CSSR for 🌍

Culturally Sensitive Social Robotics
for Africa

D3.3 Software Installation Manual

Due date: **1/10/2023**
Submission Date: **26/07/2023**

Start date of project: **01/07/2023**

Duration: **36 months**

Lead organisation for this deliverable: **Carnegie Mellon University Africa**

Responsible Person: **CSSR4Africa Team**

Revision: **1.0**

Contents

Executive Summary

Deliverable D3.3 serves as a comprehensive installation manual for the Culturally Sensitive Social Robotics for Africa (CSSR4Africa) project. The manual will be continually updated as a living document to ensure it reflects the current capabilities of the system. The objective of this task is to document the process of installing and executing the required software components to instantiate the CSSR4Africa system. It provides step-by-step instructions for setting up the development environment and controlling the Pepper robot in both physical and simulated environments.

1 Introduction

This Deliverable, D3.3: Software Installation Manual documents the process for the installation and execution of the software required to instantiate all or part of the CSSR4Africa system and run the unit, integration, and system tests. This installation manual assumes the system has a native Ubuntu 18.04 operating system. It was tested on a computer device with a storage of 512GB, 16GB of RAM, and 4 Intel CPU cores.

The installation is divided into 2 main sections. Section 2, is for setting up the development environment for the physical robot and the Gazebo simulator. Section 2 first goes through a prerequisite setup that is needed for both the physical robot and Gazebo simulator environments and proceeds with corresponding installation steps. Section 3, will discuss different sensors and actuators unit tests that can be used to ensure that the installations steps in section 2 were successfully completed.

2 Setting up the Development Environment

This section provides step-by-step instructions for installing a set of software packages required to control the physical Pepper robot and the robot in the Gazebo simulator. Open a new terminal *(ctrl + shift + t)* and type the following commands carefully.

2.1 Prerequisite

The prerequisite of setting up the development environment for both the physical robot and the simulator is installing ROS Melodic. The following instructions are followed to install ROS melodic.

1. Setup the computer to accept software from packages.ros.org.

   ```
   sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(
       lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
   ```

2. Install curl.

   ```
   sudo apt install curl
   ```

3. Setup your keys.

   ```
   curl -s https://raw.githubusercontent.com/ros/rosdistro/
   master/ros.asc | sudo apt-key add -
   ```

4. Update Debian package index.

   ```
   sudo apt update
   ```

5. Install ROS Melodic with the default configurations.

   ```
   sudo apt install ros-melodic-desktop-full
   ```

6. Make ROS environment variables automatically added every time a new shell is launched.

   ```
   echo "source /opt/ros/melodic/setup.bash" >> $HOME/.bashrc
   source $HOME/.bashrc
   ```

2.2   Setting up the Development Environment for the Physical Robot

2.2.1   Installing and Configuring the C++ NAOqi SDK

NAOqi serves as the main software responsible for running and controlling the Pepper robot. To control Pepper with ROS, NAOqi needs to be installed, and a suitable ROS driver must be found to enable communication between the two software platforms. The NAOqi Framework, being a cross-language programming framework utilized for programming Pepper, allows programming the robot using both C++ and Python languages. For the development of the bridge, the NAOqi C++ SDK needs to be installed, and the following steps detail the installation process.

1. Check compiler version.
   A GCC compiler version 4.8.2 or higher is required. On most Ubuntu distributions, it is installed by default. Check the GCC version by typing the following command:

   ```
   gcc -v
   ```

2. Install pip.
   If you don't have pip installed on your machine, please run the code below.

   ```
   sudo apt install python-pip
   ```

3. Install qbuild.
   qbuild is used to create a cross-platform executable.

   ```
   pip install qibuild --user
   ```

4. Update the path environment variable.

   ```
   # to load qBuild when your shell starts
   PATH=$PATH:$HOME/.local/bin
   ```

5. Configure qBuild.

   ```
   qibuild config --wizard
   # Input 1 and hit enter (1 Unix Makefiles (default))
   # Input 1 and enter (1 None (default))
   ```

