# Lunar Builders: Robotic Arm Challenge

# Hackathon Tech Pack Manual

Date: 08th Nov. 2024

Version: 1.0

Author: Dimitris Xydas

Abstract: This document explains to participants how to get started in developing their code for a simulated robotic platform.

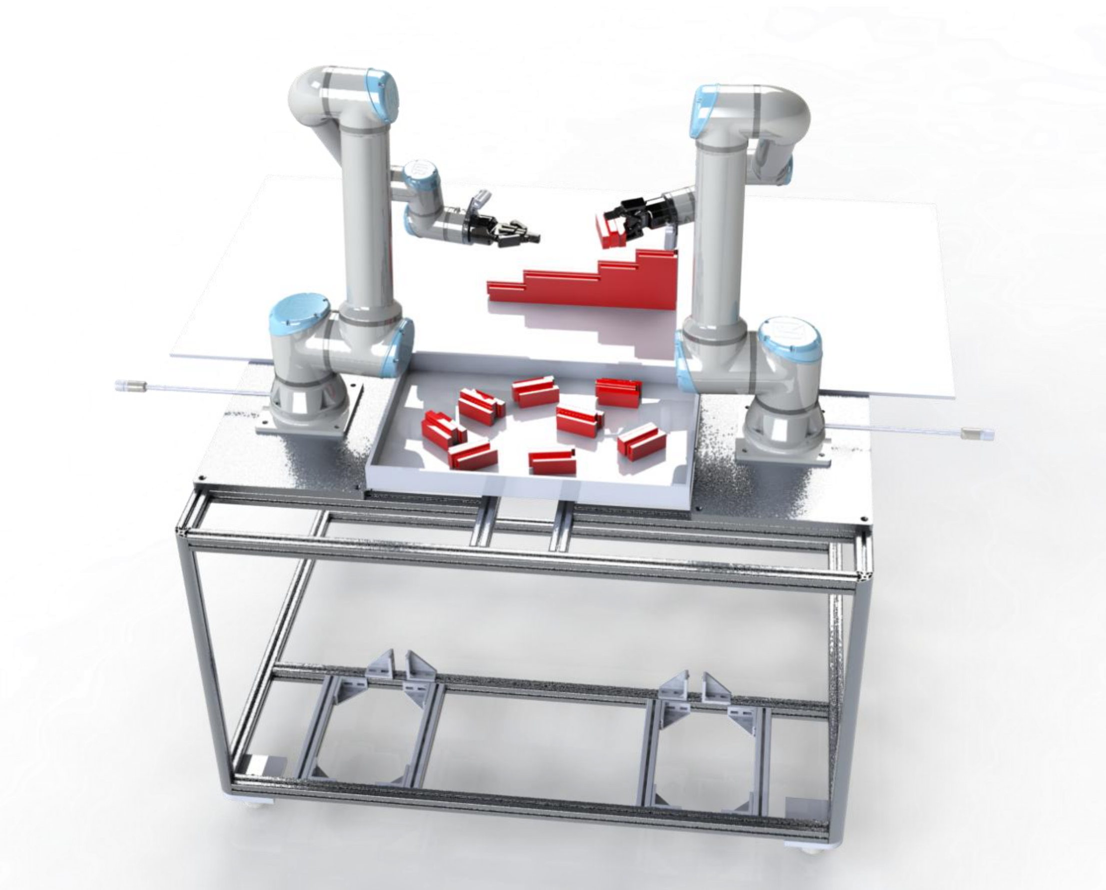
# Contents

# Version

| Revision | Date | Description |
|---|---|---|
| 1.0 | 08th Nov. 2024 | This is the initial release version. |

# 1.0 Introduction

Welcome to the Lunar Builders: Robotic Arm Challenge, co-organised by the UK's Satellite Applications Catapult (SAC) and the European Space Agency's Spaceship from the European Center for Space Applications and Telecommunications (ESA Spaceship ECSAT).



This document will provide you with a guide to get quickly started with your development in a simulated environment.

