MCT344- Industrial Robotics

Submitted by:

Salma Amr Metwally 20p4526

Hossam Eldeen Amr Torky 20p8211

Submitted to:

Dr Shady Ahmed

Eng Dina Zakaria

ABSTRACT

This report presents the development and simulation of a 6 degrees of freedom (DOF) manipulator using the MoveIt framework, a powerful tool for motion planning in robotics. The primary objective is to demonstrate MoveIt's capabilities in generating collision-free paths and visualizing these movements in virtual environments, specifically through RViz and Gazebo. The report details the planning pipeline, implementation steps, and the advantages of using MoveIt for efficient and reliable manipulator planning. Additionally, a Python script is introduced to facilitate motion planning using MoveIt, showcasing its flexibility and simplicity in defining robot models, planning paths, and executing motions. The successful execution of complex motion plans and the visualization of the manipulator's behavior underscore the effectiveness of this approach for a wide range of robotics applications, from industrial automation to service robots.

TABLE OF CONTENTS

LIST OF FIGURES

1.INTRODUCTION

The field of robotics has seen significant advancements in motion planning, particularly with the development of sophisticated frameworks like MoveIt. This report explores the application of MoveIt for planning and simulating the movements of a 6 degrees of freedom (DOF) robotic manipulator. Motion planning is a critical component of robotic systems, enabling robots to navigate complex environments and perform precise manipulations. By leveraging MoveIt's comprehensive suite of tools for motion planning, collision detection, and trajectory execution, we aim to demonstrate the framework's effectiveness in generating and visualizing collision-free paths. The simulation of these movements is carried out in RViz and Gazebo, providing a virtual environment to observe and refine the manipulator's behavior.

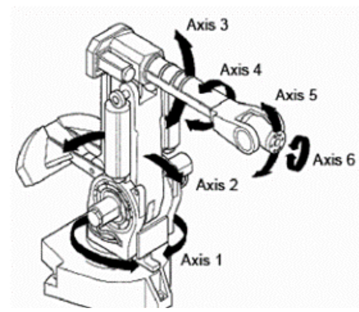


Figure 1 6 DOF Robot

In addition to the overall planning process, the report introduces a Python script designed to simplify motion planning tasks using MoveIt. This script exemplifies the framework's flexibility, allowing users to easily define robot models, plan paths, and execute motions across a variety of robotic applications, from industrial automation to service robots. The subsequent sections detail the implementation steps, highlight the advantages of using simulation tools, and showcase the successful execution of complex motion plans, ultimately demonstrating the robust capabilities of MoveIt in real-world robotics scenarios.

2.MoveIt Configuration

2.1. Launching MoveIt

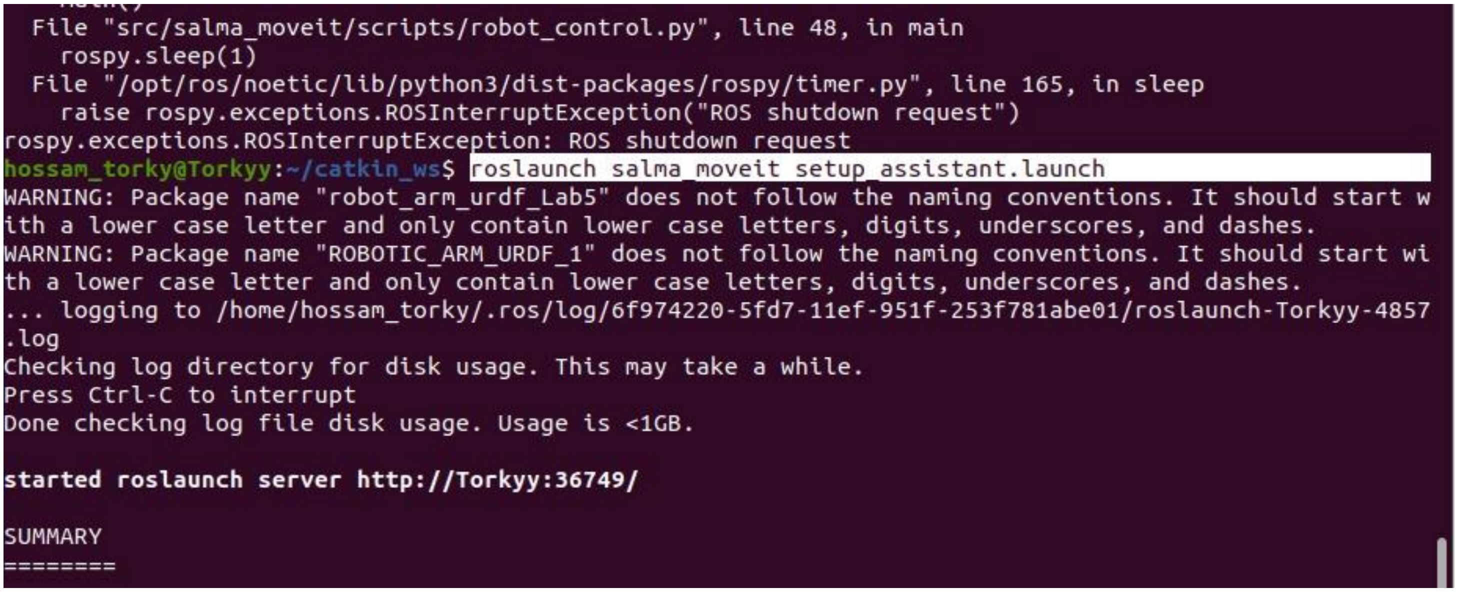
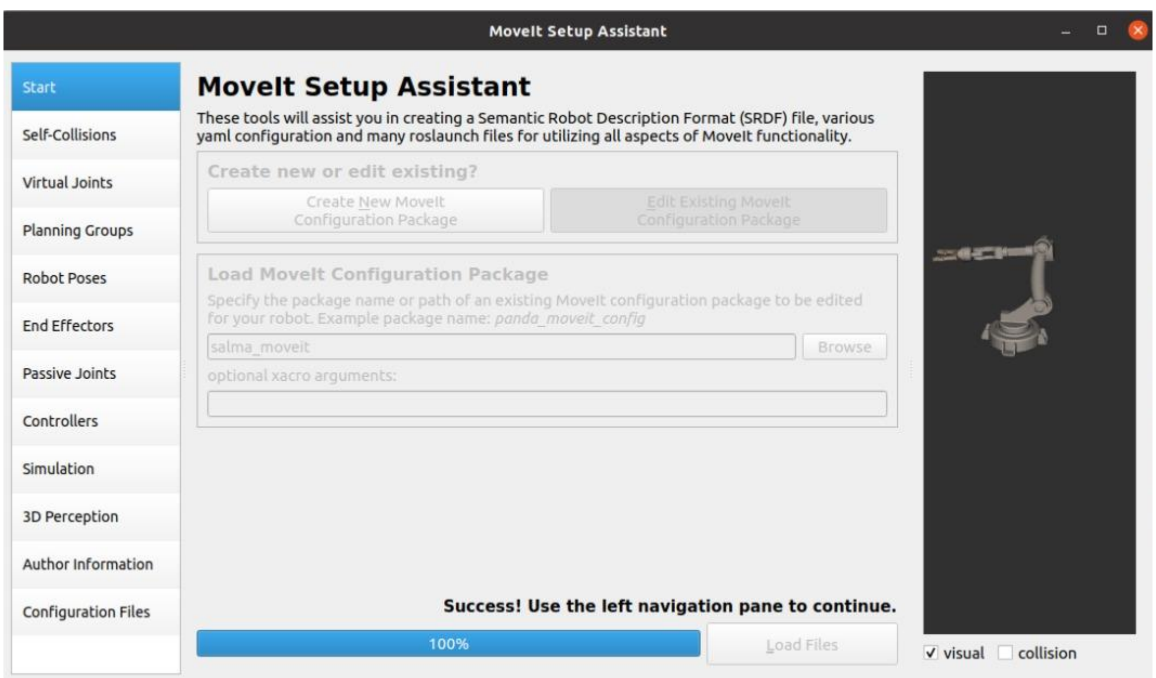


Figure 2MoveIt launch

2.2. Edit in MoveIt

- **We created a new moveit package inside of our project workspace**.