6. Make a directory to work in.

   ```
   mkdir myworktree && cd myworktree
   ```

7. Download the Naoqi-SDK.

   ```
   wget -P $HOME/Downloads/ https://community-static.
   aldebaran.com/resources/2.5.10/NAOqi%20SDK/naoqi-sdk
   -2.5.7.1-linux64.tar.gz
   ```

8. Go to the download folder and extract the downloaded file.

   ```
   cd $HOME/Downloads && tar -xvzf naoqi-sdk-2.5.7.1-
   linux64.tar.gz
   ```

9. Create a toolchain.

```
qitoolchain create mytoolchain $HOME/Dowloads/naoqi-sdk-
2.5.7.1-linux64/toolchain.xml
```

10. Go to the myworktree directory and configure the SDK.

```
cd $HOME/myworktree && qibuild add-config myconfig -t
mytoolchain --default
```

2.2.2 Installing NAOqi Driver and ROS Packages

The NAOqi driver is a module that provides some bridge capabilities. It publishes sensory data, and the robot position (Pepper in our case) and enables ROS to call part of the NAOqi API. The easiest way is to install it with the apt package tool used to install software on Ubuntu. Installing the driver through the apt will not provide all the packages needed to communicate with the robot through ROS. For example, the **naoqi_dcm_driver**, is the package controlling the robot's actuators. Thus, the remaining packages were compiled from their sources through catkin make. This requires cloning the official sources code from GitHub. Below are the steps to follow.

```
1  # Install the NAOqi driver
2  sudo apt-get install ros-.*-naoqi-driver
3
4  # Create ROS workspace
5  mkdir -p $HOME/workspace/ros/src
6
7  # Move to workspace directory
8  cd $HOME/workspace/ros/src
9
10 # Install Git
11 sudo apt install git
12
13 # Clone NAOqi DCM driver repository
14 git clone https://github.com/ros-naoqi/naoqi_dcm_driver.git
15
16 # Clone Pepper DCM driver repository
17 git clone https://github.com/ros-naoqi/pepper_dcm_robot.git
18
19 # Clone Pepper virtual repository
20 git clone https://github.com/ros-naoqi/pepper_virtual.git
21
22 # Clone NAOqi driver repository
23 git clone https://github.com/ros-naoqi/naoqi_driver.git
24
25 # Clone Pepper robot repository
26 git clone https://github.com/ros-naoqi/pepper_robot.git
27
28 # Clone Pepper moveit config repository
29 git clone https://github.com/ros-naoqi/pepper_moveit_config.git
30
31 # Build the repository
32 cd .. && catkin_make
33
```

```
34 # Add the workspace to your ROS environment by sourcing the setup file in
      devel folder
35 source devel/setup.bash
36
37 # Add the setup to your .bashrc file so that you don't have to do this
     every time you open a new terminal
38 echo "source ~/workspace/ros/devel/setup.bash" >> $HOME/.bashrc
39
40 # Install additional packages
41 sudo apt-get install ros-melodic-joint-trajectory-controller
42
43 sudo apt-get install ros-melodic-ros-controllers
44
45 sudo apt-get install ros-melodic-pepper-meshes
46 # When configuring window opens up, you may agree to the license terms
     using the right/left arrow key and select <ok> and hit enter, and then
      select <Yes> to accept the terms and press enter
47
48 # Install rosdep
49 sudo pip install -U rosdep
50 sudo rosdep init
51 rosdep update
52
53 rosdep install --from-paths src --ignore-src -r -y
```

You might encounter the following error when launching the driver in the section 2.2.3.

> 🛑 **Error**
>
> terminate called after throwing an instance of 'qi::FutureUserException'what():
> Can't find service: ROS-Driver-Audio

At the moment the possible cause and solution couldn't be found. In order to workaround the error, deactivate the audio service. To deactivate the Audio Service, access the boot config file using the below command and set the audio key to "false." After making the change, remember to save the file.

```
sudo nano $HOME/workspace/ros/src/naoqi_driver/share/boot_config.json
```

The audio service is being deactivated as a temporary workaround to ensure the functionality of the NAOqi driver. It is important to note that this action will result in the microphone not working. Despite exploring various alternatives, the problem persisted, leaving with no other viable solutions currently covered in this manual. The upcoming version of the software installation manual will address this issue and provide alternative solutions to resolve the microphone functionality along with the NAOqi driver's operation.