# 2.0 The System

Hardware components that are simulated:

- 2x UR10e robots
- 2x Robotiq 2f-85 grippers (with e-series coupling: GRP-ES-CPL-062)
- 2x Intel RealSense d435i cameras
- Regolith simulation blocks, simulated in Gazebo
- Astronaut model

Software:

- Ubuntu 22.04 LTS
- ROS2 Humble LTS binaries – standard installation
- ROS2 drivers for components – see details below
- Gazebo Classic

# 3.0 Getting Started

For this challenge we will utilise a number of pre-existing libraries provided by the hardware manufacturers, modified for convenience to set up a customised version, which you can then further modify as you like for your own development.

This challenge relies heavily on the ROS2 framework in order to get started quickly with development in both a simulated environment and with real hardware, as the ROS drivers allow an interface with a number of real robots, and the wider ROS ecosystem provides a number of simulation tools when hardware is not available.

If you are unfamiliar with some of the basic concepts, your best starting point is to read the documentation on the next major topics:

- ROS 2 Documentation
  https://docs.ros.org/en/humble/index.html
- MoveIt 2 Documentation
  https://moveit.picknik.ai/main/index.html
- Getting Started with Gazebo?
  https://gazebosim.org/docs/latest/getstarted/

Next, it is strongly recommended that you install and follow the getting started instructions of the following default libraries, before installing the customised version, to ensure you have the major dependencies in place and you are familiar with the ROS2 system.

It is strongly recommended to use choose the 'humble' branch of code, wherever that is available, as it ensures best compatibility within this documentation (as well as with our own hardware, if you end up deploying your code at Phase 2). If your lab already has these set up beforehand, you can skip ahead.

Repositories:

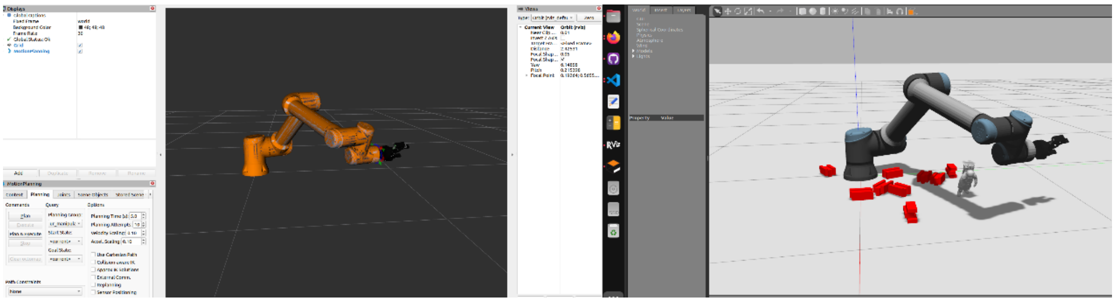
| Name | Author | Branch | Description |
|---|---|---|---|
| Universal_Robots_ROS2_Description | Universal Robots A/S | humble | UR robot description files<br><br>Package ur_description contains the robot description URDF and related files |
| Universal_Robots_ROS2_Driver | Universal Robots A/S | humble | UR robot driver<br><br>Package ur_moveit_config contains MoveIt SRDF files |
| ros2_robotiq_gripper | PickNik Robotics | humble | Robotiq gripper driver<br><br>Package robotiq_description contains the gripper description URDF and related files |
| Universal_Robots_ROS2_Gazebo_Simulation | Universal Robots A/S | humble | UR Gazebo simulation setup<br><br>Package ur_simulation_gazebo sets up Gazebo simulation for UR, with and without MoveIt |

Once you have reviewed and tested the above libraries (or have already done so before), you can clone the public repository set up specifically for this challenge, at the following link:

https://github.com/SatelliteApplicationsCatapult/isam-hackathon-2025-setup

The landing page (README.md file) will gives you the setup instructions, which should be straightforward.

Once the launch file ‘custom_ur_sim_moveit.launch.py’ has launched and initialised successfully, you will be presented with two windows similar to this:



On the left-hand side you have the RViz control of the arm, which provides a GUI for quickly planning and executing trajectories with MoveIt’s Motion Planning framework.

