Setup Franka Simulation

November 24, 2023

1 Introduction

The simulation scenario for exercises 5 and 6 will no longer involve AUVs, but robotic manipulators. The robot adopted for this assignment is the Franka by Emika, a versatile 7-DOF manipulator for researchers (see the datasheet attached and Figure 2). Together with this guide, you are provided with a Python simulator to visualize two manipulators and perform some cooperative manipulation scenarios, by implementing the task priority inverse kinematic algorithm in Matlab.

2 System requirements and installation

In the Teams channel *AA2324 - Cooperative Robotics* you have a folder called *Franka Visualization tools* in which you find the Python simulator, this guide and the guide for the setup of the virtual machine (if needed). In order to be able to complete the exercise you need to have the following:

- Matlab installed, version > 2019a with the following toolbox installed:
  - Robotic System Toolbox
  - DSP System Toolbox
- Ubuntu version > 18.04 (probably the simulator works with other distributions, but additional installations of Python packages may be required.):
  - `python3` working
  - `pyBullet` installed
- To check the correct functioning of your Python environment on Ubuntu simply type in a shell: `python3 --version`. If a Python version is retrieved the check is done, if not follow the guide: https://phoenixnap.com/kb/how-to-install-python-3-ubuntu.
- To install `pyBullet` simply digit in a shell: `pip install pybullet`.
- To run the visualization of the manipulators go to the folder *python_simulator* and type in a shell `python3 franka_panda_simulation` to launch the simulator. You'll see a scene with two Franka Panda, the left Arm and the Right Arm see Figure 1. Rotate the camera by pressing `ctrl+left_button` while dragging the mouse, move the camera by `ctrl+Alt+left_button` while dragging the mouse.
- **Consider using a virtual machine for the simulator if you don't have Ubuntu on your PC**. Matlab can run on Windows since the data are sent to the simulation using UDP communication. **NB**: Follow the steps described in the file "setup_virtual Box_for_UDP_communication.pdf" if you have problems with the virtual machine.
- Test the UDP connection between Matlab and Python simulator by running the following code on Matlab.

```
hudps = dsp.UDPSender('RemoteIPPort',1500);
hudps.RemoteIPAddress = '127.0.0.1';
q = [0,0,0,0,0,0,0]';

for t=1:0.1:30
    disp(t);
    step(hudps,[q;q]);
    pause(1);
end
```

The robots should assume the configuration depicted in Figure 2.