2.2.3 Bring up Pepper

To establish a communication between the computer device and the Pepper robot, the IP address of the computer device and its network interface name, and the IP address of the Pepper robot is needed. Below are the steps to find this information. First make sure the Pepper robot and computer device are on the same network.
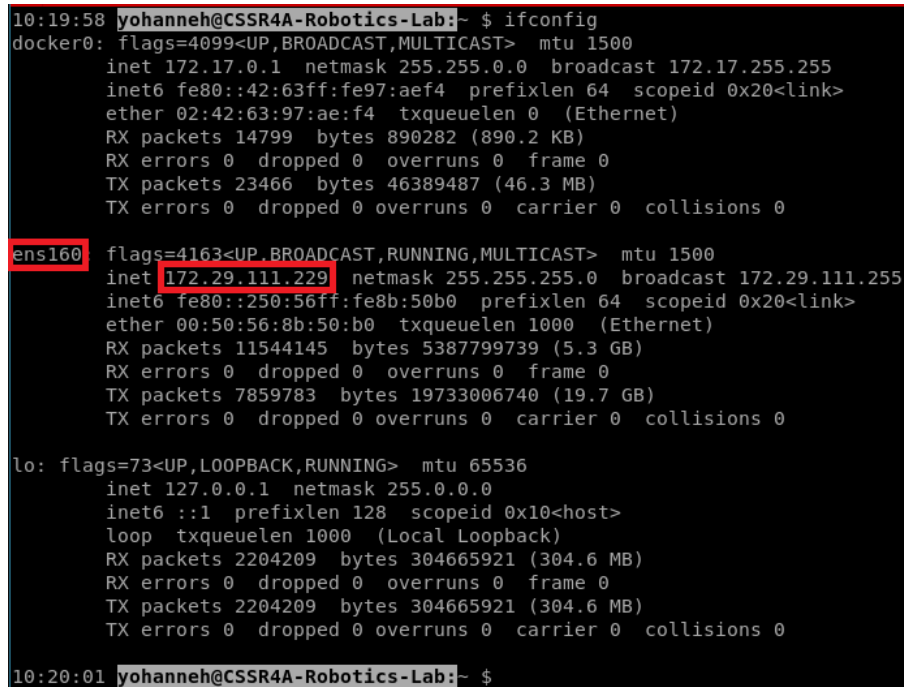
1. **IP address identification of the computer**
   In order to find the IP address of the computer device, open a new terminal and execute the following commands.

   (a) Install network tool (if not installed).

   ```
   sudo apt install net-tools
   ```

   (b) Identify network ip and interface name.

   ```
   ifconfig
   ```


Figure 1: "ifconfig" command output: Network interface name and IP address.

Upon executing the "ifconfig" command, the output will display information similar to the illustration shown in Figure 1. Take note of the network interface name along with its corresponding IP address as shown in the output. This IP address will be passed as a value for roscore_ip argument later in the document.

2. **IP address identification of the Pepper robot**
   To find the IP address the Pepper Robot, ensure the robot is powered on and connected to the network. Usually while the Pepper Robot is booting it says, "Hello, I'm Pepper, my internet address is $< robot\_ip >$ (for example, 172.29.111.230)" or in case you missed it, press the chest button once and make note of that IP address. This IP address will be passed as a value for robot_ip argument.