2.2.1. Self-collision

- **This step prevents the links from collision so it disables any possibility that they can collide with each other**

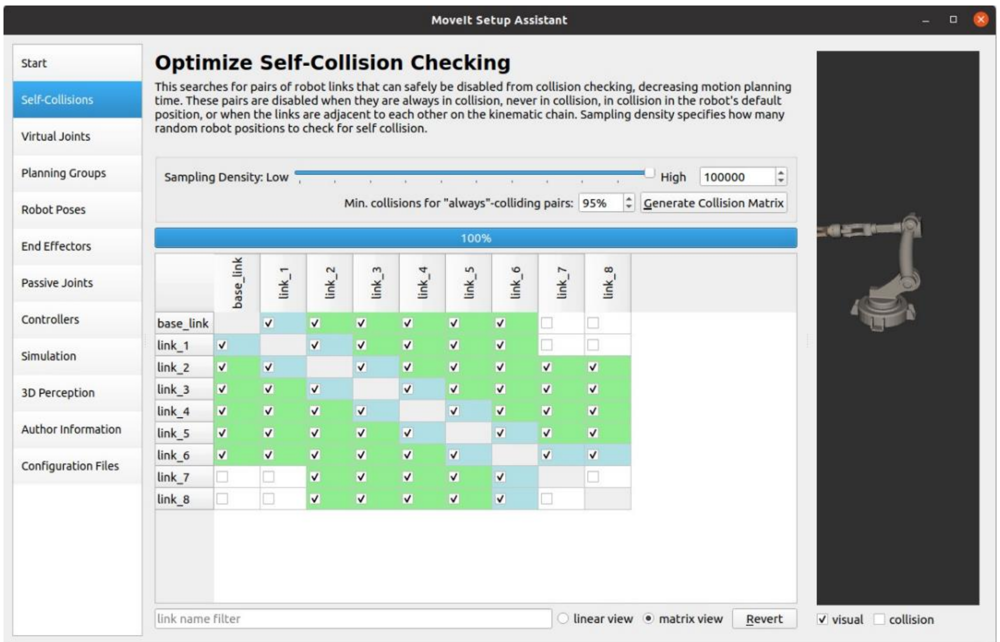


Figure 3 Self collision

2.2.2. Planning Groups

- **We divided our joints into two groups, the arm group which represents from our first to our fourth link and the hand group which represents our fifth & sixth group for the end effector**