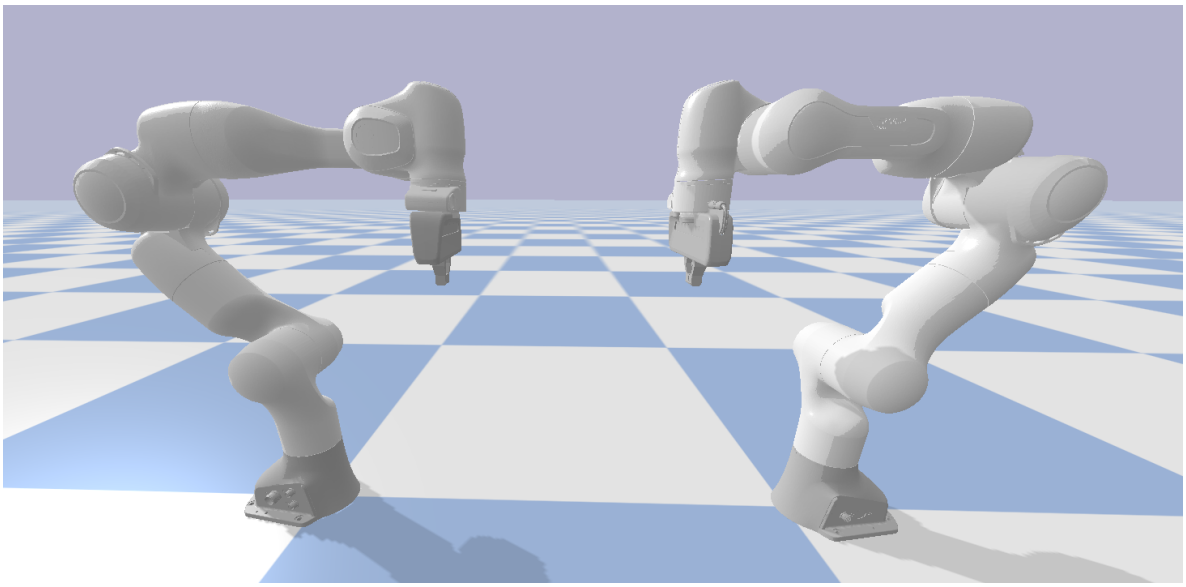

Figure 1: Left and Right robotic arms

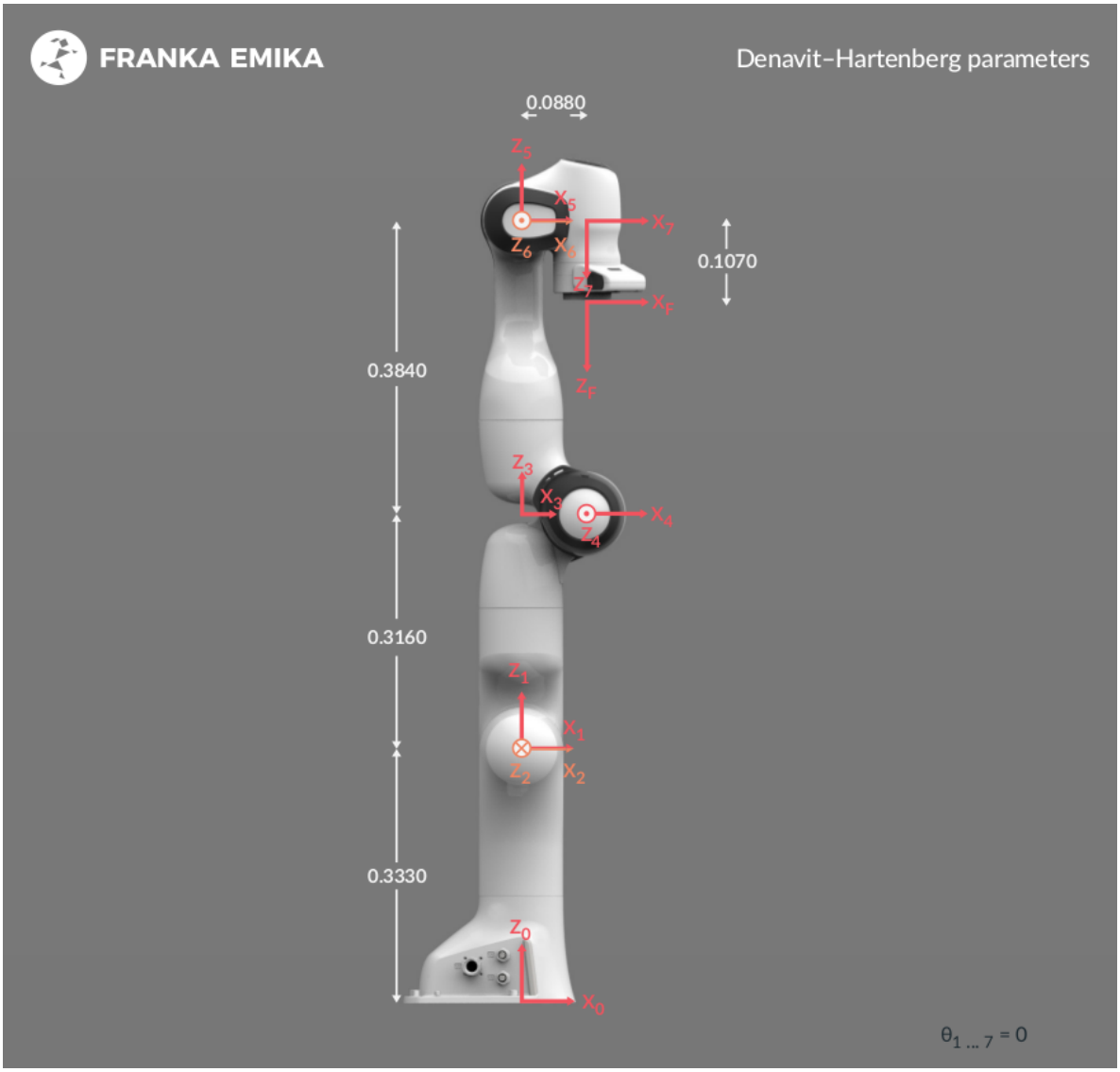

Figure 2: Robot Specifications