Using the above IP address of the robot, the IP address and the network interface name of the computer device, the following instruction will bring Pepper up over ROS and the NAOqi driver will be launched. Below are the instructions to follow:

1. Bring up the Pepper robot, assuming the robot is turned on.

```
roslaunch pepper_dcm_bringup pepper_bringup.launch robot_ip
    :=172.29.111.230 roscore_ip:=172.29.111.229 network_interface:=
    ens160
```

2. Launch the NAOqi driver.

```
roslaunch naoqi_driver naoqi_driver.launch nao_ip:=172.29.111.230
roscore_ip:=172.29.111.249 network_interface:=ens160
```

2.3   Setting up the Development Environment for the Gazebo Simulator

Section 2.2 assumes access to a physical Pepper robot. However, circumstances may arise where testing software without the physical robot becomes necessary. In such cases, a simulator can serve as an alternative. This section explains how to control the Pepper robot within a Gazebo simulator. The simulation is based on an open-source Pepper ROS environment developed by Sam Pfeiffer and Finn Rietz.[1]

2.3.1   Install Docker Using the Apt Repository

Below are a set of instructions to install the Docker engine on Ubuntu.[2]

1. Make sure to uninstall any conflicting packages.

```
for pkg in docker.io docker-doc docker-compose podman-docker
    containerd runc; do sudo apt-get remove $pkg; done
```

2. Update the apt package index and install packages to allow apt to use a repository over HTTP.

```
1 sudo apt-get update
2 sudo apt-get install ca-certificates curl gnupg
```

3. Add Docker's official GPG key.

```
1 sudo install -m 0755 -d /etc/apt/keyrings
2 curl -fsSL https://download.docker.com/linux/ubuntu/gpg |
3 sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
4 sudo chmod a+r /etc/apt/keyrings/docker.gpg
```

4. Use the following command to set up the repository.

```
1 echo \"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/
    keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \"$
    (. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \sudo
    tee /etc/apt/sources.list.d/docker.list > /dev/null
```

---

[1]The Pepper ROS simulator is available on the following link: https://github.com/frietz58

[2]This is a subset of the official documentation available at https://docs.docker.com/engine/install/ubuntu/.

5. Update the apt package index.

```
sudo apt-get update
```

6. Install Docker Engine, container, and Docker Compose.

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-
    buildx-plugin docker-compose-plugin
```

7. Verify that the Docker Engine installation is successful by running the hello-world image.

```
sudo docker run hello-world
```

This command downloads a test image and runs it in a container. When the container runs, it prints a confirmation message and exits.

2.3.2 Install the Pepper Gazebo Simulator

1. Clone the GitHub repository.

```
git clone https://github.com/frietz58/pepper_virtual
```

2. Move to the pepper_virtual directory.

```
cd pepper_virtual
```

3. Build image from provided Dockerfile.

```
sudo docker build -t awesome-pepper-sim
```

4. Allow Docker to open GUIs on the surrounding OS.

```
xhost +local:root
```

5. Run the Docker container.

```
sudo docker run -it -v /tmp/.X11-unix:/tmp/.X11-unix -e DISPLAY=
    unix$DISPLAY awesome-pepper-sim
```

6. Start the Gazebo simulation.

```
roslaunch pepper_gazebo_plugin pepper_gazebo_plugin_in_office_CPU.
    launch
```

If everything goes well, your simulation environment should be similar to the figure 2.

2.3.3 Driving the Pepper Robot in the Simulator

In this section, the Pepper robot is driven in the simulation environment using the **rqt** GUI joint trajectory controller application using the following instructions:

1. Open a new terminal, run the command below and take note of the returned container ID.
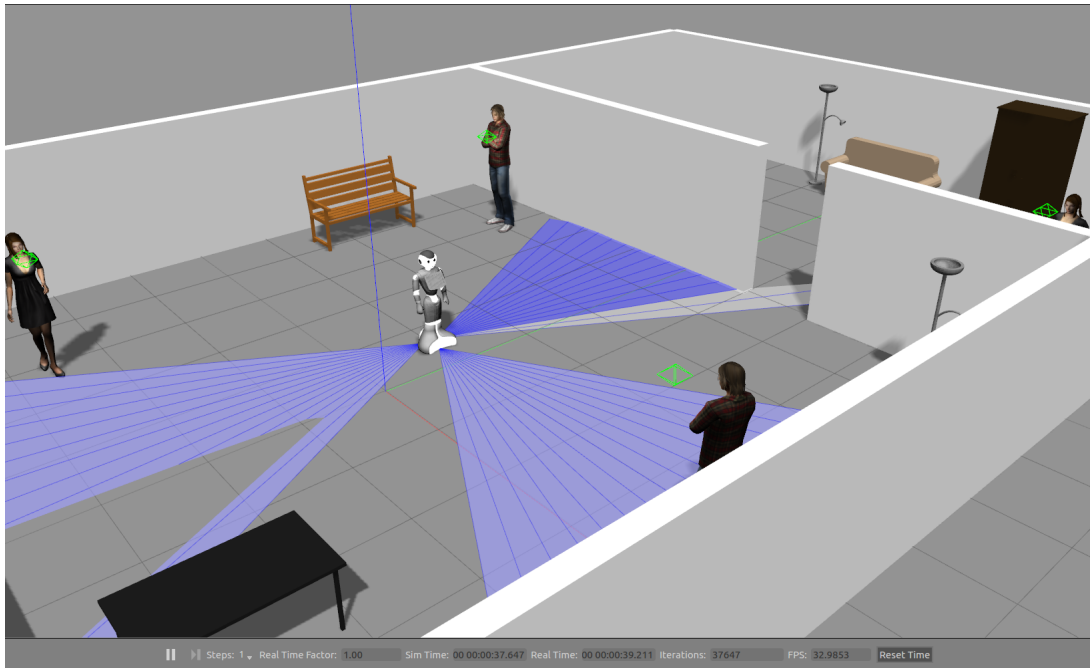
```
sudo docker ps
```

Figure 2: Pepper Gazebo simulation environment.

2. Attach a new shell to the container with the ID obtained from running the previous command.

