TECHNICAL UNIVERSITY OF CLUJ-NAPOCA
FACULTY OF AUTOMATION AND COMPUTER SCIENCE

ROS Project

Student name: *Bădău Elena-Nicoleta*

Course: *Robotics Control Systems* – Professor: *Dr.Eng. Tassos Natsakis*
Due date: *January 17th, 2023*

1.Introduction

Project goal and choice motivation

Project goal. The **goal** of this project was for us, the students, to get used to working with **Robotic Operating System (ROS)** and to perform a simple task with a chosen robot.

Choice motivation. I chose the **Crane-X7 robotic arm** because it offers great mobility (it possesses 7 revolute joints) and it is fairly easy to understand and control. The reason behind ROS is straightforward: it is the most used and capable robotic platform that exists. It is also easy to learn.

2. Description and Implementation

Crane-X7.Robotic Platform

Crane-X7. Because this robot possesses 7 revolute joints, it provides an orthogonal 7-axis design. This mobility is a great feature because it resembles the human arm pretty well, allowing more complex motions to be performed, such as object manipulation, following a specific trajectory, overcoming obstacles, and so on. The packages encapsulate high-level programming functions which help the user understand the background mechanics and also develop projects faster. This robot has some features such as work operation with obstacle avoidance, selectable mode, ROS adaptive, two finger hand end effector equipped, compact body, and selectable body color.

Description of the task

In order to be able to install and configure it on my computer I did the following steps:

(*1*) Install ROS using the Windows Subsystem for Linux (WSL) - available on Windows 10.

(*2*) Clone the GitHub repository from the Crane-X7 official GitHub page.

(*3*) Install package dependencies.

(*4*) Build packages using *catkin_make*.

(*5*) Study an example. I chose *pick_and_place_in_gazebo_example.py*

(*6*) Run the simulation on Gazebo using:
roslaunch *crane_x7_gazebocrane_x7_with_table.launch*

(*7*) Create the script. In the beginning, I picked and placed a cube with hardcoded coordinates, using the robotic arm. After this, I tried with two cubes, and so on.

(*8*) Run the script I created using: rosrun *crane_x7_examplesfinal_project_crane_x7.py* (*final_project_crane_x7.py* being the file where I stored my code)

Implementation of the task

Implementation. After setting up the simulation part in Gazebo, I choose an object manipulation task consisting of 3 pick and place tasks of 3 cubes, placing the cubes on top of each other, using the robotic arm. In order to do that I chose to start from Crane-x7 pick and place in gazebo example. I added the following features or changes:

(*1*) I adapt the code such that it grabs an object from an identified position and moves it to another, where I want to put the cubes on top of each other.

(*2*) Instead of using hardcoded coordinates the robot always tracks the real-time coordinates of the objects using ModelStates imported from *gazebo_msgs.msg* library where simulation objects coordinates are stored.

(*3*) After each move of a cube, the robotic arm returns to its initial position. The names of the cubes are put in a list.

The "simulated world" where the action takes place contains the robot set up on a table and 3 cubical objects.

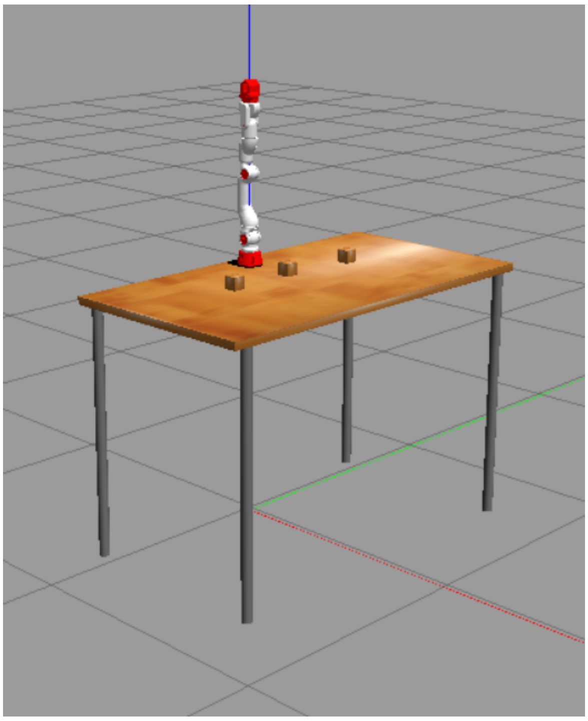

Figure 1: Initial position

Visualisation. Using the x, y, z coordinates from ModelStates imported from *gazebo_msgs.msg* library the robot is capable of tracking the desired object location. The process can be described in 7 steps:

First step: The robot will moves in "home" posture define in SRDF for 3 seconds.

