**Student Name: Student Id:**

**Lab Instructions:**Please show your work to the instructor present in the lab before submitting.

**Submission Due**: End of laboratory class, submit the file on Moodle at least 10 minutes before the end of laboratory class.

**Total Marks** = 10 marks

Marks will be given only to students who attend and participate during 2-hour laboratory class. Submission on Moodle is mandatory as an evidence of participation.

**Learning Outcomes:**

|  |  |
| --- | --- |
| LO1 | Define Robotics, and Robotics Process Automation. |

**Marking Criteria:**

|  |  |  |
| --- | --- | --- |
| **Task Details** | **Submission Requirements** | **Marks** |
| **Task 1-6:** Designing and Testing Pepper Robot Use Cases | Same **Word file** containing the source code/ model and output. | 10 |

**Lab 6 – Robotics and Robotic Process Automation (Part 2)**

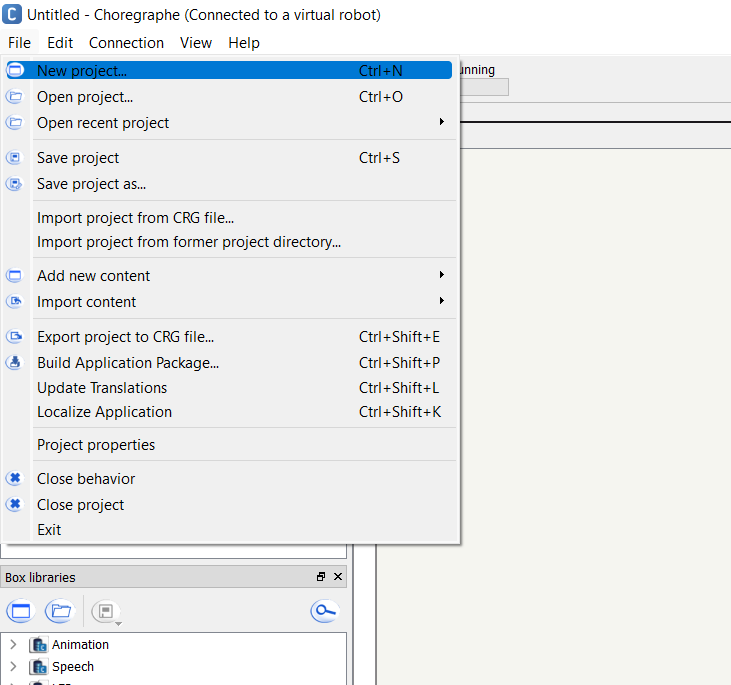
## **Task 1:** Making Pepper Speak

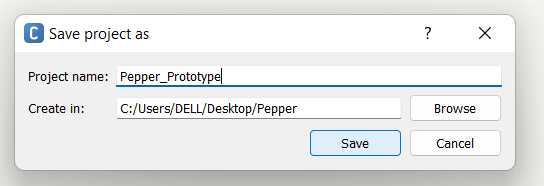
**Objective:**

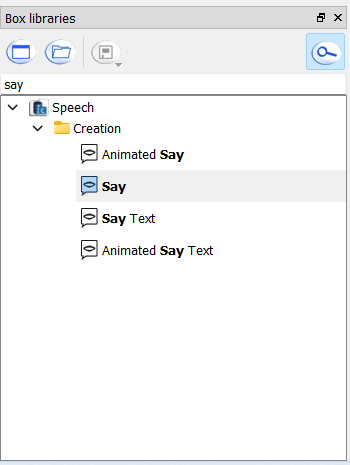
Learn how to make Pepper say a simple phrase using Choregraphe.

**Steps:**

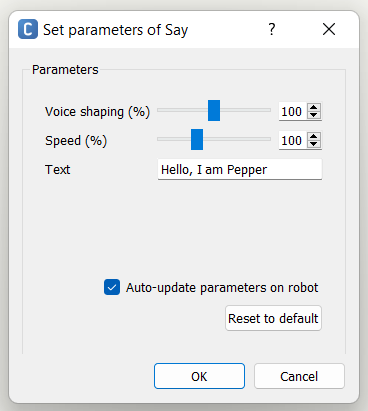
1. Open Choregraphe and create a new project (File > New Project).