```
sudo docker exec -it <CONTAINER-ID> bash
```

3. Start rqt steering with Pepper's base topic.

```
rosrun rqt_robot_steering rqt_robot_steering --default_topic:=/pepper
    /cmd_vel
```

4. Start joint trajectory controller to drive Pepper robot in the simulator.

```
rosrun rqt_joint_trajectory_controller
    rqt_joint_trajectory_controller
```

Upon initiating the joint trajectory controller, a user interface resembling Figure 3 will be displayed. To steer the robot, the rqt_robot_steering GUI can be utilized, where two sliders are available for adjusting velocity and steering. Additionally, the rqt_joint_trajectory_controller provides a GUI for the joint controller, allowing movement control through sliders corresponding to the available joints. For instance, the pelvis joints, HipPitch and HipRoll, can be controlled using this interface.

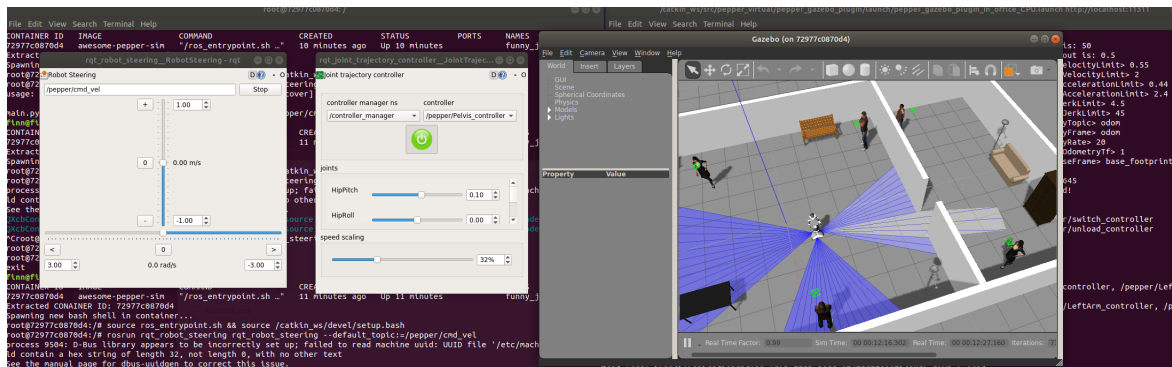Figure 3: Joint trajectory controller.

3 Installing and Running the CSSR4Africa Software

3.1 Installing the CSSR4Africa Software

This section will guide you through the installation process of the CSSR4Africa software in the source directory of the workspace. The outlined steps for installing the CSSR4Africa software are as follows:

```
1 # move to the source directory of the workspace
2 cd $HOME/workspace/ros/src
3
4 # clone the CSSR4Africa software from the GitHub repository
5 git clone https://github.com/cssr4africa/cssr4africa.git
6
7 # build the source files
8 cd .. && catkin_make
```

Upon completing the installation above, the **src** folder within the development environment's workspace will contain the CSS4Africa software meta-package and all the packages used to enable communication between ROS and NAOqi. Below is the directory structure of the workspace for reference.

