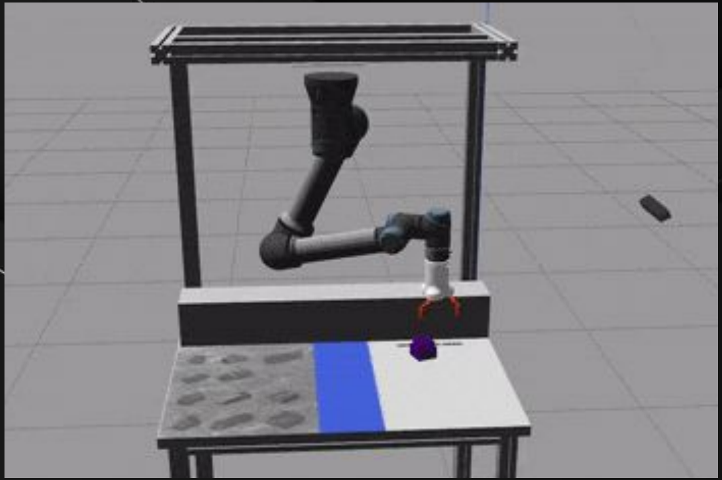
Fundamentals of Robotics project

Group K

Authors:

- Conti Filippo [218297]
- Gianuzzi Nicola [209309]
- Meneghin Mattia [210561]



GitHub repository:

https://github.com/LordBions/Robotics_ICE23_UNITN

University of Trento

Fundamentals of Robotics project

Project Overview - Proposal 1

A number of objects are stored randomly on a stand located within the workspace of a robotic manipulator (UR5), an anthropomorphic arm, with a spherical wrist and a three-fingered gripper as end-effector.
The objects can belong to different classes but have a known geometry.
The goal of the project is to pick the objects using in sequence and to position them on a different stand using a calibrated 3D sensor to locate the different objects and to detect their mutual position in the initial stand.
The project is organised as a sequence of assignments of increasing complexity.

Fundamentals of Robotics project

Tool used

ROS noetic

Locosim

Gazebo & Rviz

Catkin

Yolov5 & Pytorch

Programming Languages used

C++

Python

Bash

University of Trento

Fundamentals of Robotics project

Structure of the Project

We decided to split the project into 3 packages:

1. Environment
   1.1. Spawner Lego
   1.2. Lego models

2. Motion
   2.1. Planner
   2.2. Movement

3. Vision
   3.1 Vision.py
   3.2 RecogniseLego
   3.3 RecogniseArea

Locosim (light version)

1 Environment

This module is in charge of launch all the environment.
It will open Gazebo and Rviz using the didactic framework **Locosim**.
In this way the UR5 is ready to do the tasks that will be assigned from the planner module. This module is mostly done by the author of this framework, we edit the models in the *worlds* folder in order to **customize the world**

- different version of the lego models
  - from 1.4 to 1.6
- Mesh on the table with the object silhouette
- Custom lego.world

1.1 Spawn Lego

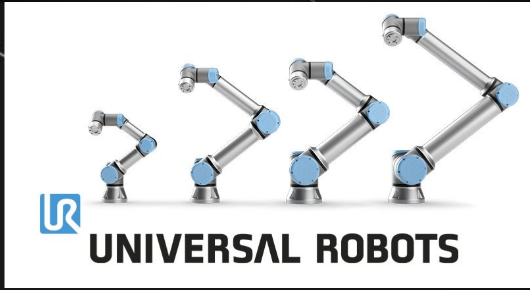
- Generates the lego objects on the table
- Possibility to set position, orientation and distances between objects)
- Capability to select the assignment number and also special features
- Parameters:
  - No parameter
  - -a[assignment] where [assignment] = {1, 2, 3, 4} (e.g. -a2)
  - -s1 (beta)
    - Wait for some external spawn commands (for testing)
  - -s2 (beta)
    - Spawns lego and removes them after a certain delay
    - Used to take pictures

