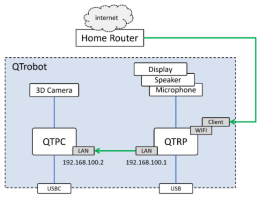
QTRobot Demo Description

Connecting to Network

Before anything is done, you need to ensure that the robot and the pc that you are working with are connected to the same network. The process of connecting the robot to the internet is described in LUXAI's documentation https://docs.luxai.com/docs/intro_code, but to try and simplify the process we will describe it here:

Image	Description
	The robot consists of two PC's: QTPC (the one you work on via mouse and keyboard) and QTRP. The network connection is part of the QTRP which you have to connect to within the terminal of the QTPC. To access the QTRP you have to follow these steps
``ssh developer@192.168.100.1 # or ssh developer@QTRP```	In the terminal, write the following code to access the QTRP: *ssh developer@192.168.100.1* The password to enter is *qtrobot*, all in small caps. (NOTE: You won't be able to see what you write, when entering the password)
``cd /etc/wpa_supplicant sudo nano wpa_supplicant-wlan0.conf```	Once you are on the QTRP, you have to enter the following file path and then access the *wpa_supplicant-wlan0* configuration file using the *sudo nano* command: *cd /etc/wpa_supplicant* *sudo nano wpa_supplicant-wlan0.conf*
``country=LU ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev update_config=1 network={ ssid="<your router SSID>" psk="<your router passphrase>" }```	If you don't already see this text within the configuration file, then write it in: *country=LU* *ctrl_interface=DIR=/var/run/wpa_supplicant* *GROUP=netdev* *update_config=1* *network={* *ssid="<your router SSID>"* *psk="<your router passphrase>"* *}* The **ONLY** thing **YOU** have to specify is *"your router SSID"* (Name of the network you want to connect to) and *"your router*

	passphrases" (Password to the network) Once written save the file with CTRL+S and then exit with CTRL+X
`sudo systemctl restart qt_wlan0_client.service`	Write this in the console to restart the network, so that the PC can connect to the new network: *sudo systemctl restart qt_wlan0_client.service*
`ping www.google.com`	To see if you are connected you can try to ping google, by writing the following: *ping www.google.com* You should see messages printed in the console that indicate it is pinging google. Use CTRL+C to make it stop pinging
`sudo systemctl enable qt_wlan0_client.service` `sudo systemctl disable qt_wlan0_ap.service`	To make the connection permanent, write the following lines in the command line: *sudo systemctl enable qt_wlan0_client.service* *sudo systemctl disable qt_wlan0_ap.service* You are now ready to go back to the QTPC to continue your work. (you can close the console and open it again, or make a new tab in the console to return to the QTPC. Alternatively you can write *ssh developer@QTPC* and again write the password *qtrobot*)

Socket Streaming - Service & Client

Choosing amount of participants

```
44
45      participant_Amount = 4
46
```

On line 45 there is a variable called *participant_Amount*. This variable is used for setting how many participants will be participating in the whole session with the robot.

This is used to tell the robot which gestures to run when pointing to participants when letting them speak, and also of course to set the amount of rounds that the robot will listen for participant ideas.

Image	Description
	The first round initiated is the pitch round for the first participant, in which they will present their idea. On line 146 the robot will orient its head towards the first participant (starting from its left, going right). The head orientation in each round is determined by the amount of participants set in the *participant_Amount* variable. All commands for the robot is sent via the *set_send_message* method, which makes sure that the message sent is in the correct format for the robot's system to interpret. Communication with the robot is done via ROS services, which makes us unable to run code on the robot in parallel. Instead code to be run is queued, which can potentially mess the structure of the session up. This is why a while loop on line 147 is initiated, until the bool *service_in_progress* becomes false. This while loop is used each time a ROS service is initiated.
	(Line 161-168) This is the set-up for the start pit. There's numerous pitches throughout for each participant, but each pitch normally is set up with a *duration*, which accounts for how long the mic is recording during the pitch. a *currentParticipantNumber* which accounts for participant number. A *messageList* that adds pitch to an overall list when done and lastly a *roundDoneBool* to account for when said participant is done talking
	During each pitch round in both sessions the button "*SPACEBAR*" can be clicked to suspend recording early. This is done in case a pitch round does not need the entire duration time - so users do not sit for an extended amount of time (contrary to the function name it is "SPACEBAR" that needs to be clicked & not "enter")

	Then for the pitching round each participant also has a function where they get asked for a duration, where *QuestionType* also is defined to make sure not all participants get different question types. This pitch round call also be suspended earlier with SPACEBAR if needed