CS 6341 Robotics Homework 2

Professor Yu Xiang

October 6, 2025

In this homework, write down your solutions for problems 1 and finish the coding problem 2. Upload your solutions and code to eLearning. Our TA will check your solutions and run your scripts to verify them.

Problem 1

(2 points)

Homogeneous Transformation Matrices. Exercise 3.18 in Lynch and Park, Modern Robotics.

Consider a robot arm mounted on a spacecraft as shown in Figure 1, in which frames are attached to the Earth {e}, a satellite {s}, the spacecraft {a}, and the robot arm {r}, respectively.

(2.1) Given T_{ea}, T_{ar}, and T_{es}, find T_{rs}.

(2.2) Suppose that the frame {s} origin as seen from {e} is $(1, 1, 1)$ and that

$$T_{er} = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Write down the coordinates of the frame {s} origin as seen from frame {r}.

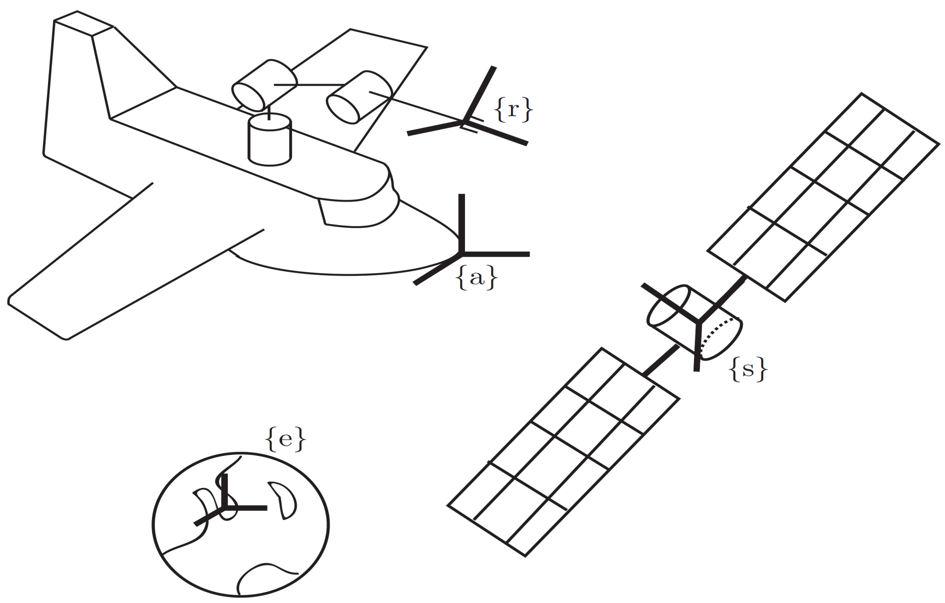


Figure 1: A robot arm mounted on a spacecraft.

Problem 2

(8 points)

ROS2 programming and Transformations.

In this problem, you will learn the homogeneous transformations in ROS2. **You can directly use Ubuntu, or Docker or virtual machine to install ROS2 according to your own set up**. Refer to the ROS2 Jazzy page if needed:

`https://docs.ros.org/en/jazzy/`.

For the following steps, start with your ROS2 workspace from Homework 1.

(2.1) Install the Gazebo simulator:

- `apt install ros-jazzy-ros-gz`
- For testing: `gz sim`

(2.2) Install and launch the Fetch Gazebo Simulator that I developed based on panda_gz_moveit. Follow the steps:

- Git clone the source code to the src folder of your ROS workspace (use the master branch): git clone `https://github.com/IRVLUTD/panda_gz_moveit2.git`
- Use the following command to install dependencies):

  `rosdep install -i --from-path src --rosdistro jazzy -y`

- Build your ROS workspace by running using symbol links to install

```
colcon build --symlink-install
```

Make sure that the worksapce is correctly built without seeing any error from the terminal.