Fundamentals of Robotics project

2 Motion

It is the package in charge of handle the movement and tasks to do of the Universal Robot 5, it is composed by several modules and systems:

1. Planner
2. Movement
3. Kinetics
4. Task Waiting System
5. Acknowledgement System



University of Trento

Fundamentals of Robotics project

2.1 Planner

- Communicates with the **vision** module with acknowledgements capability:
  - vision publisher → planner subscriber
- Get info about the object detected through ***legoFound.msg***
  - (e.g. Coordinates, orientation, class)
- 3 available commands from vision
  - 0 - No Command
  - 1 - Detect
  - 2 - Quit planner
- With a given class, it defines the final coordinates and orientation

Fundamentals of Robotics project

2.2 Movement

- Communicates with the planner module with acknowledgement and authorization capability.
    - planner publisher → Movement subscriber
- It receives the ***legoTask.msg*** from the Planner and gets the info about:
    - command to execute without caring about the class
        - most frequent is the 0: *catch_obj*
    - Initial coordinates and final coordinates
    - Diameter gripping and the final object position [optional]
- It answers to the planner through a **eventResult.msg**
- Inside the movement.cpp there are basic instructions to move the UR5 robot referring to a library called **Kinetics**

2.2 Movement commands from the planner module

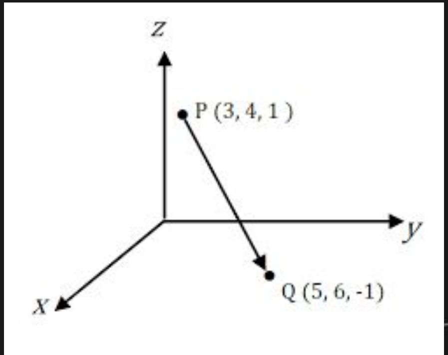
0 - No Command
1 - Test
2 - Wait
3 - Move
4 - Grasp
5 - Ungrasp
6 - Default position
7 - Fast catch
8 - Catch

Security system commands:

9 - Handshake
10 - Auth key

Fundamentals of Robotics project

2.3 Kinetics



- Kinetics is the library that contains all the primitive instructions in order to calculate *angles*, *spatial displacement*, *matrix* and all the *mathematical functions* related.
- It includes the necessary classes and functions for dense matrix operations, it provides a wide range of functionalities, including matrix algebraic operations.
- We handled the Euler's singularities by avoiding the arm to pass in these critical areas with specific commands.

