RoboCup Autonomous Robot Manipulation (ARM) Challenge

Supported by MathWorks

Participation Guide

Challenge Overview

Problem: In this challenge, your goal is to program an intelligent robot manipulator to identify and sort objects using vision data in dynamic virtual environments.

Prizes:

- **1st place:** RCF Grant for Robotics/AI related research or public intellectual contributions (Up to $5000 USD)
- **Top 4 teams:** Invite to participate in the RoboCup 2024 Finals (Eindhoven) using an in-person setup with real Universal UR5e robots.
- **All teams:** Certificate of participation and skills developed in robotics manipulation algorithm development.

Simulation Environment: The manipulation environments used for this challenge are Gazebo worlds derived from the MathWorks documentation example Pick and Place Robot Manipulator. A virtual machine is used to run the RoboCup competition environments, instructions on installing this virtual machine can be found on the GitHub repo here. The robot manipulator used is a Universal UR5e. It is equipped with a RGB camera and a depth camera which both can be used when solving the challenge.

Rules:

- Participants must register in a RoboCup league
- Submissions must contain reproducible MATLAB and/or Simulink files. Any necessary file dependencies must be included.
- Robot perception algorithm must not use environment information available from the Gazebo API
- Any movement induced to the game objects must be a reaction to interactions with other environments components (Do not use Gazebo API to set positions or states)
- Modifications to the Virtual Machine Image and Gazebo worlds provided is not suggested as code will be evaluated under the same conditions provided in test environments
- Robot start position must be the same as in the Templates for RoboCup Virtual Robot Manipulation Challenge

Scoring

- Your model will be scored using the scoring rubric found on the last page.
- During the semifinals, your submission will be graded on an **unknown 3rd world** not provided in the test environments.
  - Top 4 submissions from the qualification round will compete in the finals
- During the finals, your submission will be scored based on an in-person setup at the RoboCup finals, that will feature similar objects. Any new conditions will be provided to qualifying teams.
- Objects will not be reset during judging - If an object is knocked over, your algorithm should account for this.

Awards:

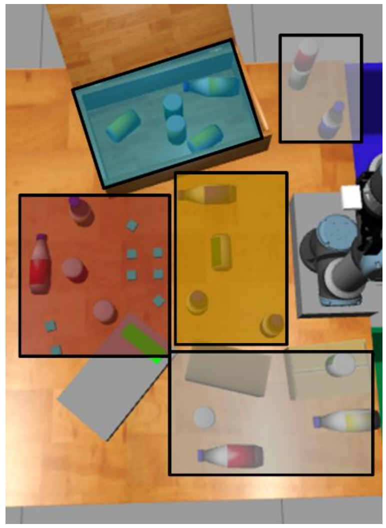
- **All teams:** Certificate of Achievement for demonstrating skills in programming, control, simulation, and perception of robotic pick-and-place and grasping applications in dynamic environments.
  - Subject to a minimum score of 60 points
- **Top 4 teams:** Invite to RoboCup 2024 Finals (Eindhoven) for top 4 scoring teams during the qualification round
- **1st Place:** Winners of the finals will win a **grant for a Robotics/AI related research** project or publish intellectual contribution to the field for the 2024-2025 season
  - The winners are guaranteed a grant but must submit a brief project description to the RoboCup Federation to confirm the funds will be allocated fairly towards robotics/AI related projects.

Important Dates:

Registration Opens	February 15th
Live Challenge Kickoff and Q&A	March 13th
Technical Support and Development Q&A for teams	April 18th
Submission deadline for semifinals round	May 26th
Announcement of teams qualified for finals	May 30th
In-person finals (Eindhoven - Netherlands)	July 18-21 (During RoboCup Finals)

Item Layout Conditions

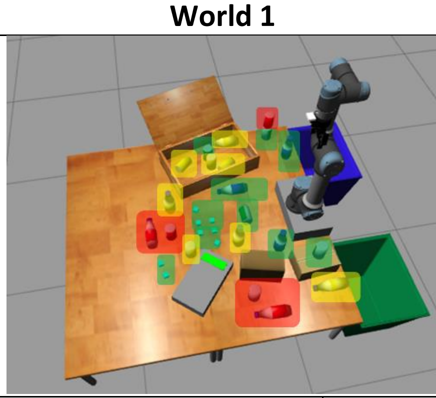
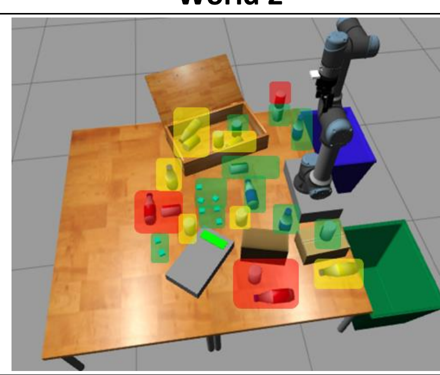
Items present in the virtual world will follow the conditions outlined below in the unknown environment that will be used for scoring.