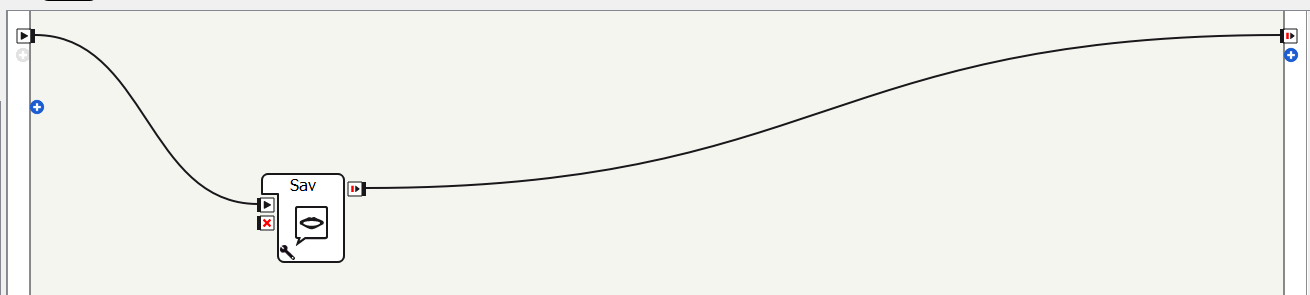
1. Enter a simple name, for example, “Pepper\_Prototype.”
2. Save the project in a dedicated folder. 
3. Locate the **Say Box** in the Box Libraries Panel.



1. Drag and drop the Say Box into the Flow Diagram Panel.
2. Double-click the Say Box and enter the text: *"Hello! I am Pepper."*



1. Connect the **onStart** input to the Say Box and connect the **onStopped** output to the Stop Box.



1. Click **Play** to test Pepper’s speech in simulation.
2. Deploy it to Pepper and observe the result.

**Student Code and Output:**Please wait for faculty instructions before connecting to the robot. Once instructed, present your results to the lecturer/lab faculty.

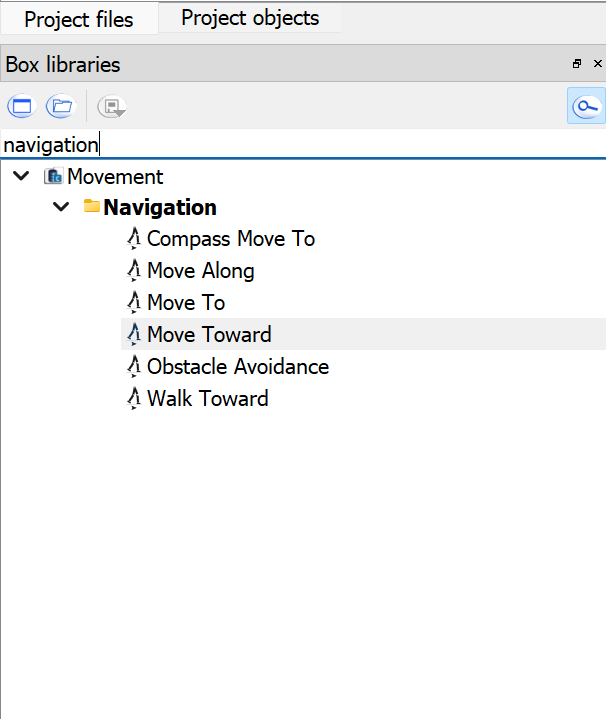
## **Task 2 : Making Pepper Move**

**Objective:**

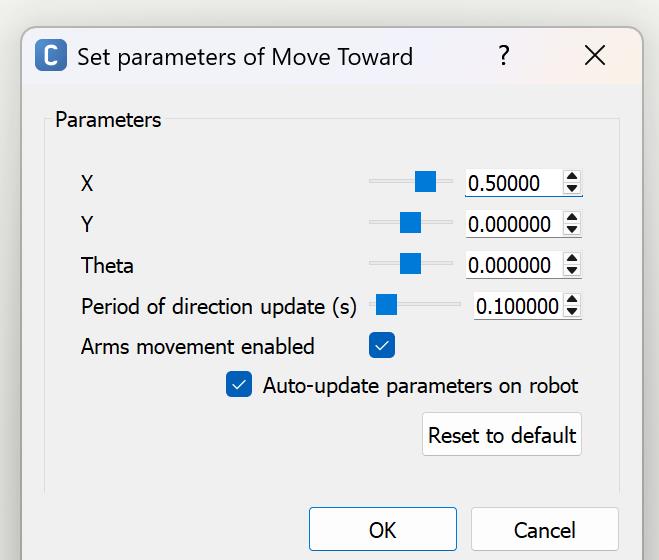
Learn how to program Pepper to perform basic movements.