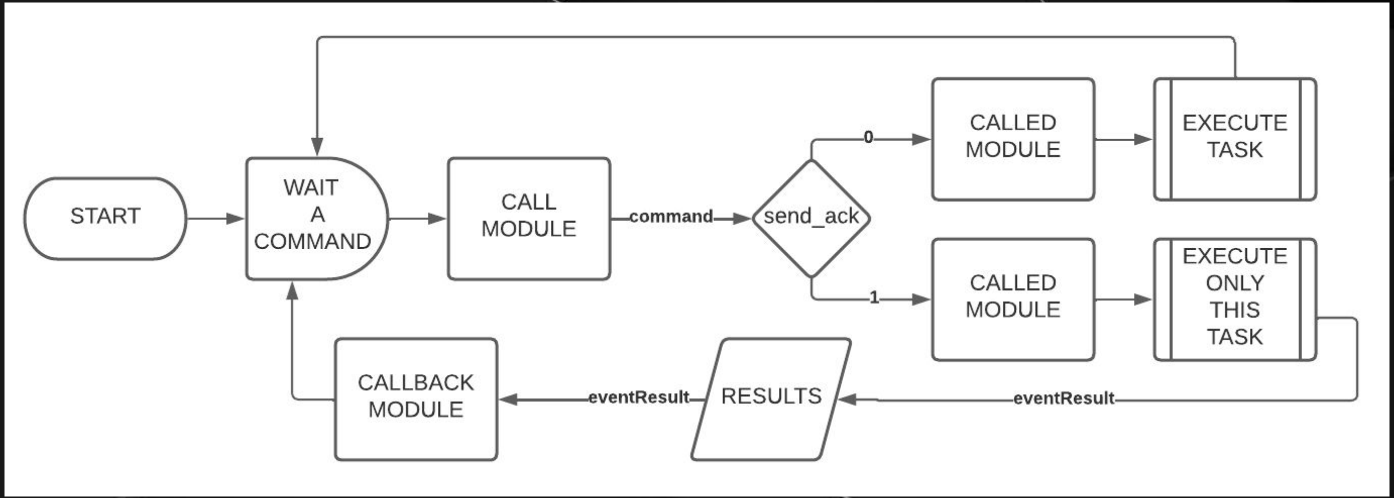
University of Trento

2.4 Acknowledgement system

- It allows you to choose the operating mode
- Since the communication between two different modules is done through messages with a certain *command IDs*, each of these is defined in order to give back a variable called **send_ack**.
- If you want an execution results back to the *caller module*
  - send_ack must be equal to **1**.
  - It enforces the *called module* to complete the requested command and **send back** the results via event message.
- If you just need the execution without caring about its completions,
  - leave "send_ack" equal to **0**

Fundamentals of Robotics project

2.4 Acknowledgement system flow

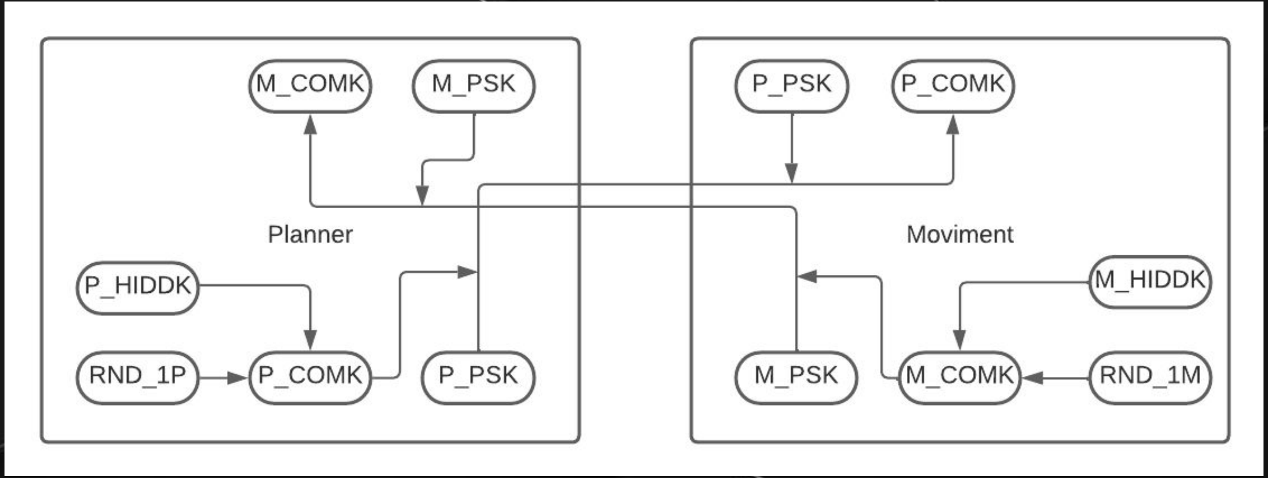


2.5 Secure communication system

The planner can be rosrun with the -s parameter which enables the authorization system: In the first time will be an handshake procedure where the planner and the module share their keys in a encoded channel. One ready, this system prevents to send commands to the movement module without a valid authorization code that changes over the time.

