Choregraphe (ZeMo)
Zemin XU and Mohamed ABDUL GAFOOR

Assignment:

The purpose of this task is to build up a robot speaking system with body language and speech text and speech sound in Choregraphe.

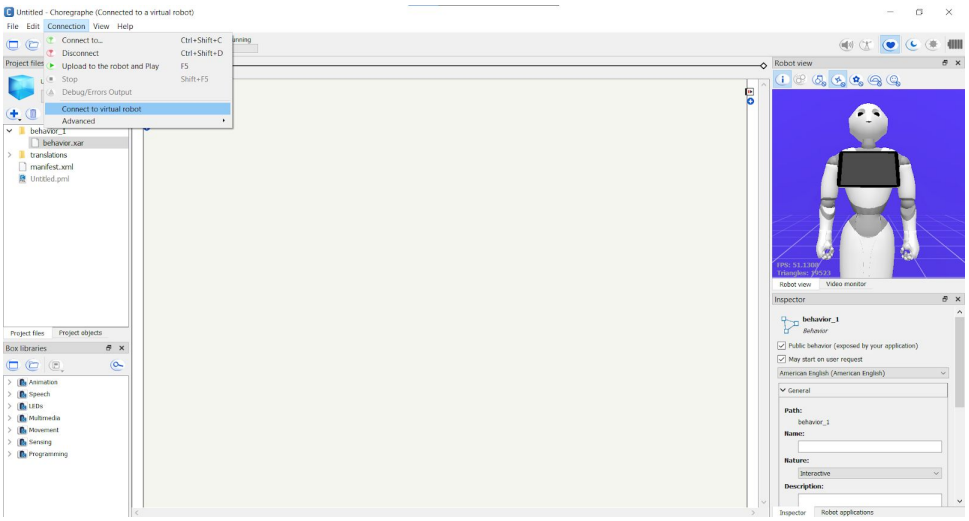
Collaboration:

The way we have collaborated is to create a project with Choregraphe on a local machine. By using Zoom, we were able to collaborate and finish the homework together. We have taken screenshots to show the steps to accomplish the task and finally, we wrote the report by using Google Docs.

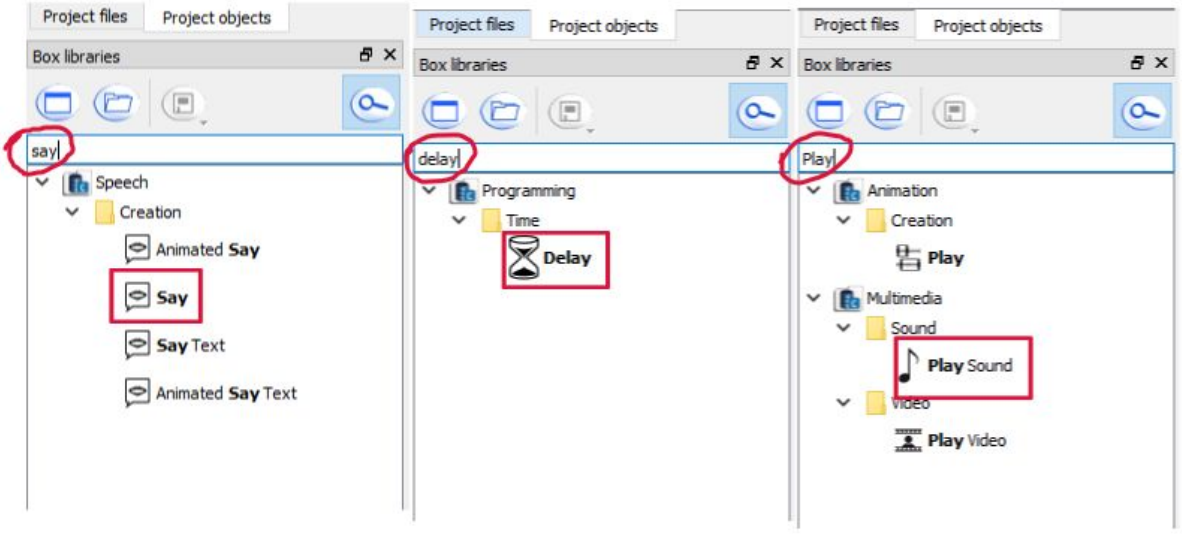
How we did it:

