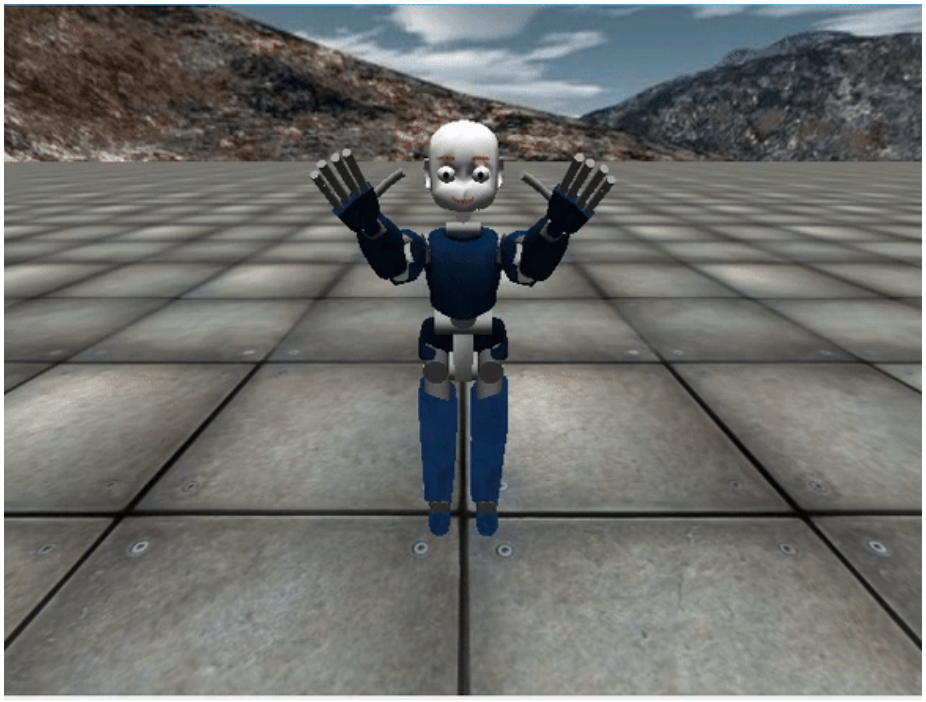
Motor control with YARP

Prerequisites

For this assignment, you just need to be aware of: - YARP ports - yarp::os::Bottle - yarp::os::Port - yarp::os::BufferedPort - YARP devices - some Yarp Motor Interfaces: - yarp::dev::IEncoders - yarp::dev::IControlMode - yarp::dev::IPositionControl

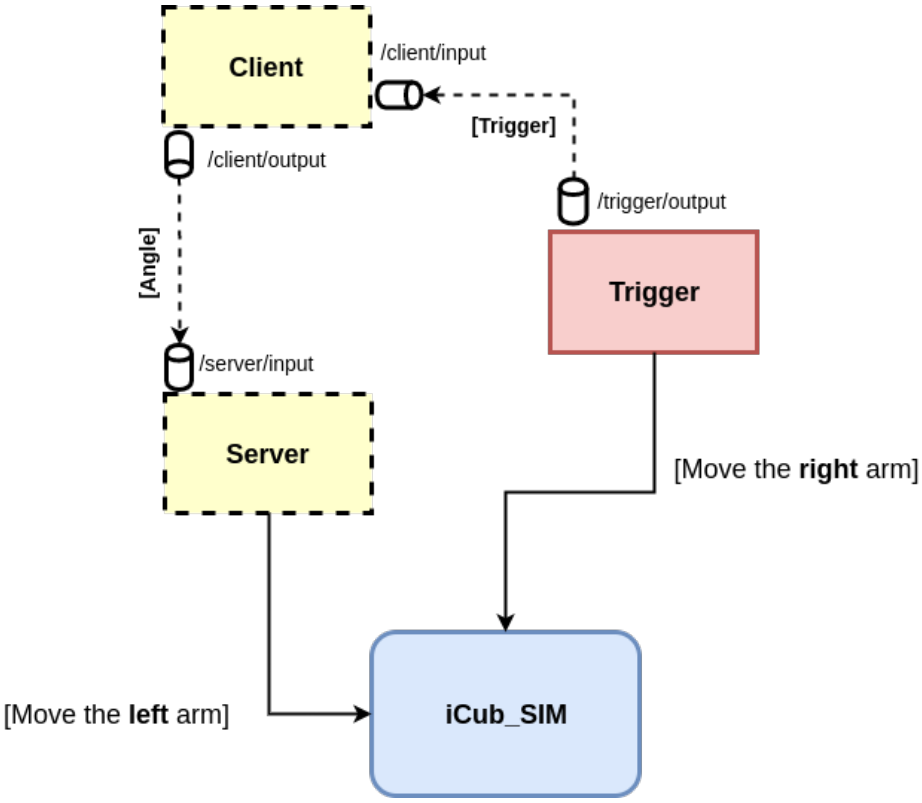
Assignment

Let's make iCub wave !



Scenario

During this assignment, you will be provided with the system described in the following diagram:



In this system, the **Trigger** moves the **joint 2** of *iCub_SIM* `right_arm` periodically, between **X** and **-X**, and sends to the **Client** a signal for waking it up.