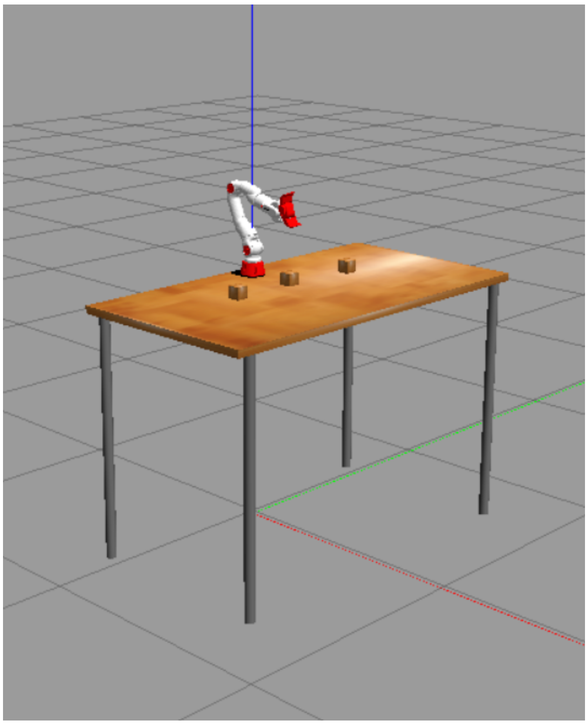

Figure 2: "Home" posture

Second step: It approaches the first cube *wood_cube_5cm*, then it grabs the object closes the gripper and moves it to the installation position.

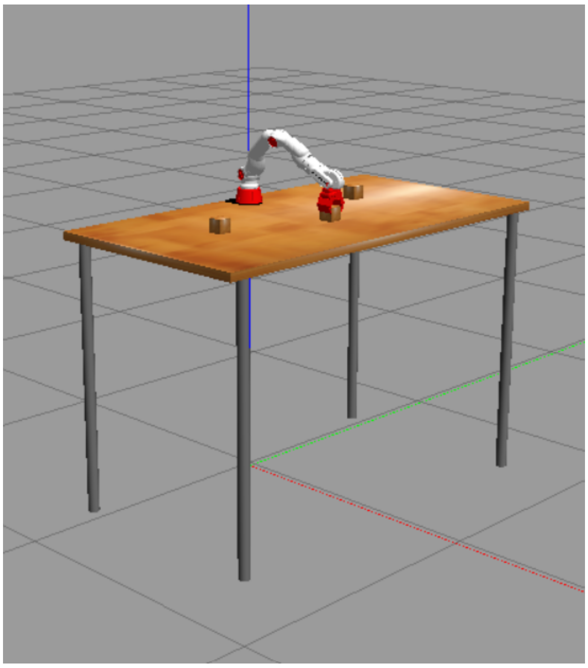

Figure 3: Robot moves the first object

Third step: The robot returns to "home" posture.

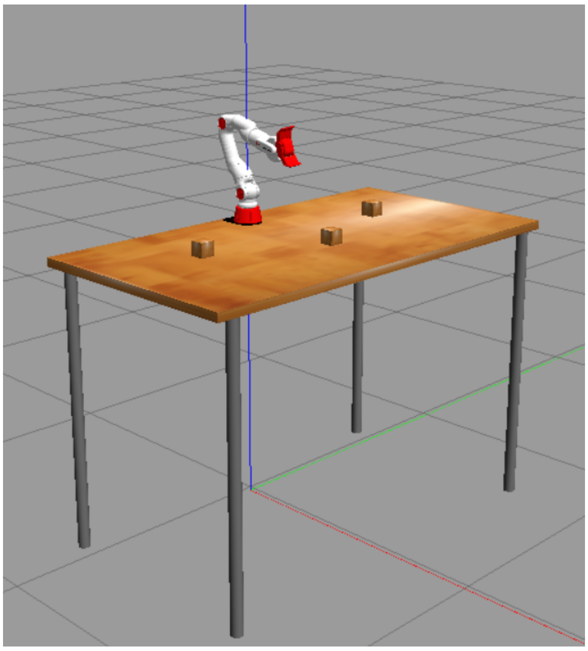

Figure 4: "Home" posture

Fourth step: It approaches the second cube *wood_cube_5cm_clone*, then it grabs the object, closes the gripper and moves it to the installation position. But, here we have different values for pick and leave variables.

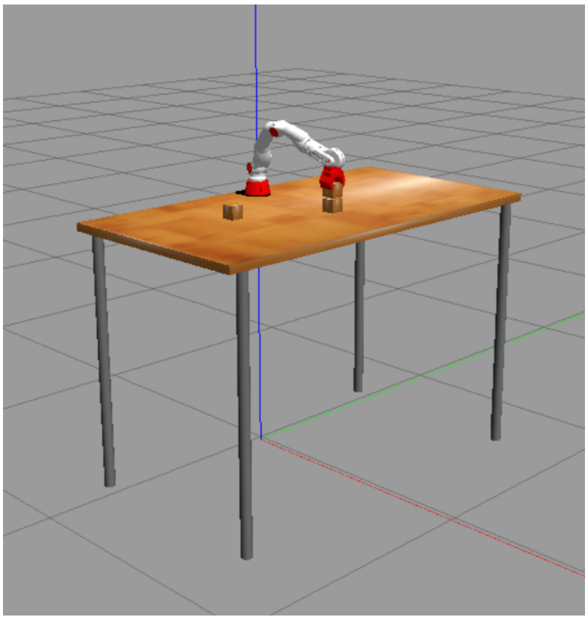

Figure 5: The robot moves the second object

Fifth step: The robot returns again to "home" posture.