Step 1:

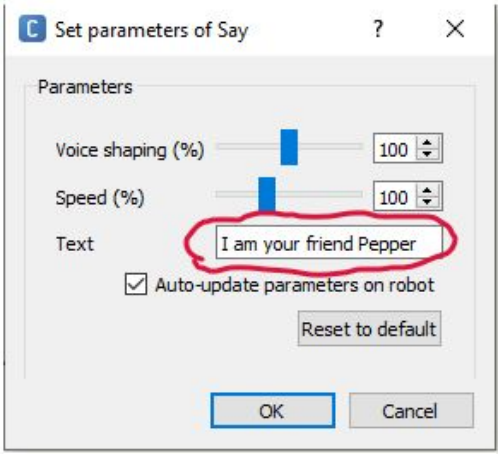
Firstly we opened Choregraphe App and connected to the virtual robot '**NAO H21**'. After that, we selected the **'Say'** node, **'Delay'** node and **'Play Sound'** node on the Box library. In the **'Say'** node, we put in the words that Pepper will speak. (Here we test with **'NAO H21'** but finally we changed it to **'Pepper'**). Then go to **Connection** and **Connect to virtual robot** in menu.



Add the following nodes in order to create **text**, **delay** and **sounds** in the virtual object. The nodes can be added easy by searching the key words in the search windows (show filters). The following screenshot shows how this can be accomplished.

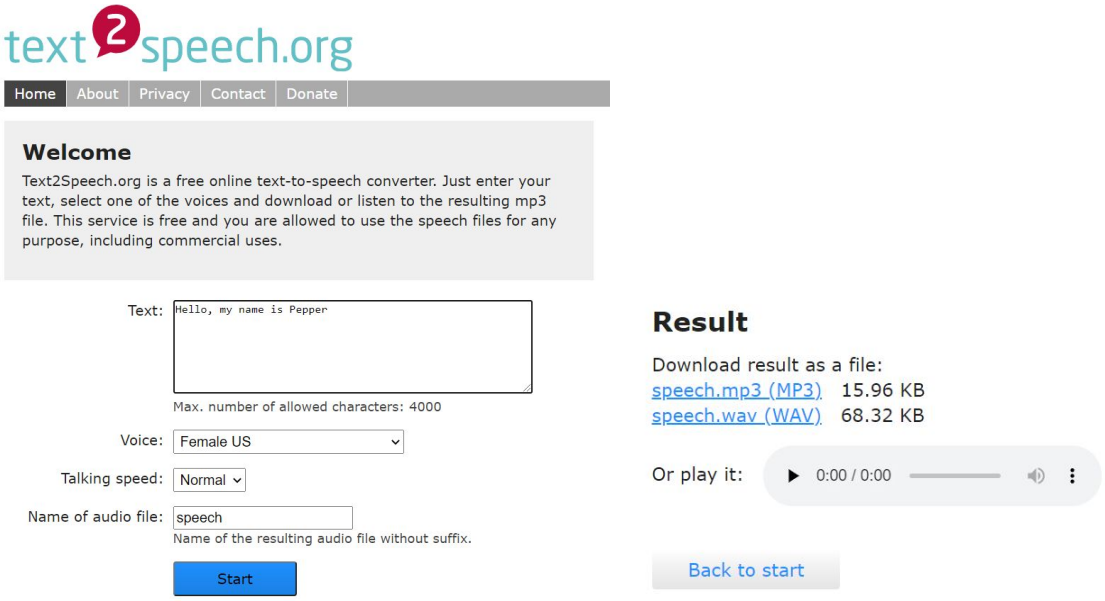


If we want to add the text to the **'say'** node, we can simply **right click** and **'set parameters'** of say. The following screenshot shows how this can be achieved.