```
workspace
└── ros
    ├── build
    ├── devel
    ├── src
        ├── cssr4africa
        │   └── pepper_diagnostic_routines
        ├── naoqi_dcm_driver
        ├── naoqi_driver
        ├── pepper_dcm_robot
        ├── pepper_moveit_config
        ├── pepper_robot
        ├── pepper_sensors_py
        └── pepper_virtual
```

3.2 Running the CSSR4Africa Software

In this section, the execution of the unit, integration, and system tests of the CSSR4Africa software packages for the physical robot and the robot simulator will be presented.

3.2.1 The Pepper Diagnostic Routine Test

The pepper_diagnostic_routines package contains the source files that have been written to perform the unit tests to evaluate the functionality and performance of sensors and actuators on the physical robot and the robot simulator.

3.2.1.1 The Pepper Diagnostic Routine Test on the Physical Robot

The pepper diagnostic routine test will be conducted on the physical robot as outlined below.

```
1  # move to the workspace directory
2  cd $HOME/workspace/ros/
3
4  # launch the diagnostic routines
5  roslaunch pepper_diagnostic_routines diagnostic.launch robot_ip:=<
       robot_ip> roscore_ip:<roscore_ip> network_interface:=<
       network_interface>
6
7  # launch the naoqi driver
8  roslaunch naoqi_driver naoqi_driver.launch nao_ip:=<robot_ip> roscore_ip:
        <roscore_ip> network_interface:=<network_interface>
9
10 # Test the back sonar
11 rosrun pepper_diagnostic_routines back_sonar
12
13 # Test the bottom camera
14 rosrun pepper_diagnostic_routines bottom_camera
15
16 # Test the depth camera
17 rosrun pepper_diagnostic_routines depth_camera
18
19 # Test the front camera
20 rosrun pepper_diagnostic_routines front_camera
21
22 # Test the wheels
23 rosrun pepper_diagnostic_routines goToPosition
24
25 # Test the hand touch
26 rosrun pepper_diagnostic_routines handTouch
27
28 # Test the head control (HeadPitch and HeadYaw)
29 rosrun pepper_diagnostic_routines head_control
30
31 # Test the laser
32 rosrun pepper_diagnostic_routines laser
33
34 # Test the left arm (LElbowRoll, LElbowYaw, LElbowYaw, LShoulderRoll and
       LWristYaw)
35 rosrun pepper_diagnostic_routines left_arm_control
36
37 # Test the left hand
38 rosrun pepper_diagnostic_routines left_hand_control
39
40 # Test the pelvis (HipPitch, HipRoll and KneePitch)
41 rosrun pepper_diagnostic_routines pelvis_control
42
43 # Test the right arm control (RElbowRoll, RElbowYaw, RShoulderPitch,
       RShoulderRoll, and RWristYaw)
44 rosrun pepper_diagnostic_routines right_arm_control
45
```

```
46  # Test the right hand
47  rosrun pepper_diagnostic_routines right_hand_control
```

3.2.1.2 The Pepper Diagnostic Routine Test on the Robot Simulator

In the upcoming version of the document, the execution of the Pepper diagnostic routine test on the Gazebo simulator will be included.

Principal Contributors

The main authors of this deliverable are as follows (in alphabetical order).

Adedayo Akinade, CMU-Africa
Deogratias Amani, CMU-Africa
Yohannes Haile, CMU-Africa
Mihiretab Taye Hordofa, CMU-Africa
Kleber Kabanda, CMU-Africa
Natasha Mutangana, CMU-Africa
David Vernon, CMU-Africa
Pamely Zantou, CMU-Africa

Document History

Version 1.0
> First draft.
> CSSR4A Team.
> 26 July 2023.