On the right-hand side, you have an example Gazebo world which simulates the behaviour of the arm, and its interaction with components. In this example world we have provided 10 regolith bricks and an astronaut model to get you started.

# 4.0 Upgrades

The above instructions should allow you to get started with this challenge fairly quickly with minimal restrictions. You may have noticed however that a number of things need your attention.

For starters, we only provided you with a single-arm implementation, customised with a Robotiq gripper attached. How to create and attach another robot to the world is down to you, but this MoveIt tutorial by PikNik Robotics is a good starting point if you have never built a multi-robot ROS system before.

Next, you may notice that the implementation used here is based on Gazebo Classic, because it what UR provides out-the-box.

The recommended Gazebo installation for ROS Humble builds is Gazebo Fortress, but it is up to you to pick what works best for you. You can read more about the differences between Gazebo Classic, Ignition and Gazebo here.

Of course, it's also entirely up to you if you want to forego Gazebo entirely and use the simulator of your choice!

Next, you will notice that if you move the gripper, it will 'snap' open/closed, using either the provided MoveIt planning group or a GripperCommand action:

- Open (position: 0.0):

```
ros2 action send_goal /robotiq_gripper_controller/gripper_cmd
control_msgs/action/GripperCommand "{command:{position: 0.0, max_effort: 100.0}}"
```

- Close (position: 0.8):

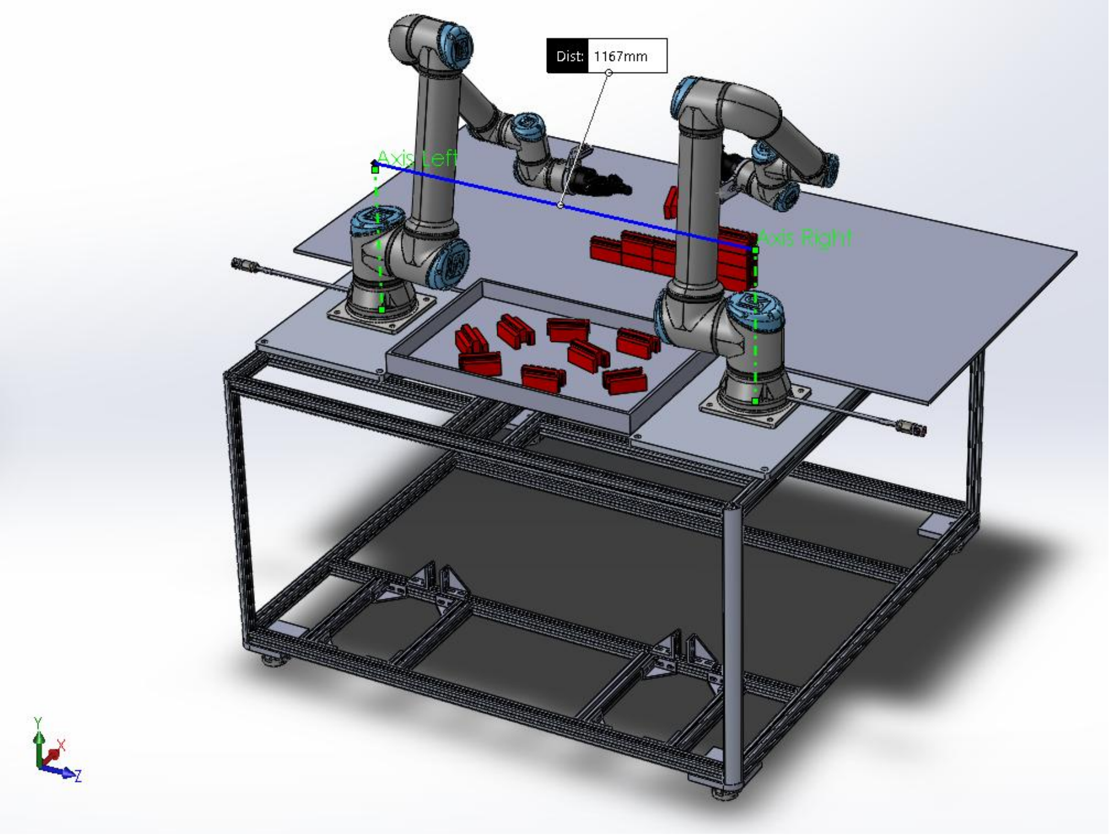
```
ros2 action send_goal /robotiq_gripper_controller/gripper_cmd
control_msgs/action/GripperCommand "{command:{position: 0.8, max_effort: 100.0}}"
```