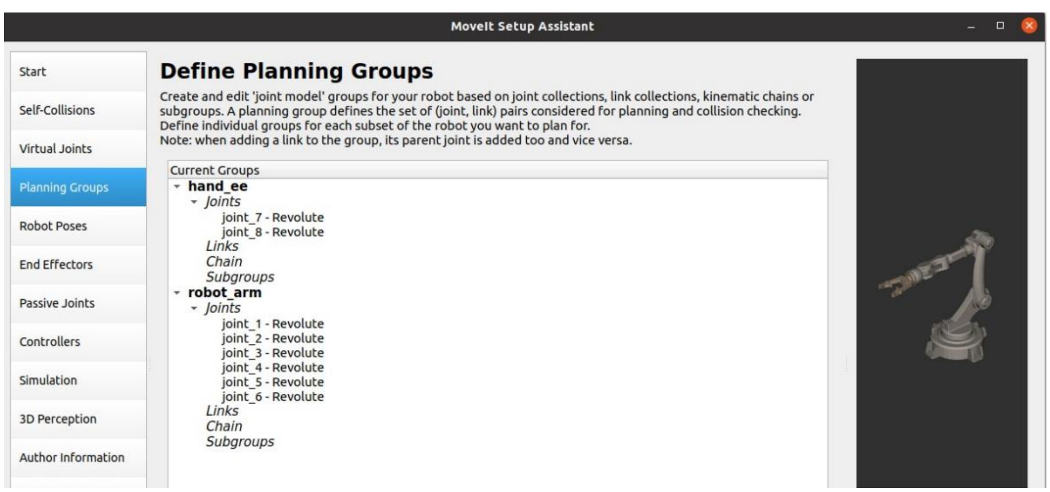


Figure 4 planning groups

2.2.3. Robot Poses

- **Hand close position**

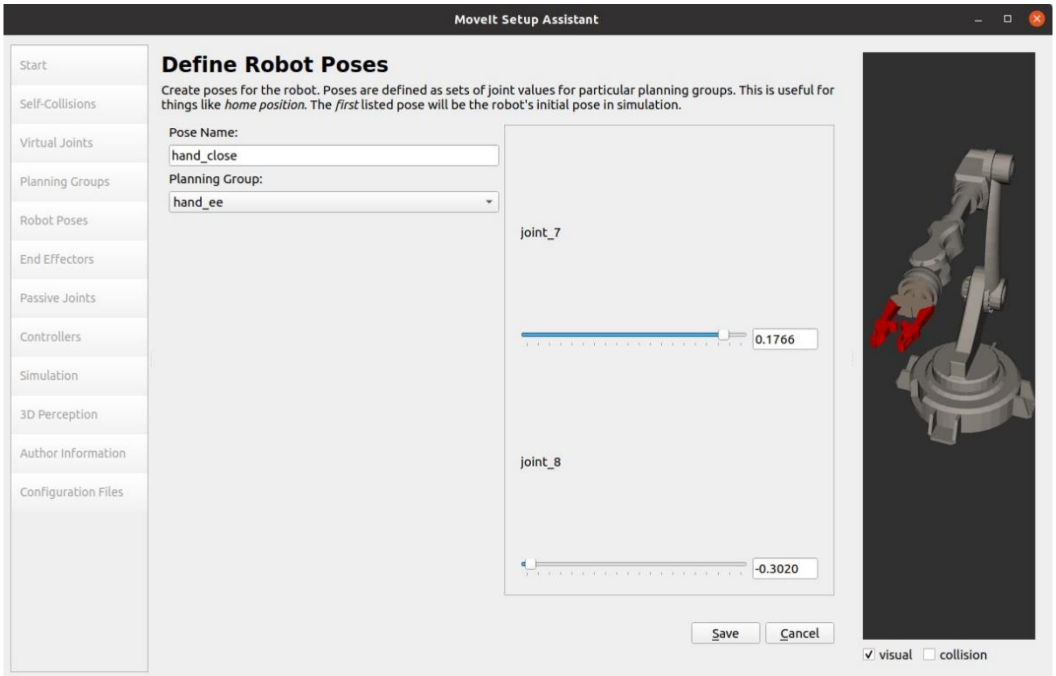


Figure 5 Hand close

- **Hand Open position**

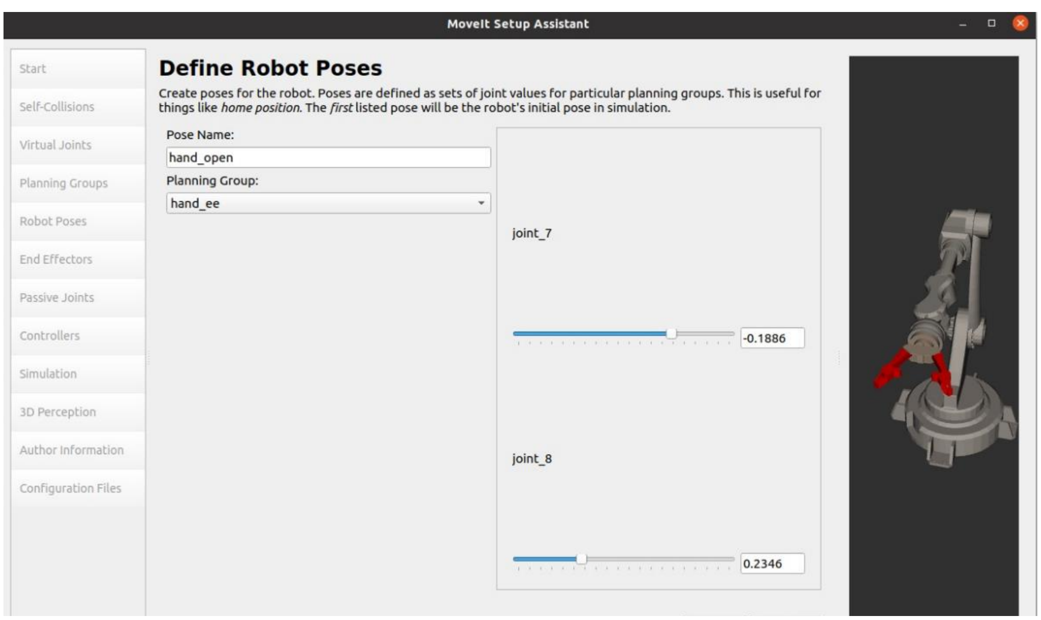


Figure 6 Hand open

- **Pose 1**

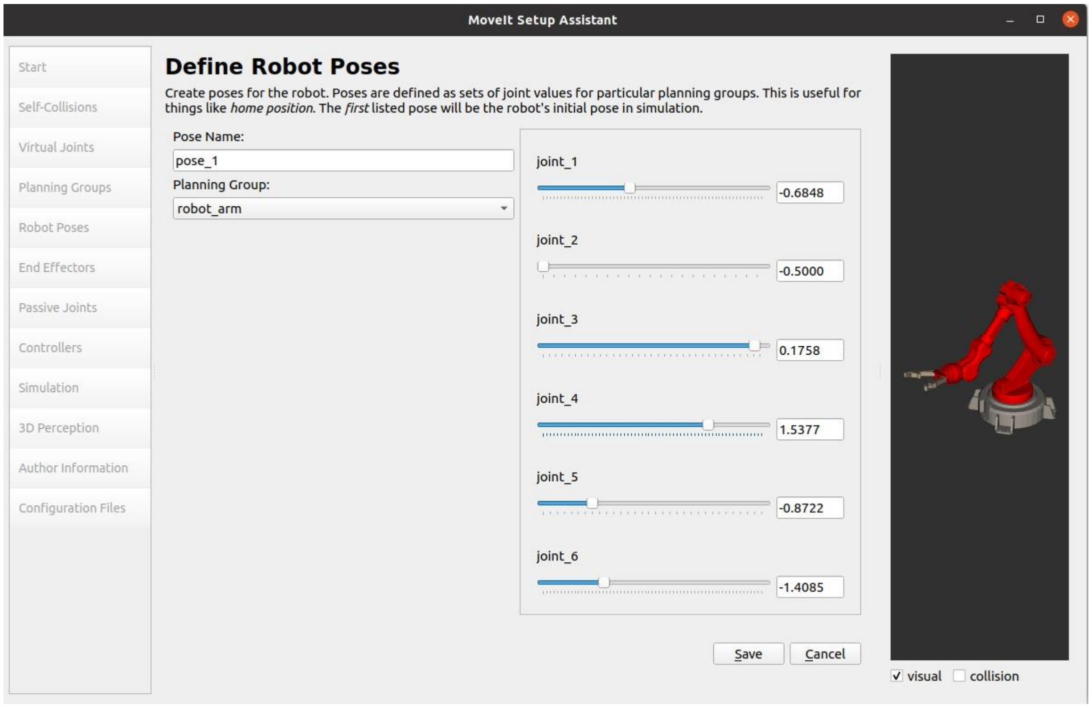


Figure 7 Pose 1

- **Pose 2**

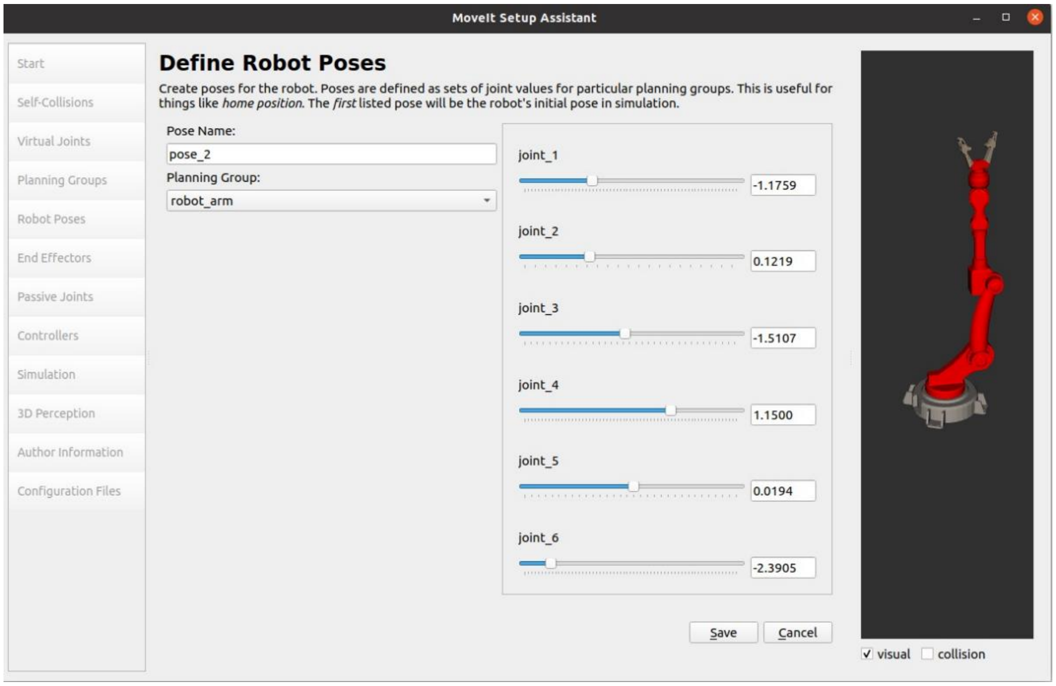


Figure 8 Pose 2

- **Pose 3**

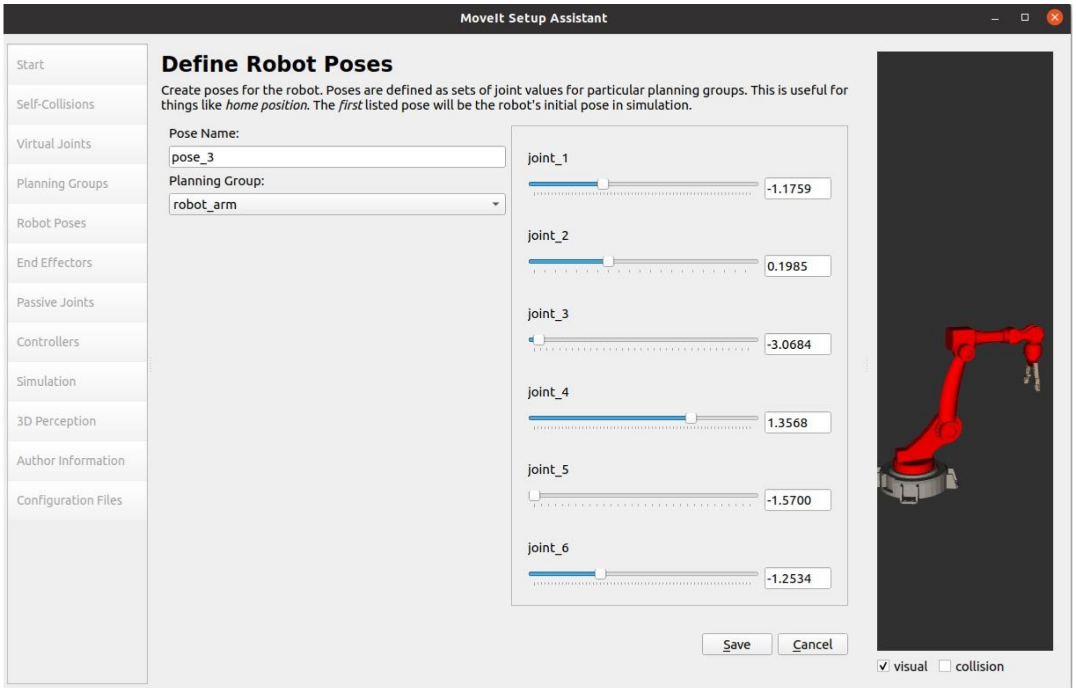


Figure 9 Pose 3

2.2.4. End Effectors

- **This represents our end effectors' definition where we assign it a name, a group, and a parent link**