**Steps:**

1. Open a new project in Choregraphe.
2. Find the **Move Toward Box** in the Box Libraries Panel.



1. Drag it into the Flow Diagram Panel.
2. Set movement parameters (e.g., move forward 0.5 meters).



1. Connect the Start Box to the Move Toward Box.



1. Click **Play** and test the movement on Pepper.

**Student Code and Output:**Please wait for faculty instructions before connecting to the robot. Once instructed, present your results to the lecturer/lab faculty.

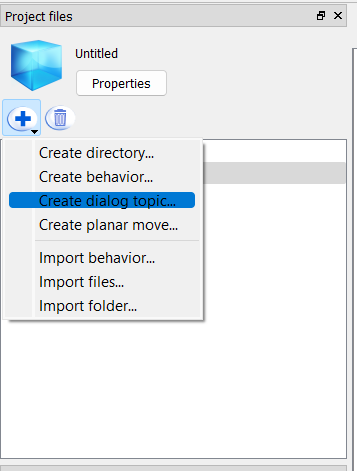
## **Task 3: Simple Interaction with Pepper**

**Objective:**

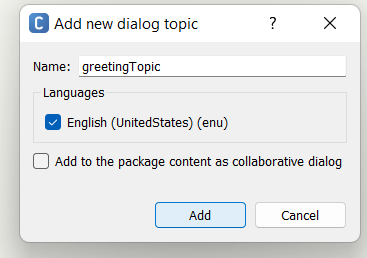
Create a basic interactive conversation using **Dialog Boxes**.

**Steps:**

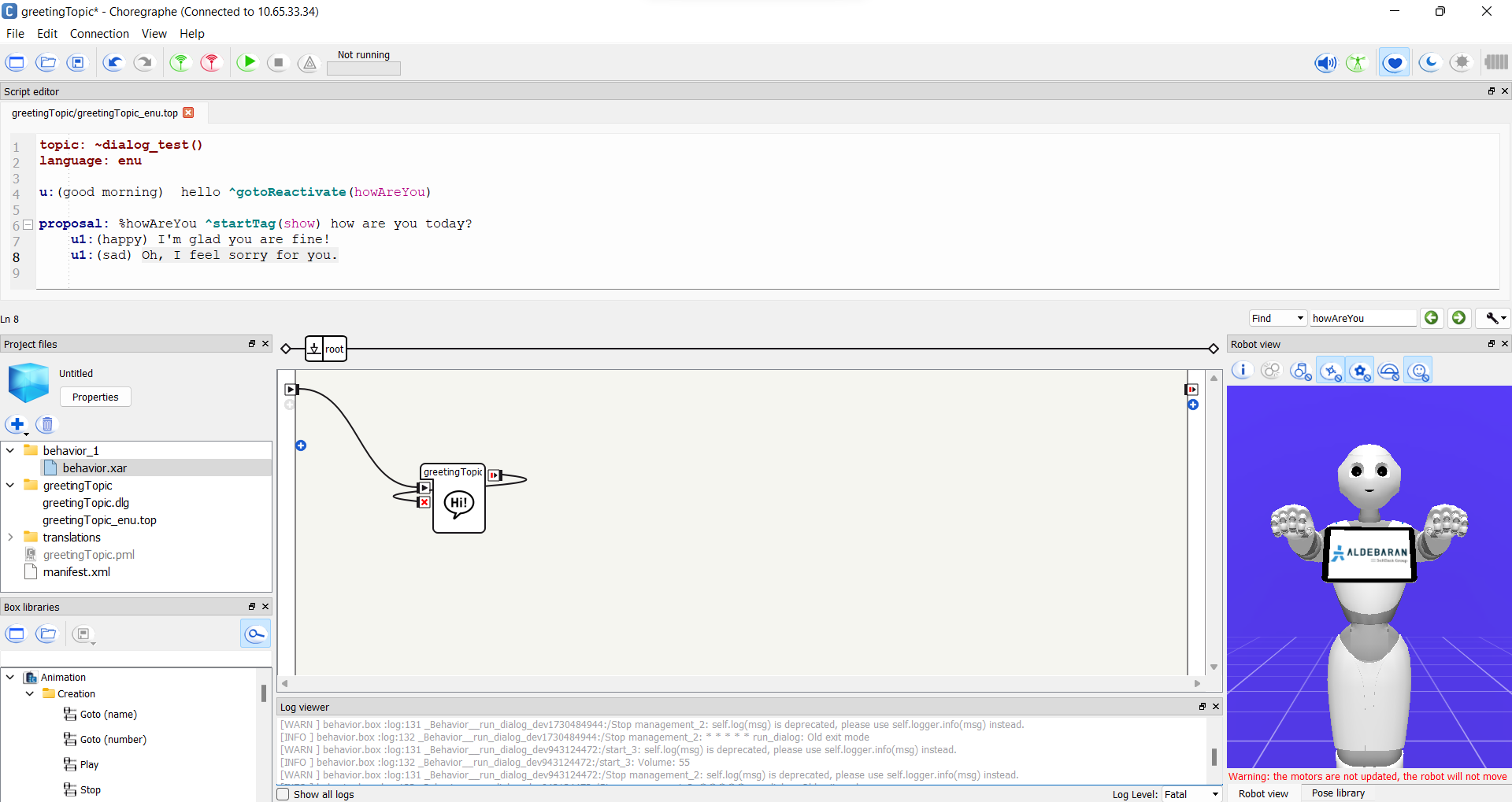
1. Open Choregraphe and go to the dialog section.
2. Click on “Create dialog topic.”