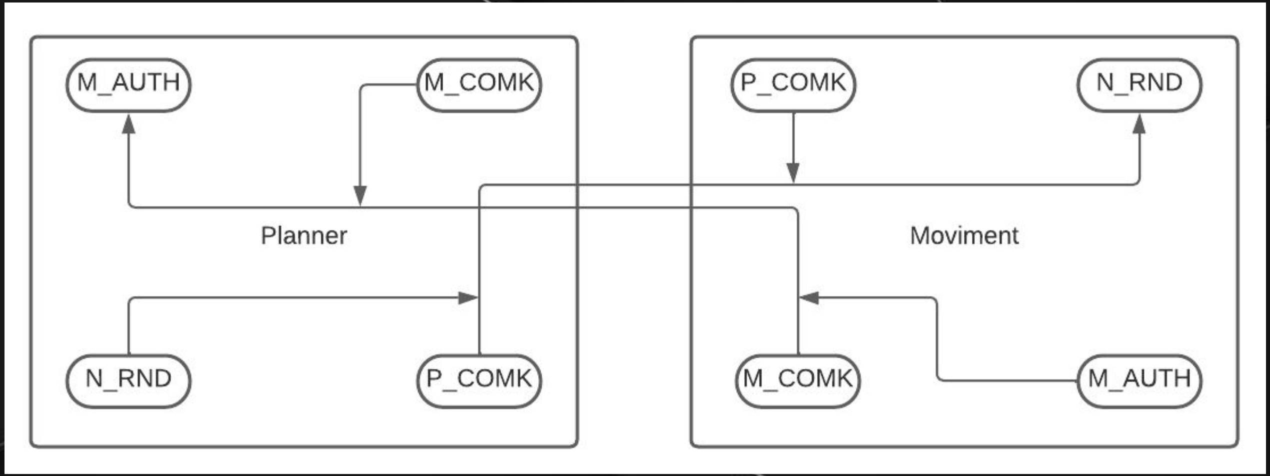
Fundamentals of Robotics project

2.5 Secure communication system diagram: Handshake part 1



Fundamentals of Robotics project

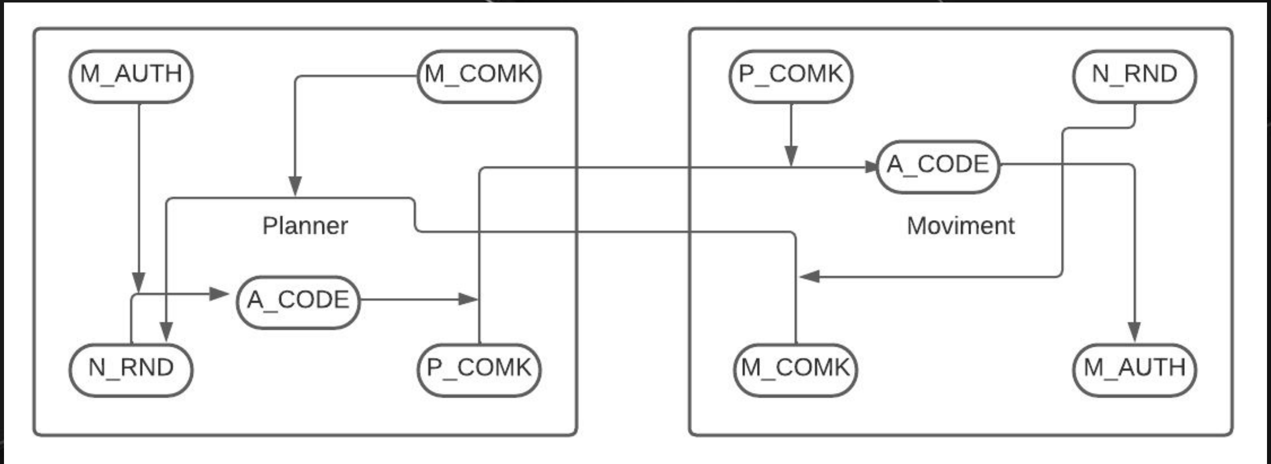
2.5 Secure communication system diagram: Handshake part 2