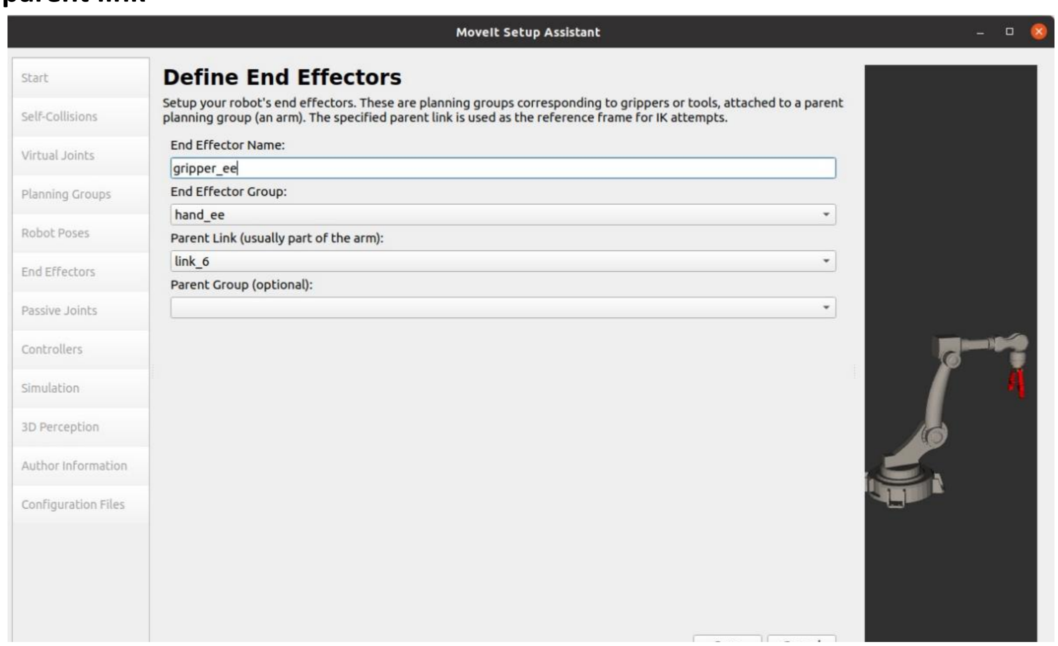


Figure 10 Define end effector

2.2.5. Controllers

- **We defined a controller for each group, and we named them as in the .yaml file as well as defining their type**