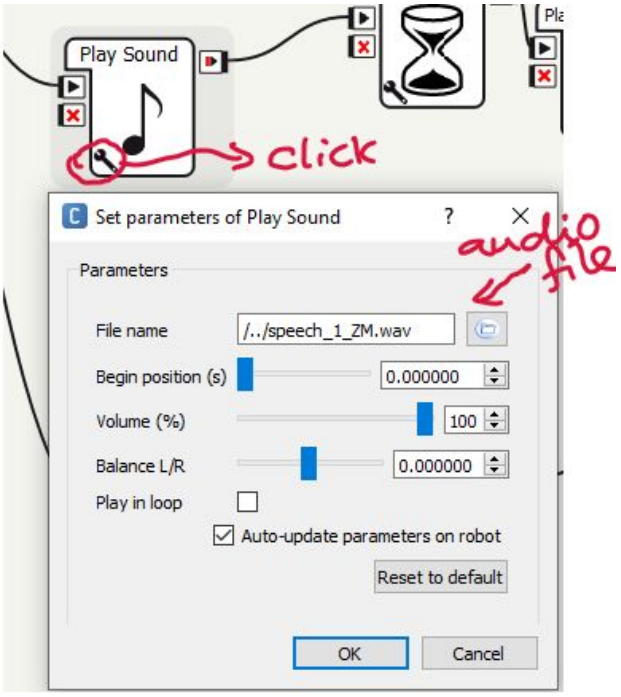


Step 2:

In order to listen to Pepper speaking, we added several **'Play Sound'** nodes in the Box Library and put the **.wav** file as parameters. Here we used **'text2speech'** to generate sound files. We only need to type in the words to be converted as voice and download the generated file. The link of **'text2speech'** is provided in reference.

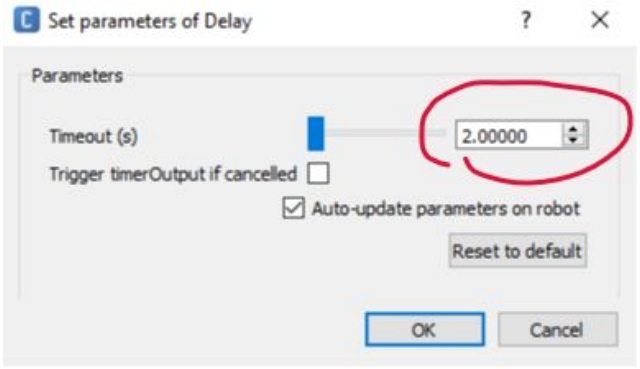


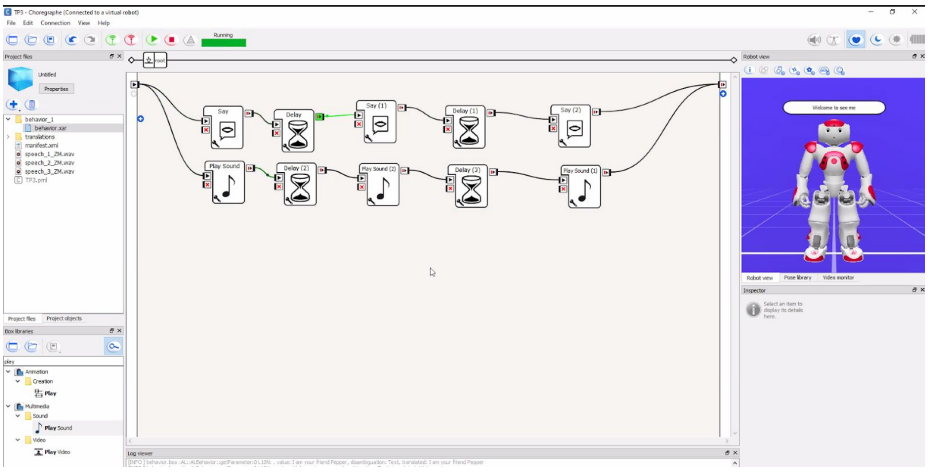
The following figure shows how an audio file can be added to the Play Sound node.



Step 3:

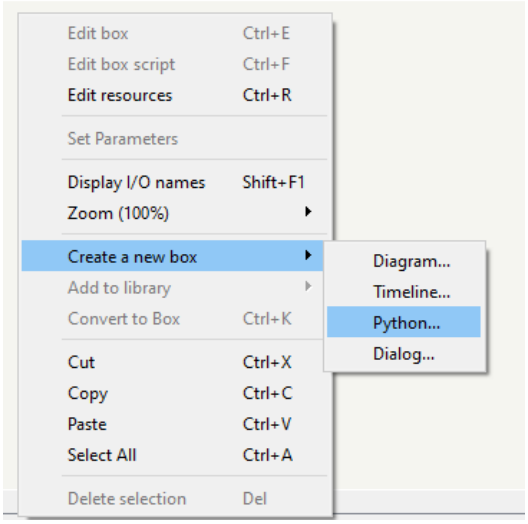
We created as well the **‘Delay’** node in order to have the text and sound synchronized and to imitate the natural way of speaking. We tuned the time in each Delay node to achieve it. Right click on the Delay node and set parameters. Where we can set the time of delay after each text/audio.