	Won't change – Items will remain the same. No changes to orientation, position, or shape.
	Shape change – Items may switch color and item type (from can to bottle and vice versa). No changes to orientation or position.
	Random item bin – This bin will contain the same items (1 green can, 3 yellow cans and 1 yellow bottle), however the items may change position or orientation.
	Orientation change – Items will remain in the same position but may change orientation. I.e., rotated

Example worlds

The two provided example worlds show how the layout conditions can affect the unknown environment used for testing the submitted algorithm.

World 1	World 2
	

	10 Points		20 Points		30 Points

Getting Started

Simulation Environment and Virtual Machine

Gazebo and the world files are all pre-installed in an RCF provided virtual machine. Go to this link and use VM Ware Player to launch the virtual machine. Detailed instructions are also provided in Appendix A at the end of this document.

With the virtual machine setup and launched you can launch gazebo and the docker container by navigating to the "~/src/arm_gazebo/docker" folder and executing the "./run.bash" script.

Programming the Robot with MATLAB and Simulink

To help you get started, templates on how to control the manipulator using MATLAB and Simulink are provided. These are just examples on how to control the manipulator, the models don't have to be used in your submission. These templates use ROS to control the robots.

The Simulink model contains a 'Gazebo Pacer' block. This block can be used to sync Simulink and Gazebo to run at the same simulation time. However, this block isn't required to use the Simulink model. See this documentation page for more information.

MATLAB and Simulink Learning resources:

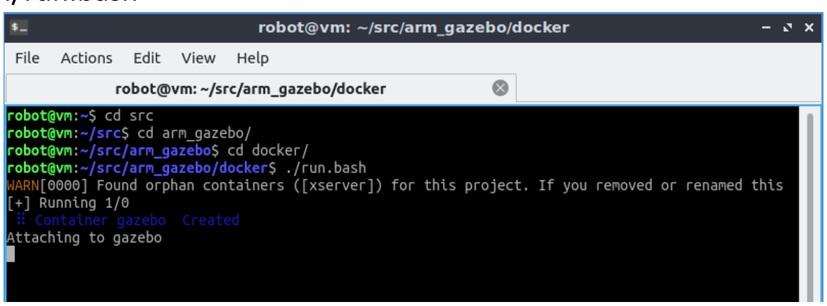
- Templates for RoboCup Virtual Robot Manipulation Challenge
- MATLAB Academy (MATLAB/Simulink/Stateflow/Deep Learning Onramps)
- MATLAB and Simulink Robotics Arena
- MathWorks Support for RoboCup (Request complimentary license)
- MATLAB and Simulink Documentation
- Questions on MATLAB and Simulink? roboticsarena@mathworks.com

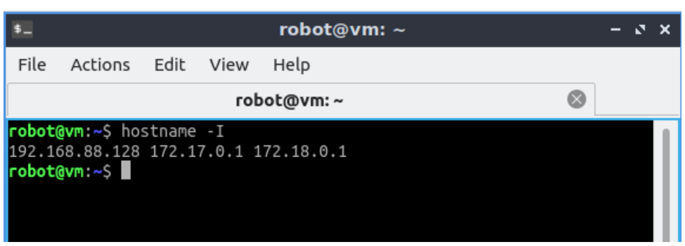
Scoring Rubric

Submitting Models *(40 points)*	*Points*
All files needed to run algorithm are included in submission. Recommendation: Use a MATLAB or Simulink project, and verify using MATLAB Dependency Analyzer or Simulink Dependency Analyzer	10
Video of algorithm working is submitted.	10
Code is reproducible without any errors.	20
Using Sensors *(80 points)*	
Effective use of RGB camera. Data processed from the RGB camera leads to successful classification of at least one bottle or can.	40
Effective use of depth sensor. Data processed from the depth camera leads to successful classification of at least one bottle or can.	40
Placing Items in Correct bins *(460 points)* *Cans go in the Green bin* *Bottles go into the Blue bin* *Detergent pouches go in the Green bin*	*(Some objects can be different shapes according to environment rules)*
Green Can	10 per can *(3-4 cans available)*
Yellow Can	20 per can *(3-4 cans available)*
Red Can	30 per can *(3 cans available)*
Blue Bottle	10 per bottle *(3-4 bottles available)*
Yellow Bottle	20 per bottle *(3-4 bottles available)*
Red Bottle	30 per bottle *(2 bottles available)*
Pouches	10 per pouch (8 pouches available)
Deductions • Item placed in wrong bin: - 10 • Item falls on floor: - 5	
Bonus Points (180 points) *Points will be awarded regardless of whether the item is placed in a bin.*	
Robot lifts at least one item from table in the Shape change region (Awarded once only)	10
Robot retrieves at least one item from random item bin (Awarded once only)	20
Single pouch is released and then picked up from the weighing scale (Awarded per pouch)	15 (per pouch, max 150 total)
Minimum Points for Certificate	**60**
Total Possible Points	**760**