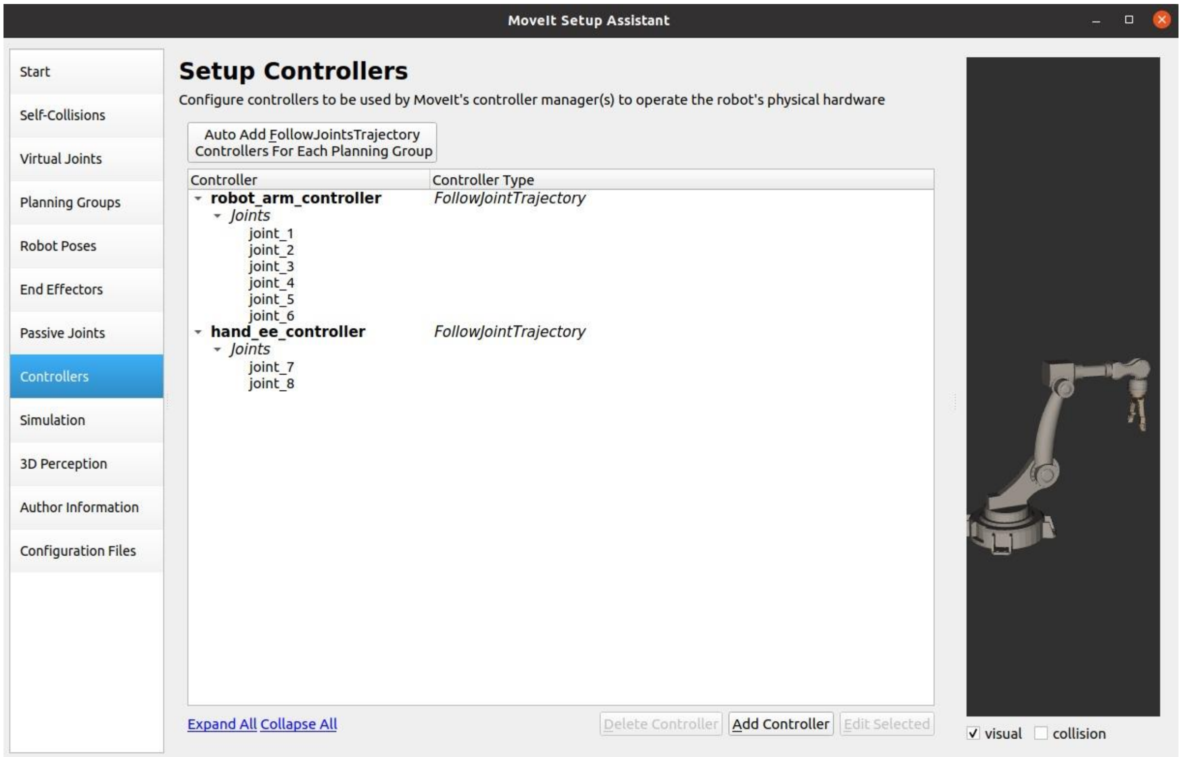


Figure 11 setup controllers

3.Launching Gazebo & RVIS

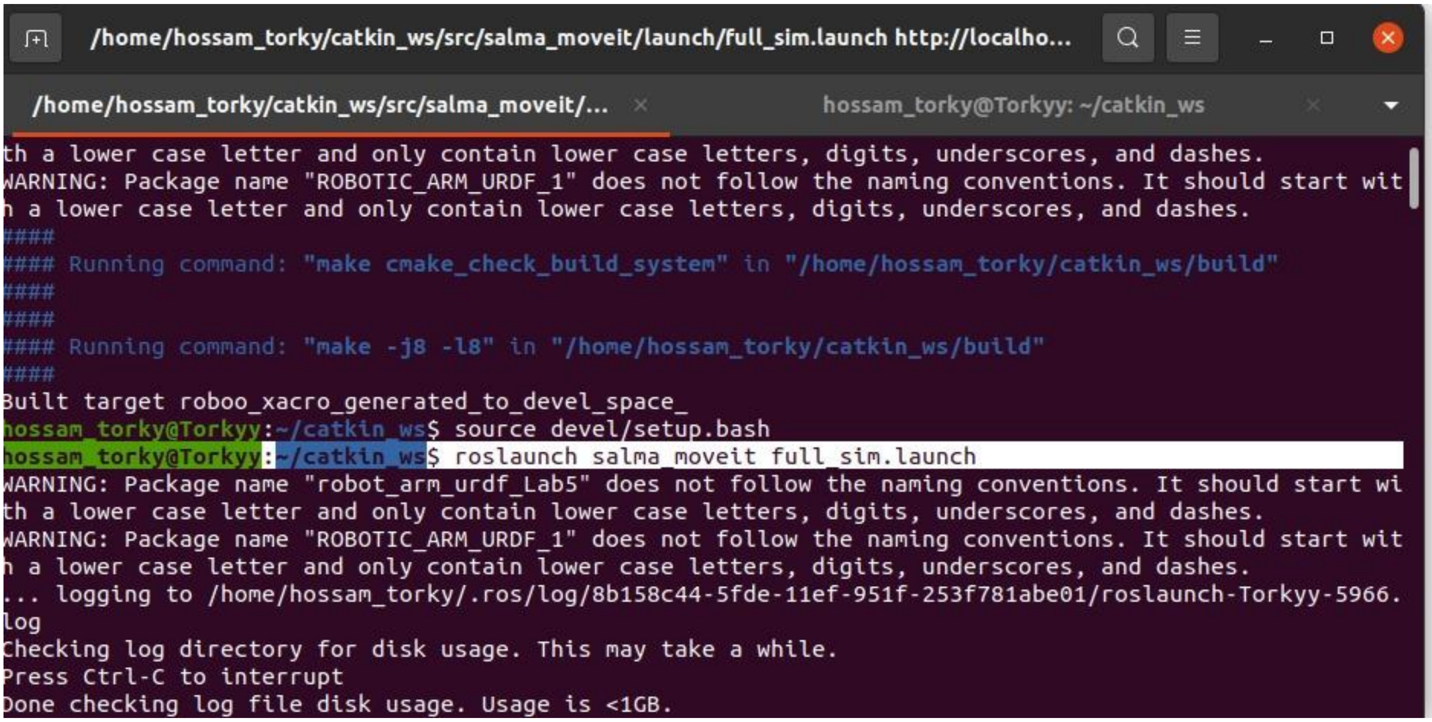
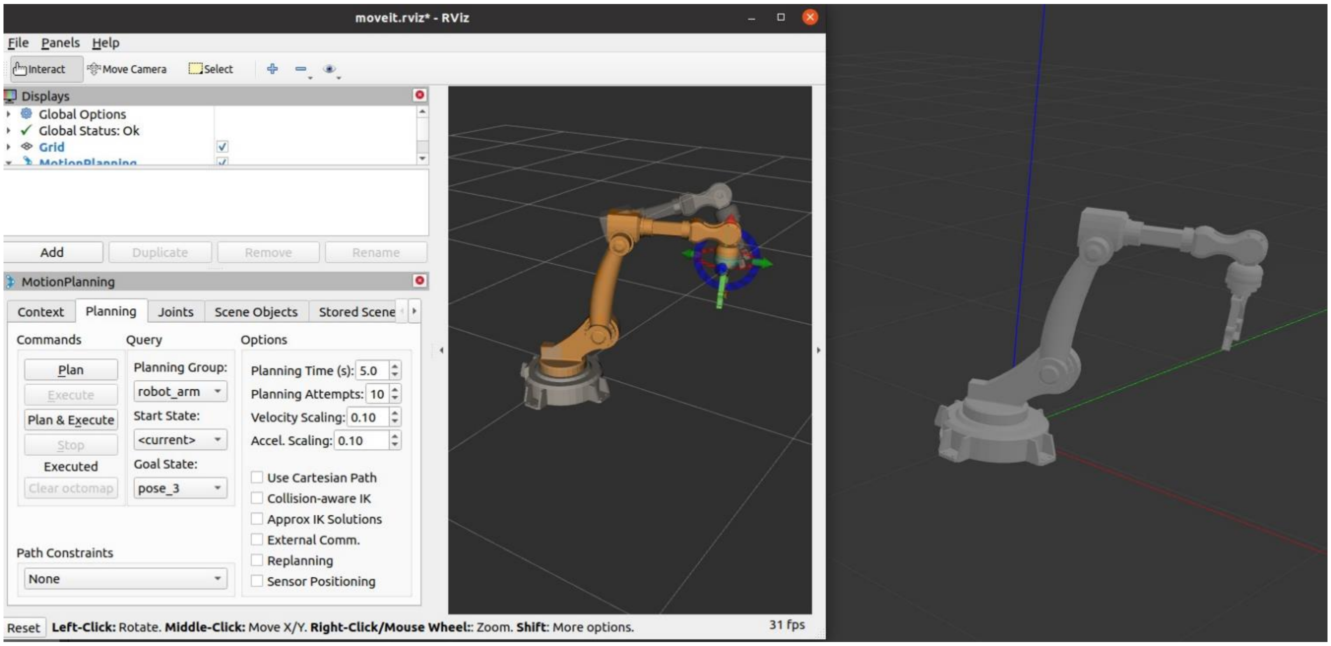
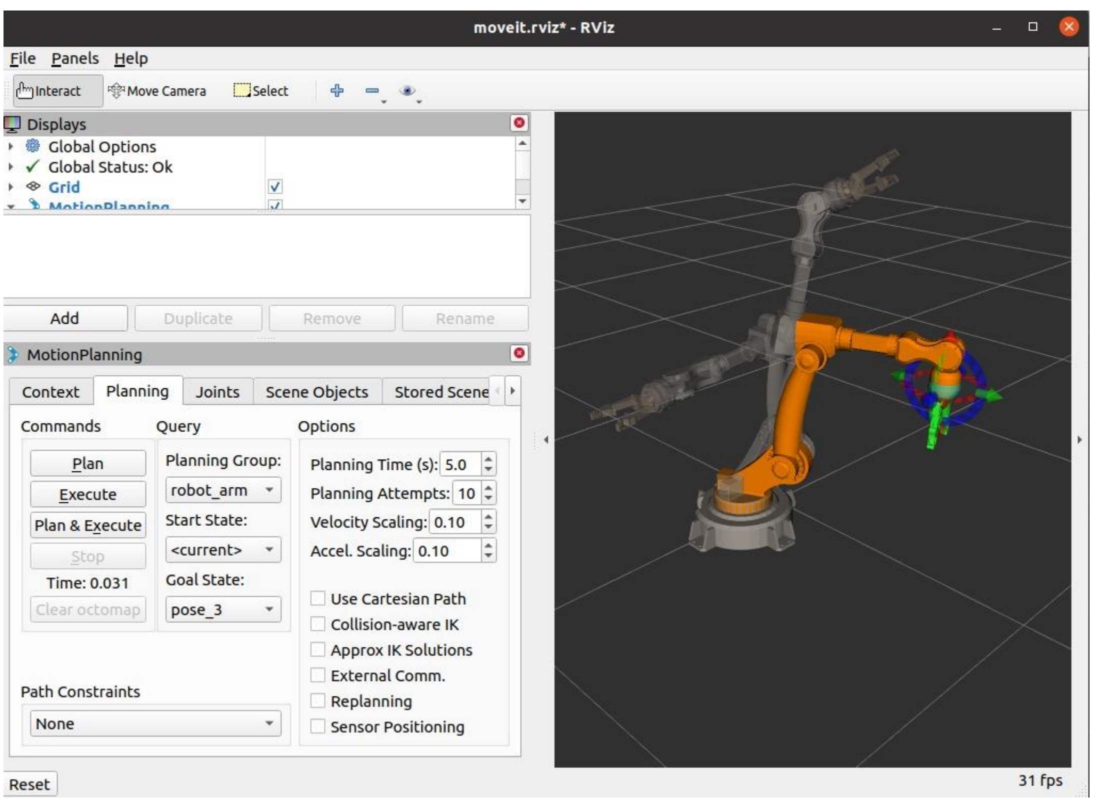
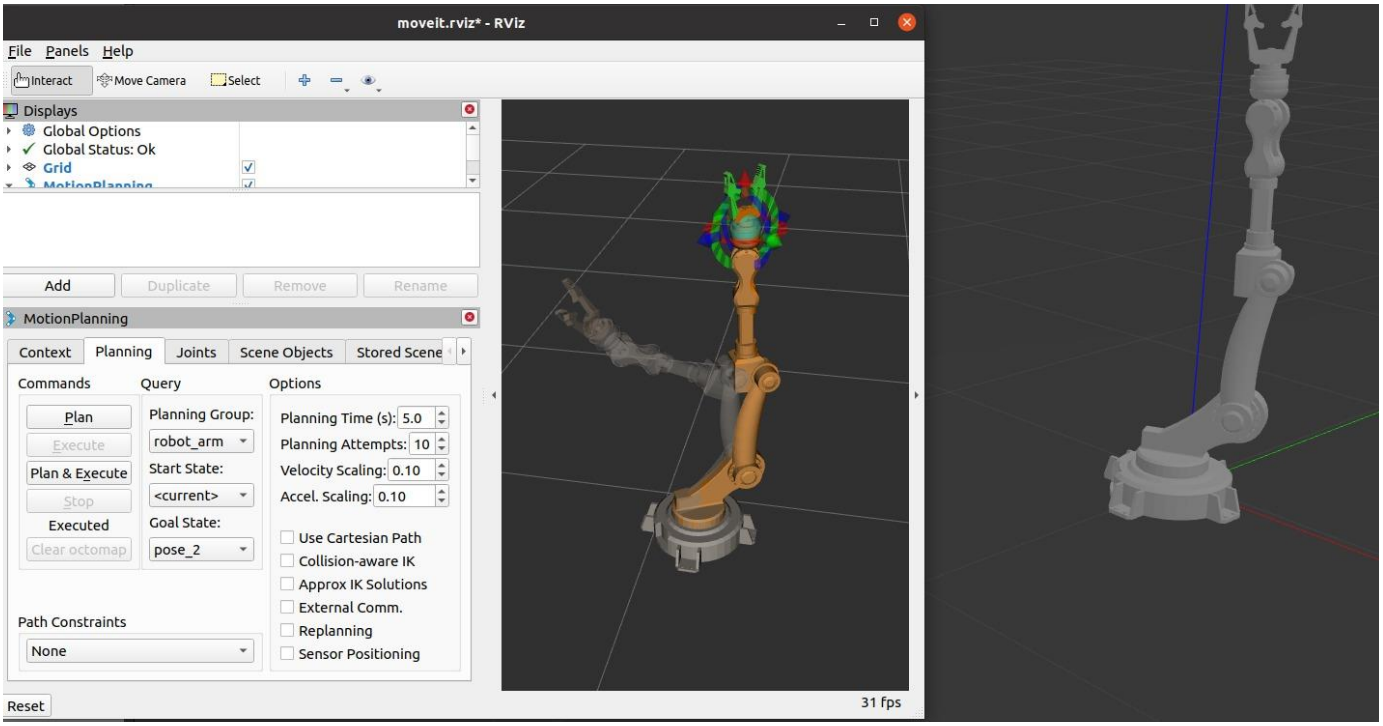