1. Enter a topic name without spaces (for example, “greetingTopic”).
2. Select the language (e.g., English) and click Add. This generates two files: a .dlg file (the dialog structure) and a .top file (the rules and tags).



1. Locate the generated .dlg file in the project files. This file holds the basic structure of the dialog. Also, find the .top file. This file is where you define the detailed rules for how Pepper will respond.
2. Open the .dlg file in built-in editor. This file might contain a basic template. You can leave it as is for now, as most of your detailed rules will be in the .top file.
3. Open the .top file for your “greetingTopic”. This is where you define the rules for Pepper’s responses.
4. Now, we will define a basic greeting rule by write a rule that listens for a greeting using qiChat. “u: (Hello) Hi there! How can I help you?” means whenever user says “Hello”, pepper will respond with ‘Hi there! How can I help you?’. Once .top file is edited, save the changes.
5. Add the dialog box to the behavior flow.
6. Click Play to test the dialog.



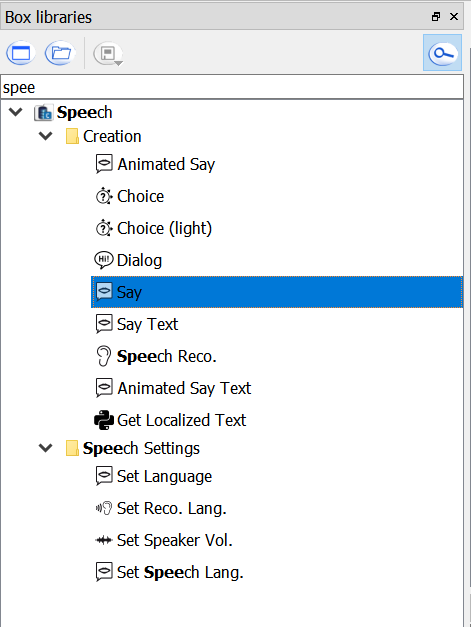
**Student Code and Output:**Please wait for faculty instructions before connecting to the robot. Once instructed, present your results to the lecturer/lab faculty.