University of Trento

Fundamentals of Robotics project

2.5 Secure communication system diagram: On work



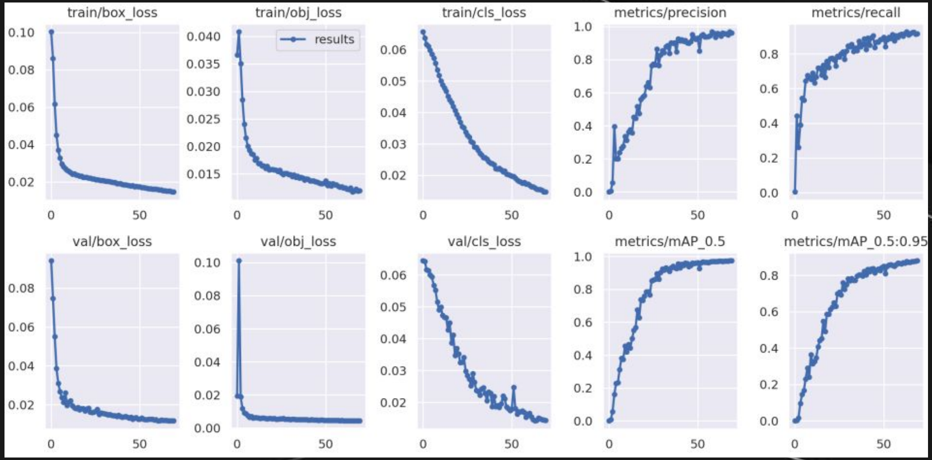
Fundamentals of Robotics project

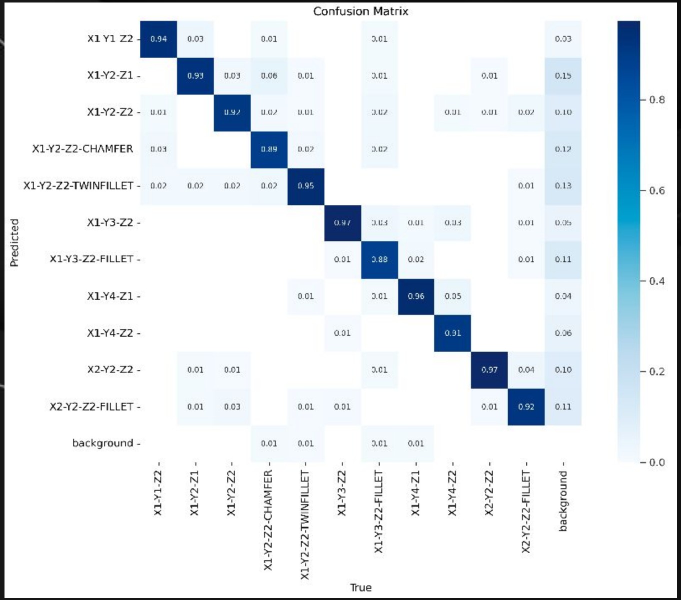
3 Vision

- In charge of practice the deep learning to create an “artificial neural network” that can learn and make intelligent decisions on its own
- The goal is to make ZED-Camera able to recognize the 11 different models (legos) starting from a dataset (pool of ~2500 images).
- Tool used:
  - **MakeSense AI** Get annotations
  - **Roboflow** Create and refine the dataset
  - **Yolov5** (Ultralytics) in Colab Train
  - **Pythorch** Test in practice

Fundamentals of Robotics project

Vision Results





Vision detection misses

- We noticed that there are some misses during the detection, in particular running the assignment 2
  - Rarely it misses the objects (no detection)
  - sometimes it recognizes the wrong class
    - especially in case of similar geometry shapes (i.e. class 3 - 6)
    - The main reason is the point of view of the camera
- Of course it is quite impossible to get 100% of precision detection
- The solution is to increase the number of annotations of the **most frequent objects missed** in the dataset
  - As shown in the previous slide, we got **~97%** of accuracy
  - so that we decide to keep this acceptable values