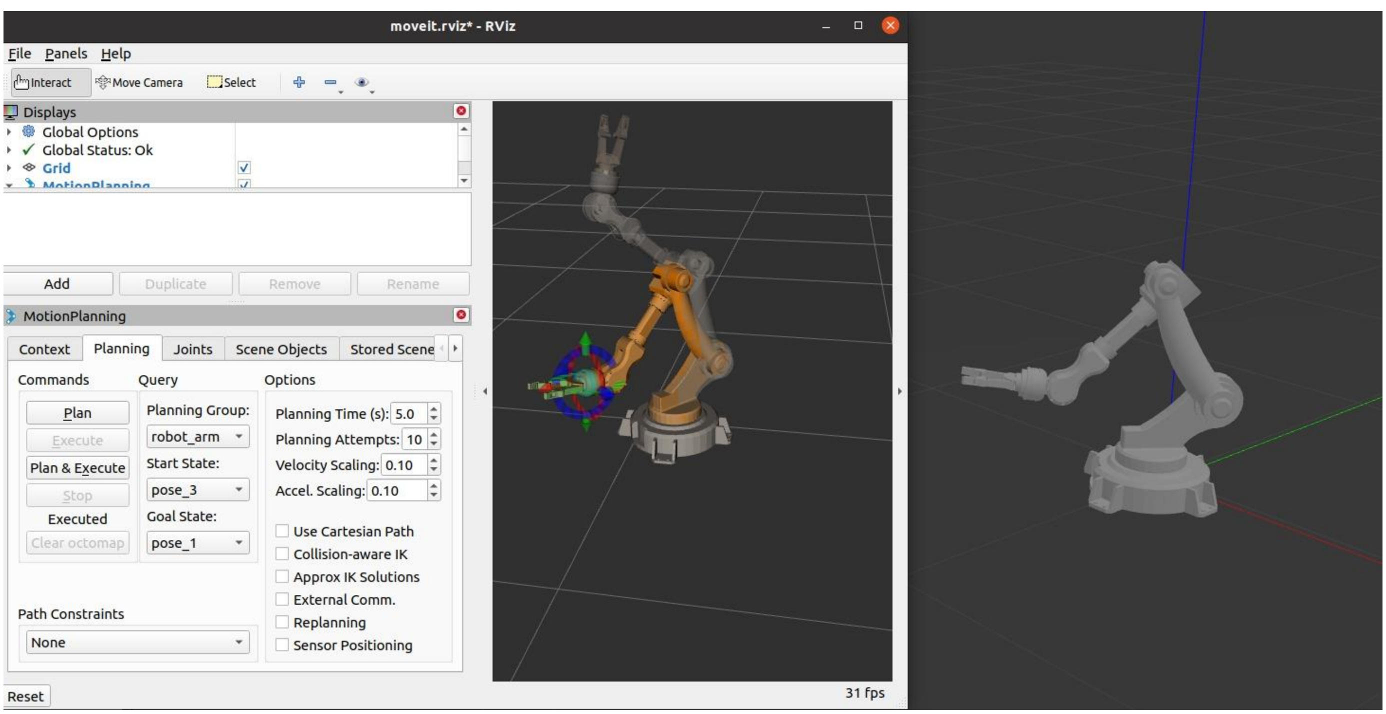
Figure 12 Gazebo and RVIS launch

3.1. Simulation






moveit.rviz* - RViz
File Panels Help
Interact
Move Camera
Select
Displays
Global Options
Global Status: Ok
Grid
MotionPlanning
Add
Duplicate
Remove
Rename
MotionPlanning
Context
Planning
Joints
Scene Objects
Stored Scene
Commands
Query
Options
Plan
Execute
Plan & Execute
Stop
Executed
Clear octomap
Planning Group:
robot_arm
Start State:
<current>
Goal State:
pose_2
Planning Time (s): 5.0
Planning Attempts: 10
Velocity Scaling: 0.10
Accel. Scaling: 0.10
Use Cartesian Path
Collision-aware IK
Approx IK Solutions
External Comm.
Replanning
Sensor Positioning
Path Constraints
None
Reset
31 fps


moveit.rviz* - RViz
File Panels Help
Interact
Move Camera
Select
Displays
Global Options
Global Status: Ok
Grid
MotionPlanning
Add
Duplicate
Remove
Rename
MotionPlanning
Context
Planning
Joints
Scene Objects
Stored Scene
Commands
Query
Options
Plan
Execute
Plan & Execute
Stop
Executed
Clear octomap
Planning Group:
robot_arm
Start State:
pose_3
Goal State:
pose_1
Planning Time (s): 5.0
Planning Attempts: 10
Velocity Scaling: 0.10
Accel. Scaling: 0.10
Use Cartesian Path
Collision-aware IK
Approx IK Solutions
External Comm.
Replanning
Sensor Positioning
Path Constraints
None
Reset
31 fps