## **Task4:** Making Pepper Speak with Different Voices

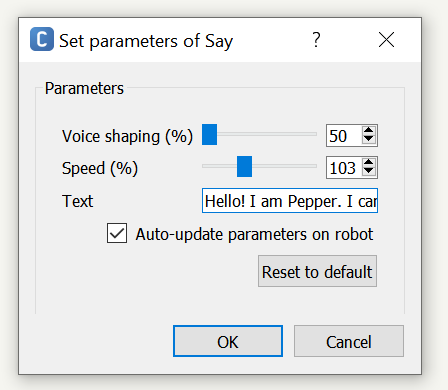
**Objective:**  
Learn how to change Pepper’s voice settings while making it speak.

**Steps:**

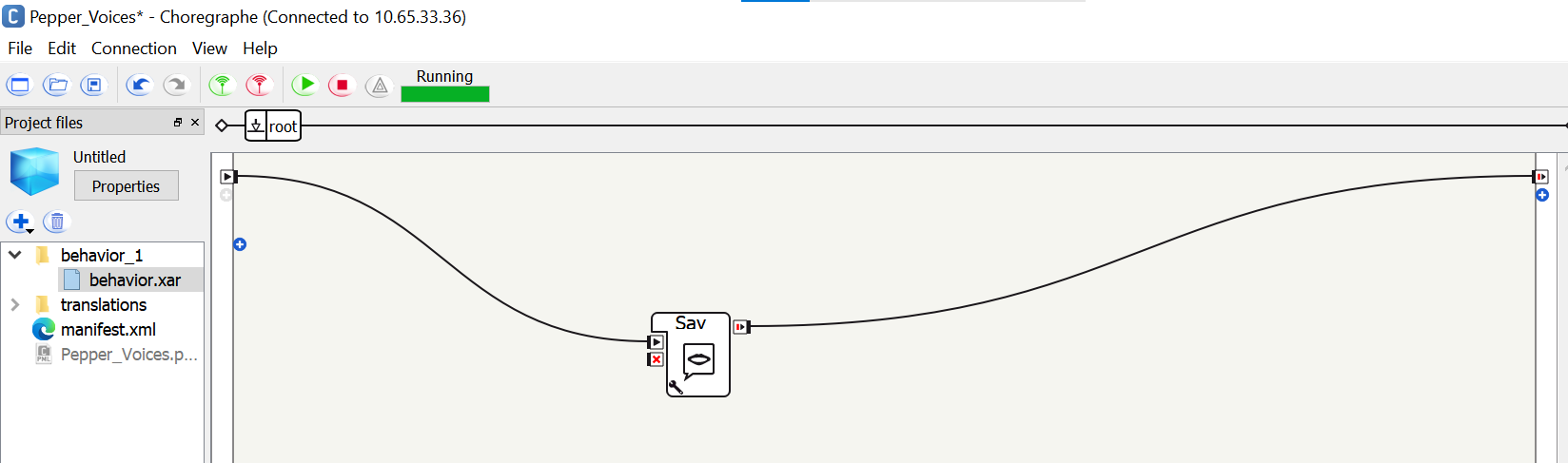
1. Open Choregraphe and create a new project (File > New Project).
2. Name it “Pepper\_Voices.”
3. Locate the **Say Box** in the **Box Libraries Panel** under Speech and drag it into the **Flow Diagram Panel**.



1. Right-click the Say box and click ‘Set Parameter’. Enter the text: "Hello! I am Pepper. I can change my voice."
2. Adjust the speed and voice shaping and test with different values.



1. Connect the **onStart** input to the **Say Box** and connect the **onStopped** output to the **Stop Box**.



1. Click **Play** to test the voice in simulation.
2. Deploy it to Pepper and listen to the difference in voice settings.

**Student Code and Output:**Please wait for faculty instructions before connecting to the robot. Once instructed, present your results to the lecturer/lab faculty.

**Task5: Making Pepper Perform a Wave and Speak**

**Objective:**  
Learn how to make Pepper combine gestures and speech using Choregraphe.

**Instructions:**  
Using Choregraphe, create a simple behavior where Pepper:

1. **Waves its hand**
2. **Says a greeting message**, such as “Hello! Nice to meet you!”

**Hint:** Set animation to ‘Stand/Gestures/Hey\_1’ which is the wave animation of the pepper.

**Student Code and Output:**Please wait for faculty instructions before connecting to the robot. Once instructed, present your results to the lecturer/lab faculty.