Step 4:

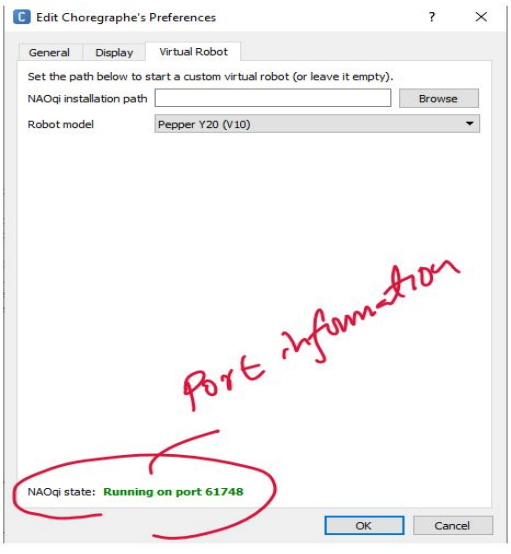
At this step, we would create animations that are specified in the provided json file. We created a python script by creating a new box with right click. The below image shows the procedure to create a Python script.



In the **__init__ method,** add the following lines. This **method** is called when an object is created from **MyClass.** This method helps the class to **initialize** the attributes of the class. First we have to create a session using **qi.Session()**, then add the **tcp IP** and **port**. The default IP is **"127.0.0.1"**.

```
def __init__(self):
    GeneratedClass.__init__(self)
    self.tts = ALProxy('ALTextToSpeech', "127.0.0.1", 61487)
    self.motion = ALProxy("ALMotion", "127.0.0.1", 61487)
    self.postureProxy = ALProxy("ALRobotPosture", "127.0.0.1", 61487)
    #self.ALMotion = ALProxy("ALRobotPosture", "127.0.0.1", 57199)
    self.session = qi.Session()
    self.session.connect("tcp://127.0.0.1:61487")
    self.motion_service  = self.session.service("ALMotion")
    self.motion_service.setStiffnesses("Head", 1.0)
    self.motion_service.setStiffnesses("Body", 1.0)
```

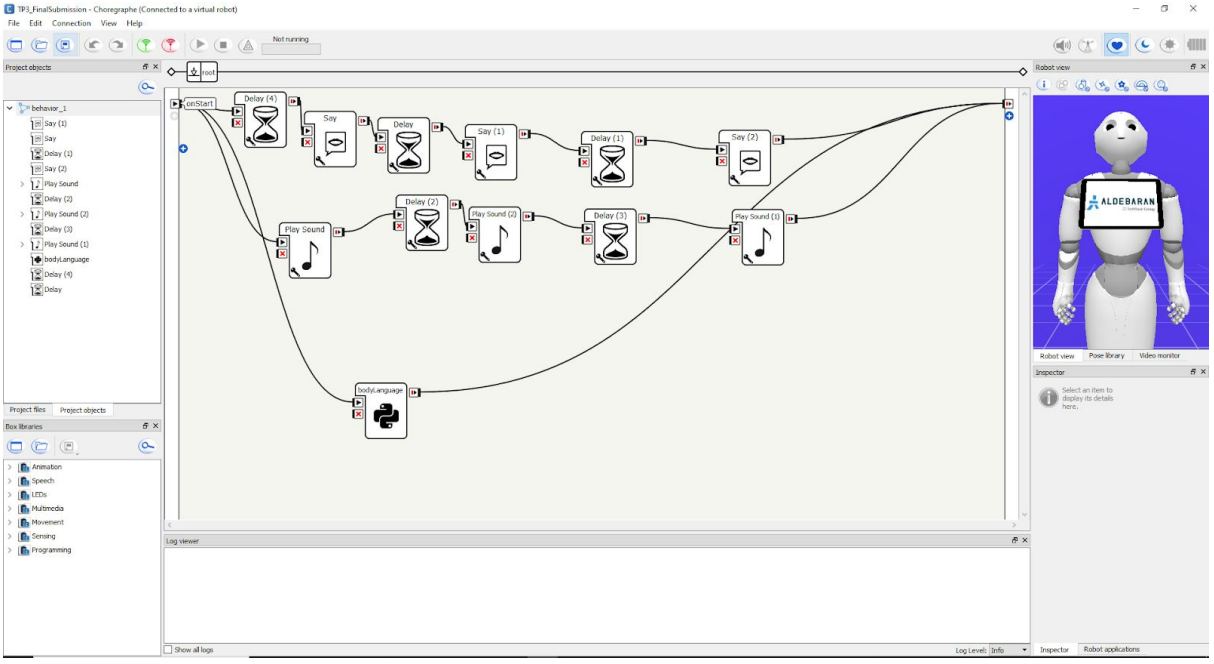
The port information can be obtained by go to Edit → Preference → Virtual Robot.