4.Python Script

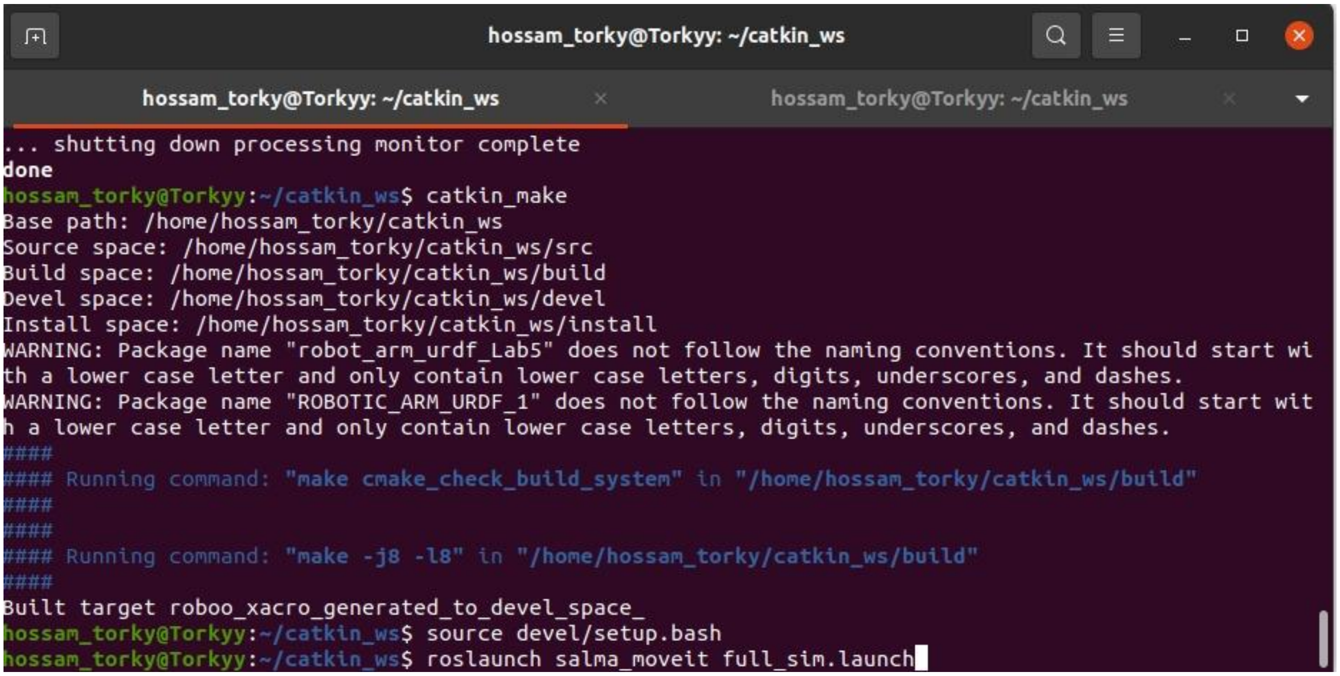
robot_control.py | salmaeltohfaa_urdf.srdf | salmaeltohfaa_urdf.srdf

```python
import sys
import rospy
import moveit_commander
import moveit_msgs.msg
from math import pi

moveit_commander.roscpp_initialize(sys.argv)
rospy.init_node('move_robot', anonymous=True)

robot = moveit_commander.RobotCommander()
scene = moveit_commander.PlanningSceneInterface()

def move_robot_arm(joint1, joint2, joint3, joint4, joint5, joint6):
    group_name = "robot_arm"  # Replace with your arm's group name
    group = moveit_commander.MoveGroupCommander(group_name)

    joint_goal = group.get_current_joint_values()
    joint_goal[0] = joint1
    joint_goal[1] = joint2
    joint_goal[2] = joint3
    joint_goal[3] = joint4
    joint_goal[4] = joint5
    joint_goal[5] = joint6

    group.go(joint_goal, wait=True)
    group.stop()

def move_end_effector(joint1, joint2):
    group_name = "hand_ee"  # Replace with your end effector's group name
    group = moveit_commander.MoveGroupCommander(group_name)

    joint_goal = group.get_current_joint_values()
    joint_goal[0] = joint1
    joint_goal[1] = joint2

    group.go(joint_goal, wait=True)
    group.stop()


def main():
        move_end_effector(-0.1243,0.2075)
        rospy.sleep(1)
        move_robot_arm(-0.6849,-0.5,0.1758,1.5377,-0.8722,-1.4085)
        rospy.sleep(1)
        move_end_effector(0.2709,-0.2501)
        rospy.sleep(1)
        move_robot_arm(-1.1759,0.1219,-1.5107,1.15,0.0194,-2.3905)
        rospy.sleep(1)

if __name__ == '__main__':
    while True:
        main()
```