Appendix A – Detailed Instructions for Simulation Environment

Game Environment Installation Instructions

1. Recommended: Download and install VMware or VirtualBox virtual desktop.
    a. Alternative: If you are familiar working with Docker containers or want to work in Linux, you can clone this Git Repository and build the Docker container.
2. Download the RCF-ARM Virtual Machine image from this link
3. Setup virtual machine settings before running:
    a. Make sure you assign enough ram to the virtual machine (Recommended ~4GB)
    b. Make sure to assign enough display memory (Recommended)
    c. Setup the Network Adapter to "Host-Only". Check this VirtualBox setup instructions if it is not already setup after installation.
4. Run VM, open a terminal and navigate to the following folder:
    a. ~/src/arm_gazebo/docker
5. Execute the following command to run the docker container:
    a. ./run.bash
    b. 

    c. A window with the Gazebo environment should open
6. Find the respective IP address of the ROS MASTER
    a. Open a separate command terminal and run the following command to find the hostname IP address of the Gazebo environment:
        i. hostname -I


        ii. Use the hostname IP address as the ROS_MASTER IP address in the MATLAB and Simulink templates for controlling the robot along with **ROS port 11311**. Try different network configurations since networking configurations can differ between computers. Here are some options:
            1. Use a "NodeHost IP Address" parameter to specify a local IP address in your MATLAB code or Simulink model in the host computer

2. Try different networking settings in the Virtual Machine like, NAT, Host-based, Bridged

Update RoboCup ARM Virtual Environments

If the challenge organizers push an update to the test environments, do the following to update them to the latest version:

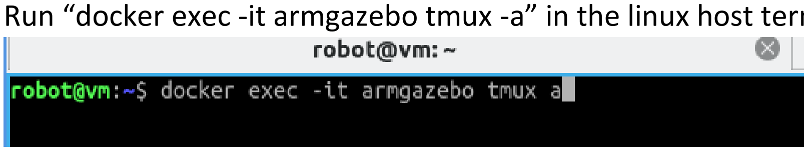
1. Verify the Virtual machine has access to the internet. If necessary, change the Network settings of the Virtual Machine to “NAT: Used to share the host’s IP address”
2. Run the virtual machine
3. Open a command terminal and navigate to the “arm_gazebo” folder
4. Execute the command “git pull” and wait for success
   a. Try different network settings if needed, verify the VM has access to the internet.
5. Power off the Virtual Machine and reset the network settings as needed.

Update Docker Image

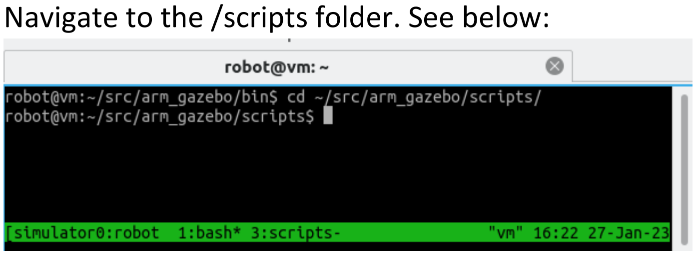
1. Navigate to the arm_gazebo folder and run “git pull”
2. After successful git pull. Navigate to the docker folder
3. Run “./build.bash”
   a. If you get an error stating “Release is not valid yet for …. hours”. Use this link to update the clock on the system and the packages.
4. Try “build.bash” again. This can take a quite a while. Be patient and check the %progress

Open Different Test Environments

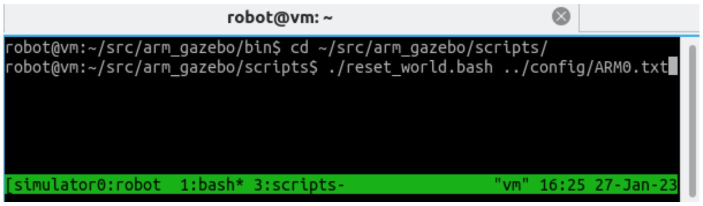
1. Open a new terminal in the virtual machine or linux environment
2. Attach a tmux session to the container (tmux allows you to open shells within the container and run commands). Here is how:
   a. Run “docker exec -it armgazebo tmux -a” in the linux host terminal



   b. Press the following while in the terminal to open a fresh command window in tmux:
      i. Ctrl+b, then press the “c” keyboard key right after
   c. Navigate to the /scripts folder. See below:



d. Run either of the following commands to reset the world objects to any of the two provided configurations.
    i. "./reset_world.bash ../config/ARM0.txt"
    ii. "./reset_world.bash ../config/ARM1.txt"



e. To end tmux session, press ctrl+b, the the letter "d" followed right after.