Then, once the **Client** wakes up, it sends to the **Server** the angle **X** every period **T**.

The **Server** makes use of that angle as setpoint to move the **joint 2** of iCub_SIM `left_arm`.

TODO

You have to modify the code provided in order to let the **Server** move the `left_arm` with the same amplitude and period of the movement of the `right_arm`, which is in turn controlled by the **Trigger** module.

The **Trigger** is already provided in its final shape :ok_hand: **Don't touch it !** :knife:

In particular, to complete this assignment you have to: - Modify `client.cpp` implementing the comunication part for receiving the trigger and sending the angle to the **Server**. - Modify `server.cpp` implementing the comunication part

for receiving the angle by the **Client** and move the left arm. - Find the correct values of **X** and **T**.

By the way, just follow the `FILL IN THE CODE` comments inside the code and it will be **easy-peasy** :smirk:

You have not to be worried about the connections between ports, they have NOT to be established inside the modules

Once done, you can test your code in two ways:

1. **Automatically**: running the script **test.sh** in the **smoke-test** directory. This will give you an idea of how many points you might score.
2. **Manually**: running the *yarpmanager scripts* provided from within **app/scripts**. This will help you interact with your code.

Test the assignment using yarpmanager

First of, if you aim to test your code manually, then you have to follow these steps:

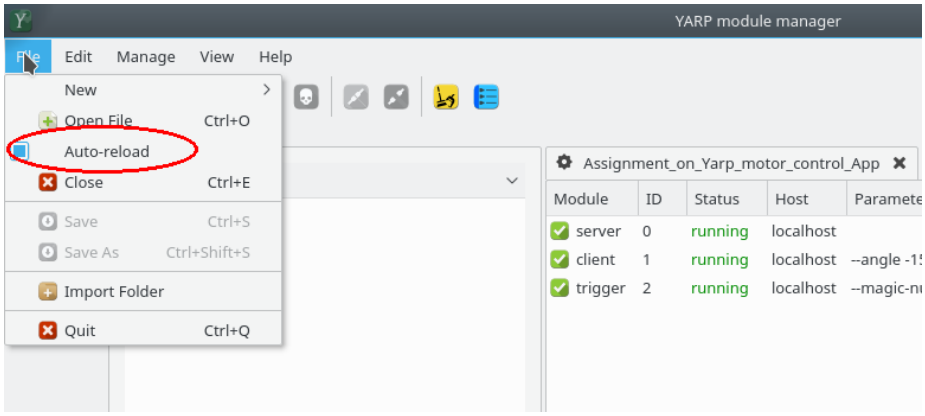
```
$ cd hw1
$ mkdir build && cd build
$ cmake ..
$ make
$ make install
```

Pay attention to the step `make uninstall && make install` every time you change your code; the risk is that you could be testing an out-of-date binary

Then, open `yarpmanager` from the terminal by simply running:

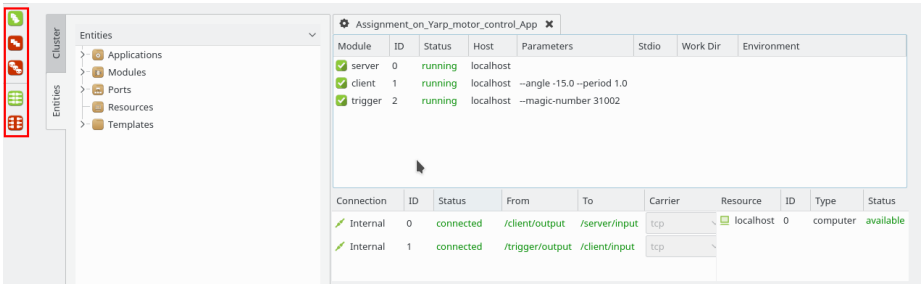
```
$ yarpmanager
```

(Hint: disable the automatic reload as is shown in the following figure)



In particular, from within the `yarpmanager` do: 1. Open `HW1_System` and press the **run all** button. 2. Open `HW1_App` and press the **run all** button again. 3. Once everything is running, press the **connect all** button. 4. To stop the modules, press the **stop all** button.

The following figure displays where the buttons are located:



Grading

The **smoke-test** evaluates the *similarity* between the movements of the right and left arm of iCub.

In particular, it records n values from the encoders of both arms storing them in the **A** and **B** vectors referenced below, respectively, to then calculate the similarity by means of the following formula:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

The **cosine similarity** is equal to **-1** if the movement of the arms are in counter-phase, equal to **1** if the movements are in phase. This measures how much the period **T** you chose is close to the one of the **Trigger** module. Moreover, the test checks that the ratio of the norms of **A** and **B** is ~ 1; it allows verifying if the value **X** you set is correct.

Here's the score map:

Requirements	Points
Open all requested ports	1
Move the arm	1
Cosine similarity > -1.0	1
Cosine similarity > -0.5	1
Cosine similarity > 0.0	2

Requirements	Points
Cosine similarity > 0.5	4
Norm ratio $== 1$	4
Maximum score	14