Now in the **def onInput_onStart(self)** method, add the following line to create the necessary animation. We must set the path to the json file and load the json file and store it into the robotpose. If you want to run this project in your local machine, don't forget to change the path of json file in the code below. Make sure to set the correct port information.

```
    def onInput_onStart(self):

        path = 'C:/Users/itube/Documents/SoftRobotics/assignment/TP3/robot_action_8_126.json'
        with open(path, "r") as f:
            robotpose = json.load(f)
            timeLists_24hz_base = [1.0, 1.04,1.08, 1.12, 1.17, 1.21, 1.25, 1.29, 1.33, 1.38, 1.42, 1.46, 1.50, 1.54, 1.58,
1.62, 1.67, 1.71, 1.75, 1.77, 1.83, 1.88, 1.92, 1.96, 2.0, 2.04, 2.08, 2.12, 2.17, 2.21, 2.25, 2.29, 2.33, 2.38, 2.42, 2.46,
2.5, 2.54, 2.58, 2.62, 2.67, 2.71, 2.75, 2.79, 2.83, 2.88, 2.92, 2.96, 3.0, 3.04, 3.08, 3.12, 3.17, 3.21, 3.25, 3.29, 3.33,
3.38, 3.42, 3.46, 3.5, 3.54, 3.58, 3.62, 3.67, 3.71, 3.75, 3.79, 3.83, 3.88, 3.92, 3.96, 4.0, 4.04, 4.08, 4.12, 4.17, 4.21,
4.25, 4.29, 4.33, 4.38, 4.42, 4.46, 4.5, 4.54, 4.58, 4.62, 4.67, 4.71, 4.75, 4.79, 4.83, 4.88, 4.92, 4.96, 5.0, 5.04, 5.08,
5.12, 5.17, 5.21, 5.25, 5.29, 5.33, 5.38, 5.42, 5.46, 5.5, 5.54, 5.58, 5.62, 5.67, 5.71, 5.75, 5.79, 5.83, 5.88, 5.92, 5.96,
6.0, 6.04, 6.08, 6.12, 6.17, 6.21]

            timeLists_24hz = [timeLists_24hz_base, timeLists_24hz_base, timeLists_24hz_base, timeLists_24hz_base,
timeLists_24hz_base, timeLists_24hz_base, timeLists_24hz_base, timeLists_24hz_base, timeLists_24hz_base]

            names = ['RShoulderRoll', 'LShoulderRoll', 'RShoulderPitch', 'LShoulderPitch', 'RElbowRoll', 'LElbowRoll',
'RElbowYaw', 'LElbowYaw']

            angleLists = robotpose
            timeLists = timeLists_24hz
            isAbsolute = True

            #self.ALMotion.angleInterpolation(names, angleLists, timeLists, isAbsolute)
            self.motion_service.angleInterpolation(names, angleLists, timeLists, isAbsolute)
```

The following screenshot shows the final implementation of the Choregraphe project.



Reference:

1. Text 2 Speech
2. Choregraphe application: NAO6 Downloads - Linux | SoftBank Robotics Developer Center
3. Joint control API — Aldebaran 2.5.11.14a documentation
4. http://doc.aldebaran.com/2-5/naoqi/motion/control-joint-api.html#ALMotionProxy::angleInterpolation__AL::ALValueCR.AL::ALValueCR.AL::ALValueCR.bCR