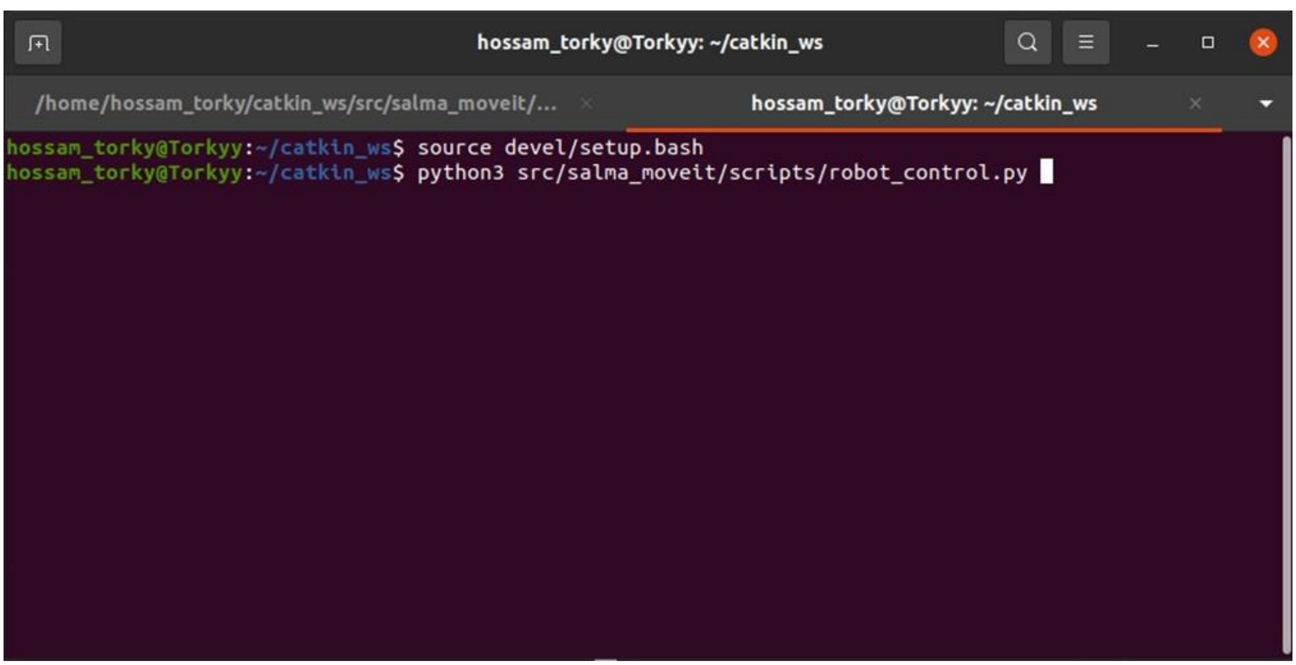
Figure 13 python script

5. Gazebo & Rviz

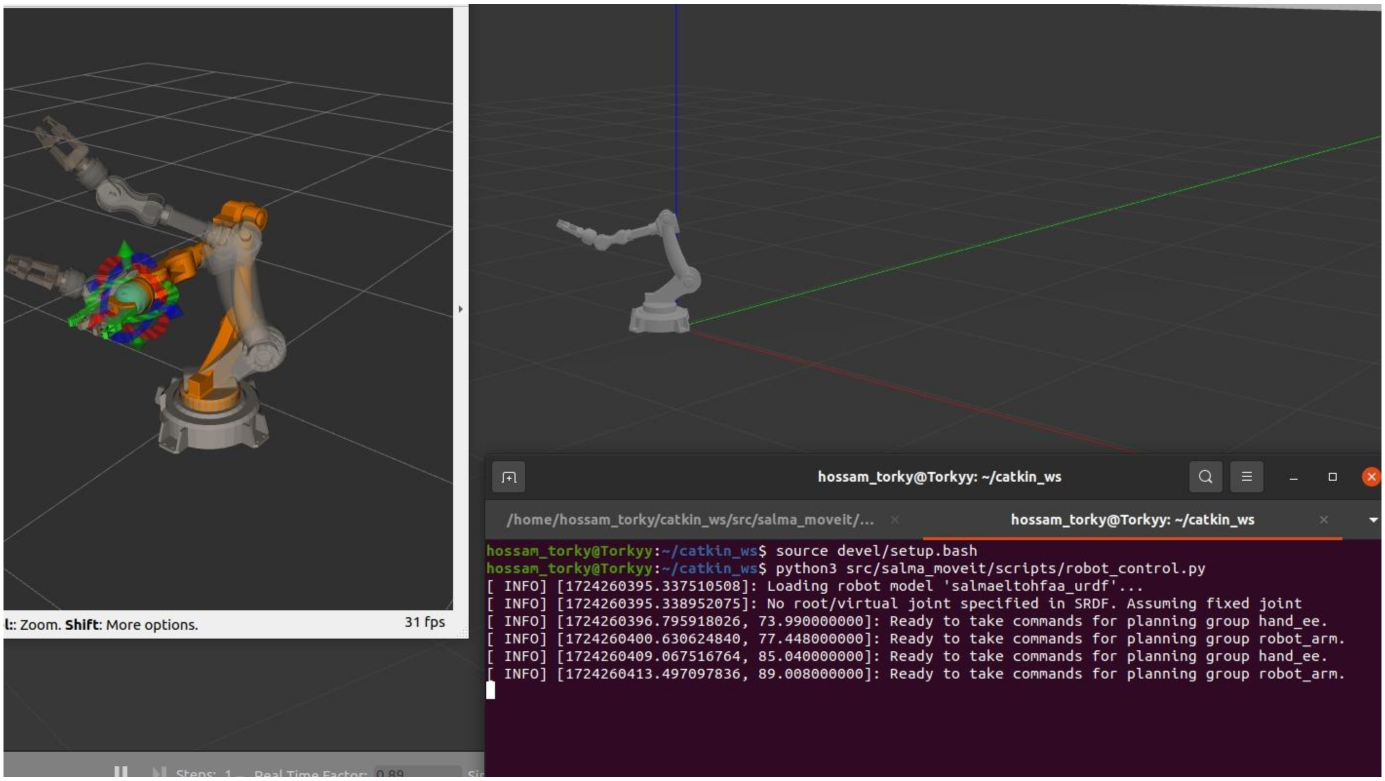
5.1. Launching Gazebo & Rviz

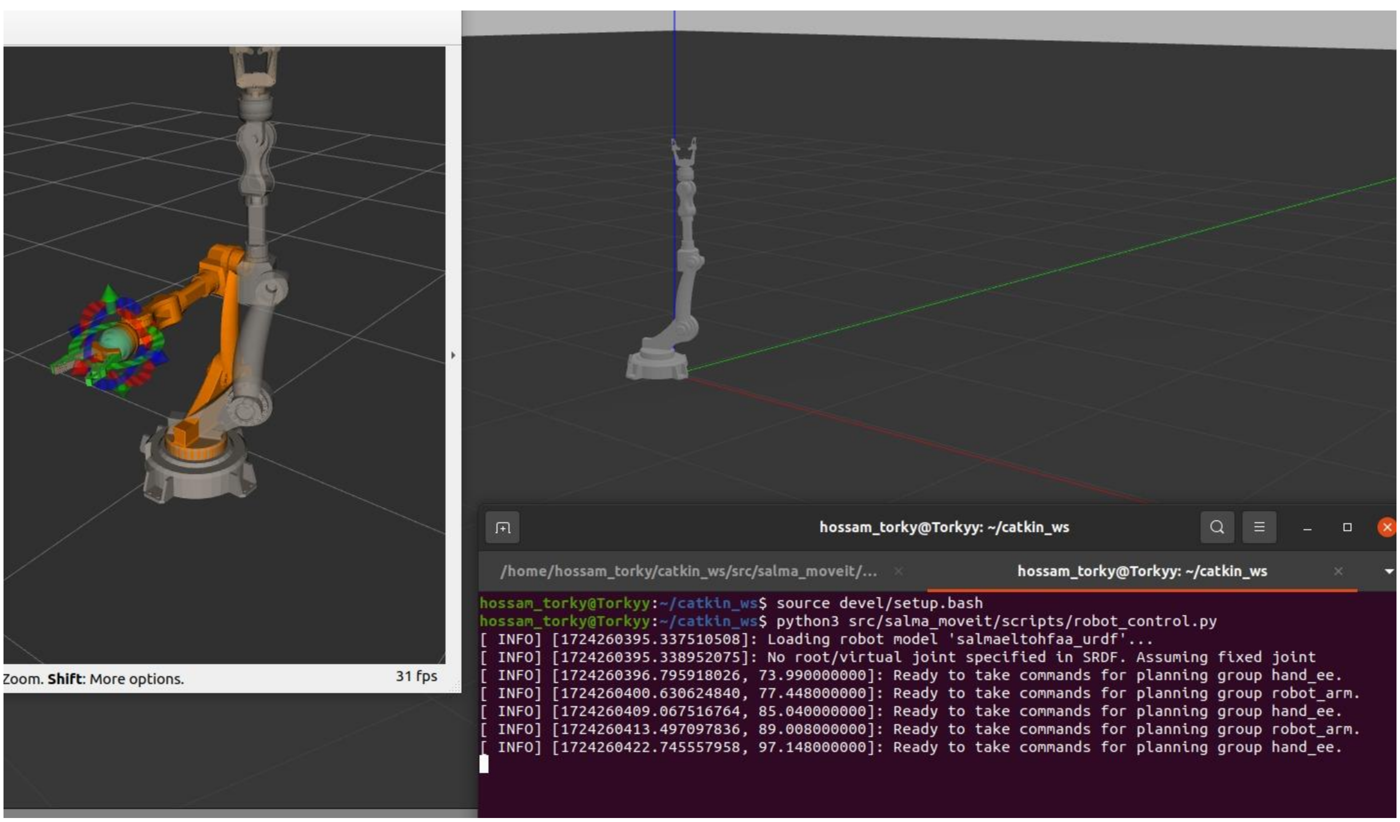


5.2. Launching Python Script



5.3. Motion in Gazebo & Rviz





6. CONCLUSION

In conclusion, this report has demonstrated the robust capabilities of the MoveIt framework in planning, simulating, and executing motion plans for a 6 degrees of freedom (DOF) manipulator. Through the integration of RViz and Gazebo, we were able to visualize and refine the manipulator's movements in a virtual environment, ensuring collision-free and efficient path generation. The Python script introduced further exemplifies the ease with which users can leverage MoveIt for various robotic applications, highlighting its flexibility and simplicity in defining models, planning trajectories, and executing complex motions.

The successful execution of these tasks underscores the effectiveness of MoveIt as a tool for reliable and efficient motion planning in robotics. By utilizing this framework, engineers and researchers can streamline the development process, reduce the likelihood of errors, and enhance the overall performance of robotic systems. This approach holds significant potential for advancing industrial automation, service robotics, and other applications where precise and adaptive motion planning is essential.

7.VIDEO & FILES LINK

https://drive.google.com/drive/folders/1jlefrTKn2qO3F6AWT4i6yDSPL_JSHrBE?usp=drive_link