- Start terminator with multiple windows. For Docker users, you need to start X server as we did for Rviz in Homework 1.
- Source your workspace

```
source install/setup.bash
```

- Use the following command to add the model path to Gazebo:

```
export GZ_SIM_RESOURCE_PATH=$GZ_SIM_RESOURCE_PATH:
$(ros2 pkg prefix fetch_moveit_config)/share:$(ros2 pkg prefix fetch_description)/share
```

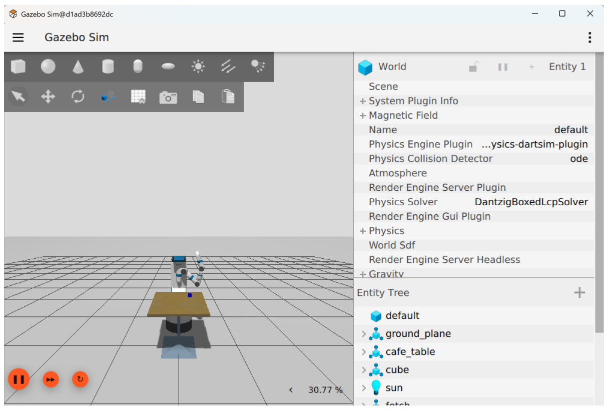
- Use one terminator window. Launch the Fetch Gazebo simulation by

```
ros2 launch panda fetch.launch.py
```

- The following command can be used to change the viewpoint of the camera in Gazebo:

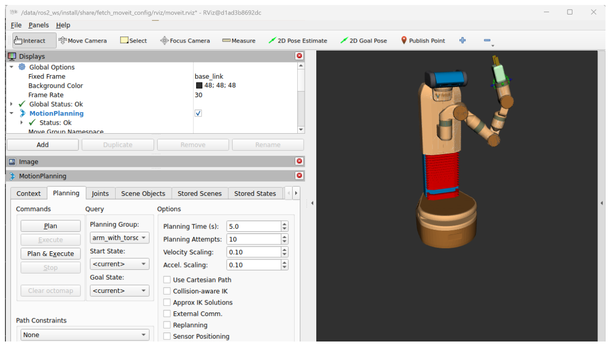
```
 gz service -s /gui/move_to/pose --reqtype gz.msgs.GUICamera --reptype gz.msgs.Boolean
--timeout 5000 --req 'pose:{position:{x:3,y:0,z:2} orientation:{x:0,y:0,z:1.0,w:0.0}}'
```

If the commands are correctly launched, you will see the Gazebo environments as in Figure 2.



Name default
Physics Engine Plugin ...ysics-dartsim-plugin
Physics Collision Detector ode
Physics Solver DantzigBoxedLcpSolver

Gazebo sim interface



Fixed Frame base_link
Background Color 48; 48; 48
Frame Rate 30
Planning Group: arm_with_torsc
Start State: <current>
Goal State: <current>
Path Constraints: None
Planning Time (s): 5.0
Planning Attempts: 10
Velocity Scaling: 0.10
Accel. Scaling: 0.10

Rviz interface

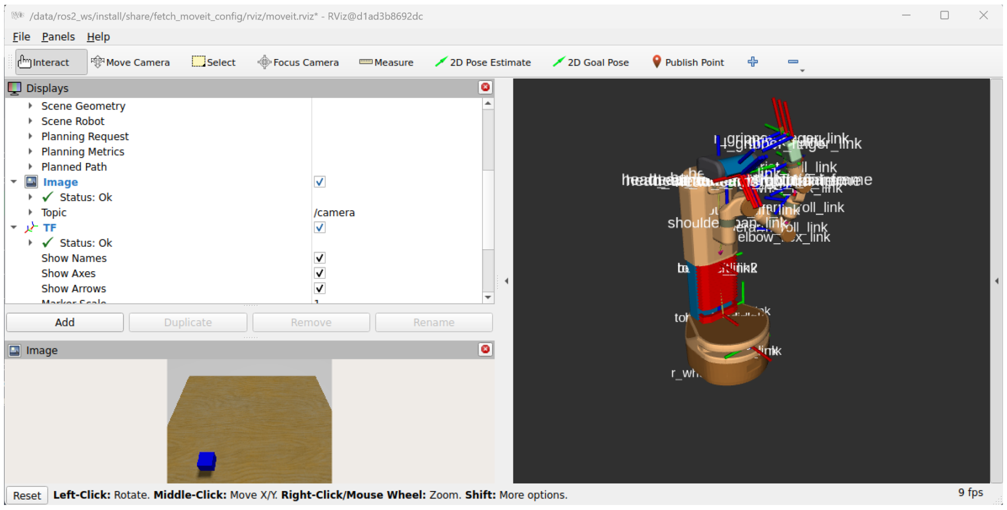
Figure 2: Two windows after launching the Fetch Gazebo simulation

