	Terminate the client
kctclientmex	Initialize the client (MEX-file)
kctcloseclientmex	Terminate the client (MEX-file)
kctsenddatamex	Send data to kctserver
kctrecdatamex	Receive data from kctserver

Kinematics	
kctreadstate	Return the current configuration of the robot
kctfkine	Compute the forward kinematics
kctikine	Compute the inverse kinematics
kctfkinerpy	Compute the forward kinematics (return the pose)
kctikinerpy	Compute the inverse kinematics (from the pose)

Motion control	
kctsetjoint	Set the joint angles to a desired value
kctsetxyz	Move the end-effector in a desired position
kctmovejoint	Set the joint velocities to a desired value
kctmovexyz	Move the end-effector with a desired linear and angular velocity
kctdrivegui	GUI for robot motion control
kctpathxyz	Generate a trajectory (operational space)
kctpathjoint	Generate a trajectory (joint space)
kcthome	Drive the robot back to the starting position
kctstop	Stop the robot in the current position

Graphics	
kctdisprobot	Plot the robot in the desired configuration
kctdisptraj	Plot the 3-D trajectory of the end-effector
kctdispdyn	Plot the time history of the joint angles
kctanimtraj	Create a 3-D animation of the robot

Homog. Transforms	
kctrotox	Hom. transform for rotation about X-axis
kctrotoy	Hom. transform for rotation about Y -axis
kctrotoz	Hom. transform for rotation about Z-axis
kcttran	Hom. transform for translation
kctchframe	Change the reference frame
kctgetframe	Change the reference frame

Demos	
kctdemo	General demonstration of the toolbox
kctdemovision	Vision demo
kctdemohaptik	Haptic demo