This is a limitation of the default Gazebo controller, so it will need to be upgraded or circumvented in order to smoothly close the gripper and pick up the regolith blocks successfully.

Finally, the reason the gripper is shown as disconnected from the UR, is to account for the coupling and camera adaptor present in our hadware, hence you will notice the 24 mm distance coded in the custom 'custom_ur.urdf.xacro' description under the folder 'custom_ur_description/urdf/'.

## 4.1 Second arm placement

It is recommended that, in order to maximise compatibility with our real hardware setup, you position the robots as shown here:




Standing behind the robots looking 'forwards', we use the following convention:

- X: Pointing 'forward'
- Y: Pointing 'left'
- Z: Pointing 'up'

Using this convention, the right robot is naturally pointing forward, as is by default for its base link called 'base_link' which is REP-103 aligned (i.e., has X+ forward, Y+ left and Z+ up), not to be confused with the UR's base frame (see this point in the UR code which clarifies it).

The Left robot is then spun 180° around, providing a symmetric system.

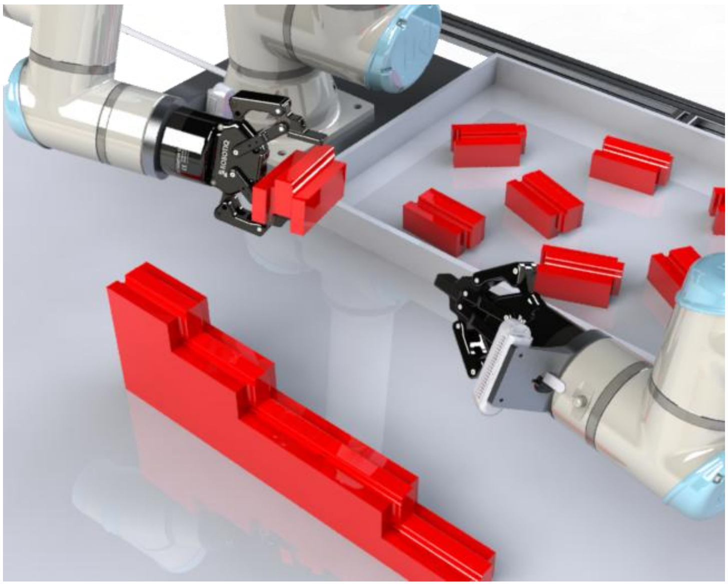
**Ensure the inter-base distance between the robots is 1167 mm.**

## 4.2 Simulation parameters

In the provided code, under the 'isam_hackathon_2025_main' folder we have provided the 'worlds' and 'models' folders where the template world and brick/astronaut models are stored respectively.

You can configure the world and models to your liking, as you can tweak the simulation parameters as required. You can even experiment with turning gravity off!

The regolith block example provided has a specific shape that allows easier interlocking between blocks:



The file 'isam_hackathon_2025_main/models/brick/model.sdf' allows you to configure inertia and collision parameters.

The collision boundary box matches the visual representation based on the provided STL file, however you will notice the inertial parameters have been simplified, to treat the block as a rectangle with dimensions of 60 x 40 x 100 mm and weighing 0.3 kg (the approximate weight of such a block 3D printed in PLA).

# 5.0 Get building

From here on it's up to you to, review the challenge rules in the detailed document here, get creative, and get building!

Of course if you can demonstrate anything on your own existing real hardware, that would be a plus, but the core focus of this hackathon is a simulated environment, so there is no need to demonstrate deployment on real hardware for Phase 1.