(2.3) Visualize TF information with Rviz. Follow the step:

- Click the Add button in Rviz to add a TF. These are frames of different robot links.

After this step, you should see an Rviz window as in Figure 3.

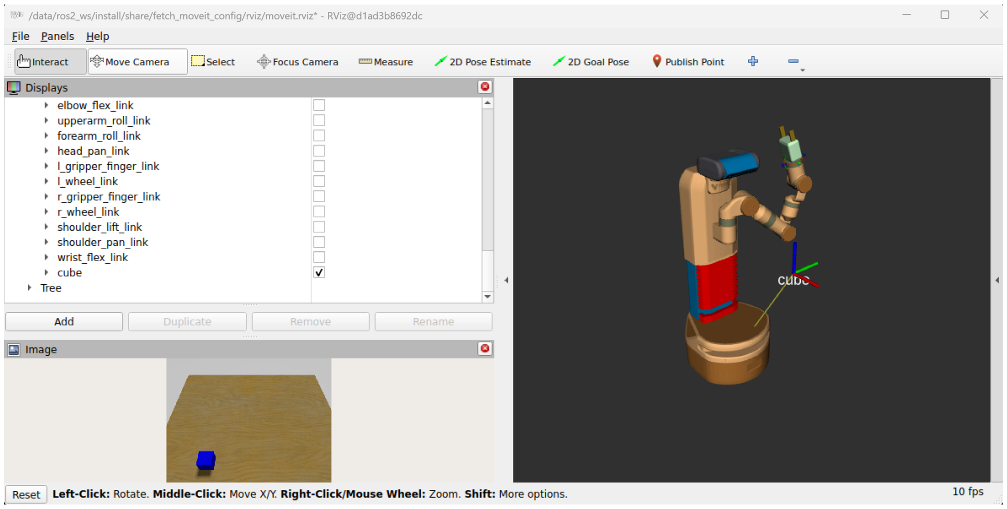
(2.4) Compute the pose of the demo cube in the Fetch base_link frame.



Scene Geometry
Scene Robot
Planning Request
Planning Metrics
Planned Path
Image ✓
Status: Ok
Topic /camera
TF ✓
Status: Ok
Show Names ✓
Show Axes ✓
Show Arrows ✓

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel: Zoom. Shift: More options.

Figure 3: The Rviz Interface.



elbow_flex_link ☐
upperarm_roll_link ☐
forearm_roll_link ☐
head_pan_link ☐
l_gripper_finger_link ☐
l_wheel_link ☐
r_gripper_finger_link ☐
r_wheel_link ☐
shoulder_lift_link ☐
shoulder_pan_link ☐
wrist_flex_link ☐
cube ✓
Tree

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel: Zoom. Shift: More options.

Figure 4: The cube pose in Rviz.

Download the visualize_block_pose.py file from eLearning. This python script first queries the pose of the cube and the pose of the Fetch base in the Gazebo world and then publishes the pose of the cube to a tf (`http://wiki.ros.org/tf`) for visualization in Rviz.

Finish the implementation of the callback function def cb() in the python script. Then you can run the python script to visualize the computed cube pose in Rviz.

```
python3 visualize_block_pose.py
```

Figure 4 shows the demo cube pose in Rviz if your implementation is correct. You might need to install the transforms3d package by *pip install transforms3d*.

Submission guideline: Upload your implemented visualize_block_pose.py file and screen captures for (2.2), (2.3) and (2.4) to eLearning.