**Task6: Making Pepper Detect and Respond to a Face**

**Objective:**  
Learn how to make Pepper recognize a human face and respond using Choregraphe.

**Instructions:**  
Using Choregraphe, design a behavior where Pepper:

1. **Detects a human face** using the face detection module.
2. **Responds** by saying a phrase like “I see someone!”

**Student Code and Output:**Please wait for faculty instructions before connecting to the robot. Once instructed, present your results to the lecturer/lab faculty.

## **Task 7: Making Pepper Spin**

**Objective:**

Learn how to program Pepper to perform specific movements.

**Instructions:**

Using Choregraphe, design a behavior where Pepper moves in spinning motion.

**Hint:** Set Theta to 1 which helps pepper to rotate in radians.

**Student Code and Output:**Please wait for faculty instructions before connecting to the robot. Once instructed, present your results to the lecturer/lab faculty.

## **Task 8: Making Pepper Spin**

**Objective:**

Learn how to make Pepper perform a relaxing Tai Chi Chuan dance using the built-in behavior box in Choregraphe.

**Instructions:**

Using Choregraphe, design a behavior where:

1. **Pepper says, *" I will show you my Tai Chi Chuan dance!"* before starting the dance.**
2. **Performs the Tai Chi Chuan animation.**

**Student Code and Output:**Please wait for faculty instructions before connecting to the robot. Once instructed, present your results to the lecturer/lab faculty.

## **Task 9: Making Pepper Follow a Person**

**Objective:**  
Learn how to use Pepper’s tracking abilities.

**Steps:**

1. Open Choregraphe and create a new project called **“Pepper\_Follow”**.
2. Locate the Python Script in the **Box Libraries Panel**.
3. Drag **the python script** into the **Flow Diagram Panel**.
4. Edir script and paste the following code in it:

from naoqi import ALProxy

class MyClass(GeneratedClass):

def \_\_init\_\_(self):

GeneratedClass.\_\_init\_\_(self)

self.PEPPER\_IP = "10.65.33.36"

self.PORT = 9559

self.tracker = None

self.motion = None

self.tts = None

def onLoad(self):

try:

self.tracker = ALProxy("ALTracker", self.PEPPER\_IP, self.PORT)

self.motion = ALProxy("ALMotion", self.PEPPER\_IP, self.PORT)

self.tts = ALProxy("ALTextToSpeech", self.PEPPER\_IP, self.PORT)

except Exception as e:

print("Error during proxy creation:", e)

def onUnload(self):

if self.tracker:

try:

self.tracker.stopTracker()

self.tracker.unregisterAllTargets()

except Exception as e:

print("Error during tracker cleanup:", e)

def onInput\_onStart(self):

try:

if self.tts:

self.tts.say("I will now start following you.")

if self.tracker:

self.tracker.registerTarget("People", 0.5)

self.tracker.setMode("Move")

self.tracker.track("People")

except Exception as e:

print("Error during tracking start:", e)

# self.onStopped() # Uncomment if needed to continue flow after start

def onInput\_onStop(self):

self.onUnload()

self.onStopped()

The onload() function connects to the al modules. ALTracker allows Pepper to track and follow objects. ALMotion controls Pepper's movements like walking, turning, etc. ‘self.tts.say("I will now start following you.")'. self.tracker.registerTarget("People", 0.5) tells Pepper to track people. ‘self.tracker.setMode("Move")’ sets Pepper's tracking mode to Move, so it will actually walk and follow the person and ‘self.tracker.track("People")’ starts the tracking behavior for the target "People".

1. Connect the **onStart** input to the Python ScriptBox to the start box, then connect the **onStopped** of the python script box to the end.
2. Click **Play** to run the behavior of pepper.
3. Pepper should say: "I will now start following you."
4. Then it will start tracking and following any nearby person it detects.
5. Click the "Stop" button in Choregraph to stop the behavior.

**Student Code and Output:**Please wait for faculty instructions before connecting to the robot. Once instructed, present your results to the lecturer/lab faculty.

**Submission Instructions:**

1. Submit your answers in this word file by renaming it in the format **“INTROAI\_** **AI2010\_Lab\_6\_08\_04\_2025\_name”** and uploading on **Moodle** in the appropriate submission link.
2. Also submit source code (.ipynb) files for Lab Tasks along with this word file.

**Please conform to the naming convention of the file.**

**END OF LABORATORY**