SCHOOL OF MATHEMATICAL AND COMPUTER SCIENCES

The iCub Simulator and vision

Exercise 4. Motor Control

In the last exercise we used position and velocity control to move the iCub_Sim head. Now the challenge is to use the following components to make the iCub Simulator use its head to draw a circle in the air, by using the iKinGazeCtrl module.

Start your yarpserver and iCub_SIM with the following commands:

```
Yarpserver and iCub_sim
```

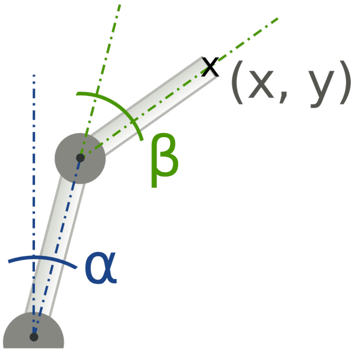
To test these settings enter the following commands:

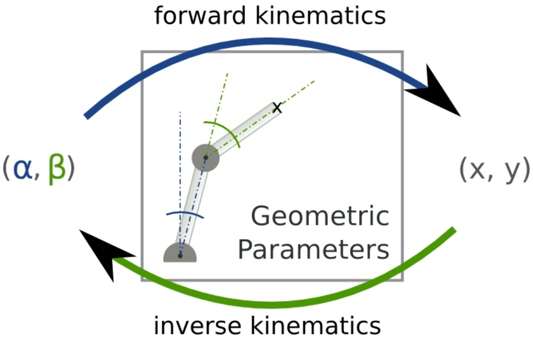
```
"C:\Program Files (x86)\robotology\icub-1.1.16\bin\iKinGazeCtrl" --robot icubSim
```

Exercise 4.1 Learn about kinematics

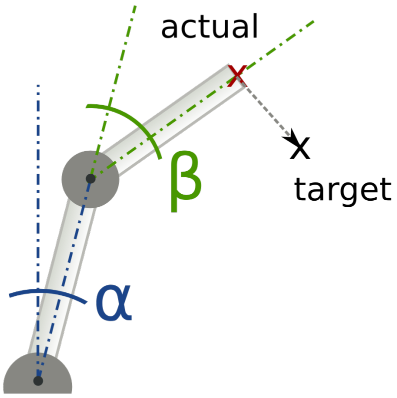
Controlling iCub (Task Space / Joint Space) :

Trajectory in Task Space (Cartesian Coordinates)
Control the Robot Joints (Joint Space)





Forward Kinematics: "Where is the end-effector?"
Inverse Kinematics: "How to move the joints?"



Provides:
Solver: Forward kinematics, Inverse kinematics
Controller: Comparison of set and actual value, Control
You will learn how to calculate the inverse kinematics in Nick’s lectures/tutorials and you can read more details here:

https://en.wikipedia.org/wiki/Inverse_kinematics

Exercise 4.2 Test the module with the Web-interface:

Go to your Browser and type:

```
Localhost:10000
```

Press enter.

Go to the port **/iKinGazeCtrl/xd:i** and type the following line in the text field:

```
-3.0 0 0.4 [enter]
```

Observe your iCub Simulator! WARNING! don’t use too big values, otherwise the simulation will crash!

```
-3.0 0.25 0.4 [enter]
```

```
-3.0 -0.25 0.4 [enter]
```

Exercise 4.3 The “bottle” you need to provide to get this in your source code:

/iKinGazeCtrl/xd:i

Your bottle? needs to include 3 double values:
1. The x axis (first value): looking into the distance; need always to be negative to look not inside the head; -3.0 is to look 3 m in the distance.
2. The y axis (second value): looking left and right; values between -0.25 and 0.25 are good.
3. The z axis (third value): looking up and down; values between 0.2 and 0.6 are good.

Write your own little port and send 4 test values, e.g. to make the robot shake its head or to node.

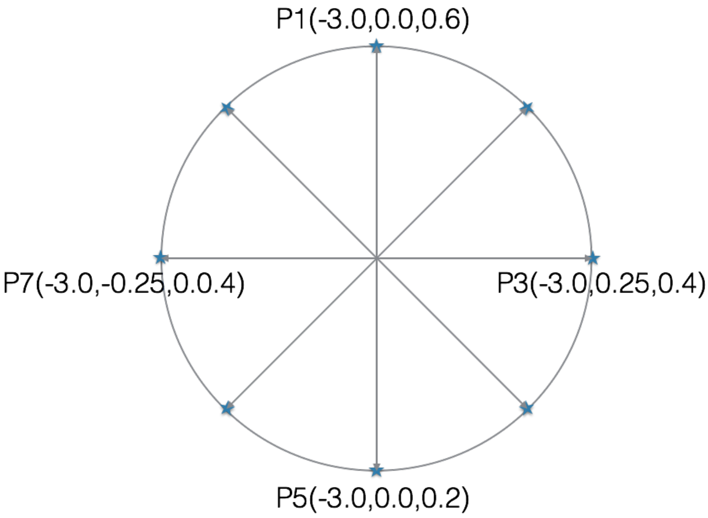
How to write a yarp port (values are in double “addDouble”):

http://www.yarp.it/yarp_cmake_hello.html

Use these links to get more details on how to write the code:
http://eris.liralab.it/iCub/main/dox/html/group__iKinGazeCtrl.html
http://eris.liralab.it/brain/icub_gaze_interface.html

Exercise 4.4 Create a trajectory and follow it

User defined (circle), by giving at least 8 target points to the robot:



Check if the head is near your current target by using the following ports:

/iKinGazeCtrl/x:o returns the actual fixation point (Vector of 3 double). Units in meters.
/iKinGazeCtrl/q:o returns the actual joints configuration during movement (Vector of 9 double). The order for torso angles is the one defined by kinematic chain (reversed order). Useful in conjunction with the viewer. Units in degrees.