Fundamentals of Robotics project

Run The Project

- Move into the repository folder, then catkin_ws
  - `cd Robotics_ICE23_UNITN/catkin_ws/`
- Build the packages
  - `catkin_make install`
- Run the **start.sh** script using
  - `bash ~/Robotics_ICE23_UNITN/start.sh`

```
:: Foundamentals of Robotics project        ::
:: Authors:                                 ::
::     - Conti Filippo                      ::
::     - Gianuzzi Nicola                    ::
::     - Meneghin Mattia                    ::

Starting the ENVIRONMENT

:: Note: Wait the end HOMING PROCEDURE      ::

Once HOMING PROCEDURE ACCOMPLISHED, check the correctly start of the Environment
Press [Y] to continue
Press [n] to restart the environment
Continue?
```

Note: Necessary to wait the accomplishment of the homing procedure in the “environment” terminal

```
Starting the VISION module
Starting the MOVEMENT module
Starting the PLANNER module

ALL STARTED - SECURE WAY

Y to re-call some modules?
k - Kill all the ROS node
r - If you want to restart
R - If you want to rebuild and restart
```

Run The Project

- Select the number of the assignment or EXIT
- Select the operating way [**S**/**N**]
- Otherwise you can
  - EXIT
  - Restart
  - Rebuild and then restart

```
    :: Please insert the assignment number:        ::
    [1] Assignment 1
    [2] Assignment 2
    [3] Assignment 3
    [4] Assignment 4
    [K] EXIT
Choose an option: 2
Will be execute the assignment [2]
Starting the SPAWNER module

    :: Would you like to use the secure operating way? ::
Press [S] to continue in a SECURE way
Press [N] to continue in a NORMAL way
Press [K] to EXIT
Press [r] to restart
Press [R] to rebuild and restart
```

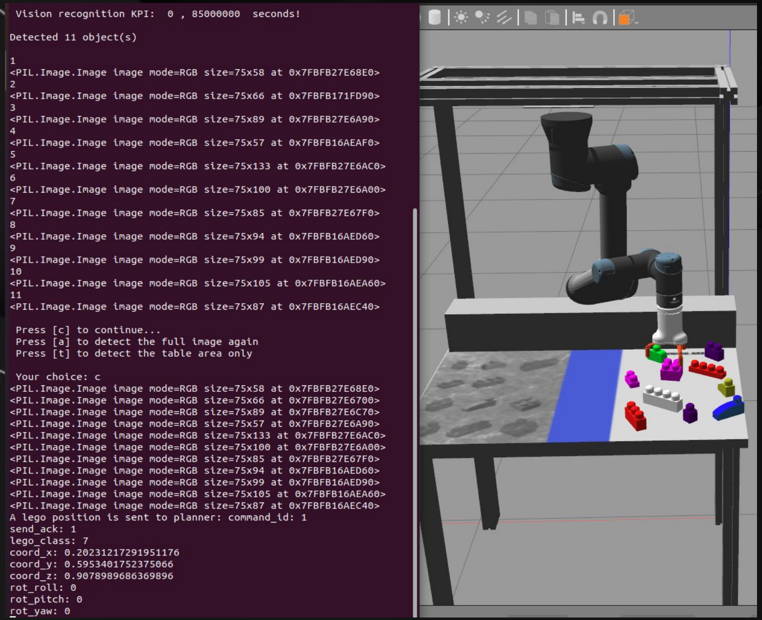
Fundamentals of Robotics project

Make the robot move

Vision module will wait until it receives a command:

- **c** - continue and sends coordinates to the planner
- **a** - Detect again the objects normally
- **t** - Detect the objects only referring to the table area

Note: the **c** command is necessary to go on



Results

We measured the KPI for each modules of this project. The performances vary from a computer to another. In one test we had:

- In all the terminal referring to each module is possible to see the **KPIs**
  - less than 1 second to detect legos by vision module
  - about 32 seconds to execute the 1° assignment
  - about 363 seconds to execute the 2° assignment