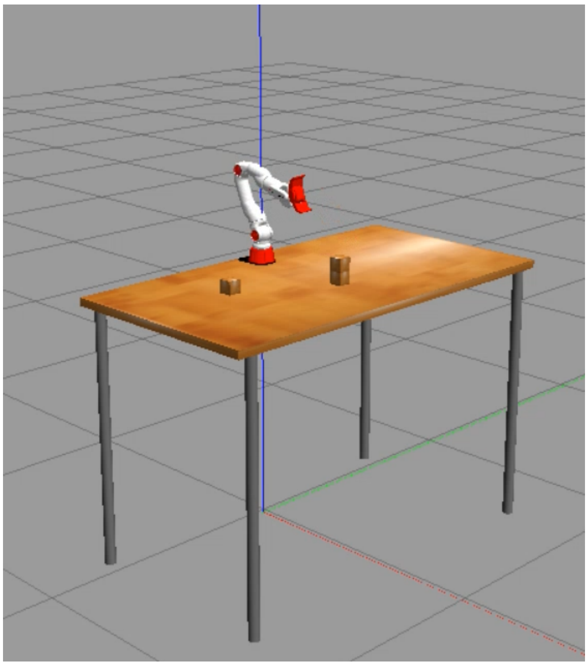

Figure 6: "Home" posture

Sixth step: It approaches the third cube *wood_cube_5cm_clone*, then it grabs the object, closes the gripper and moves it to the installation position. But, here we also have different values for pick and leave variables.

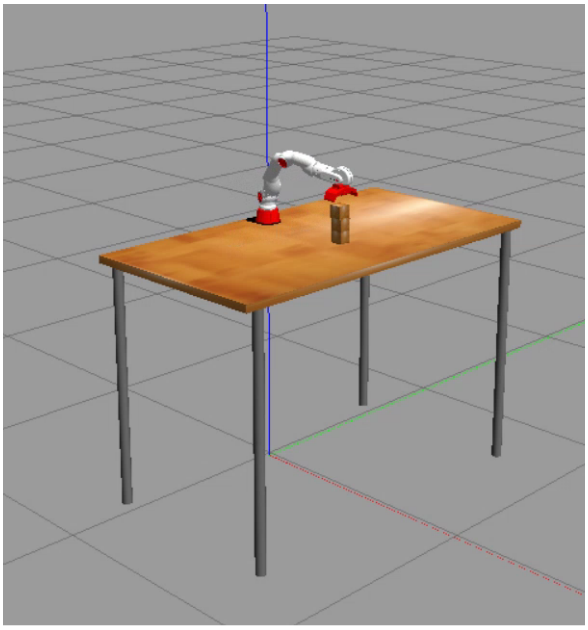

Figure 7: The robot moves the third object

Seventh step: The robot returns again to "home" posture.

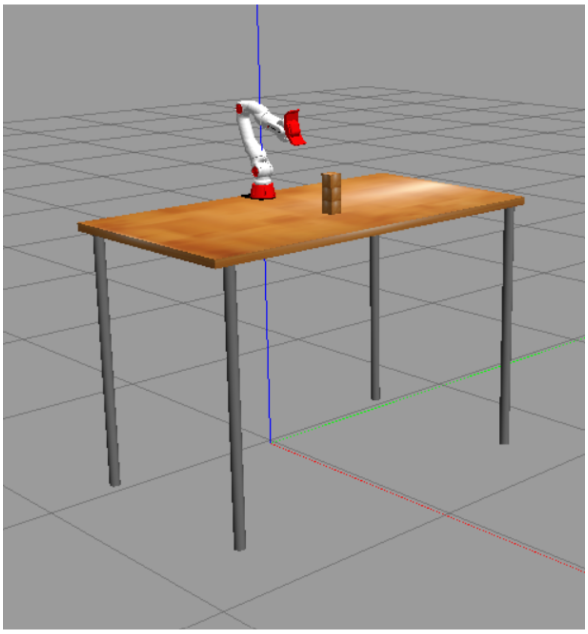

Figure 8: "Home" posture

3.Capabilities and Limitations

Performance description

Capabilities. Regarding the performances as it may be observed from the video, all movements can be executed correctly, relatively fast and "smooth". The arm velocity is constant.

Limitations. The obvious limitation of this robotic arm is its length: when it reaches maximum length. If an object is placed at a distance that is too large than the length of the robotic arm or as far as the gripper can grasp, the arm will not be able to grasp that object, displaying the message "Failed to grip an object". Another limitation would be the lifting of the cubes, these can be lifted and placed only up to a certain height, otherwise the message "Failed to place an object" will be displayed.

4.References

Resources used

(*1*) ROS Tutorials

(*2*) CPU Virtualization

(*3*) X Server

(*4*) CRANE_X7

(*5*) CRANE_X7 official github repository

(*6*) CRANE_X7 code examples

(*7*) CRANE_X7 official website

Here you will find a video simulation of the robot.

Script which performs the actions described above.