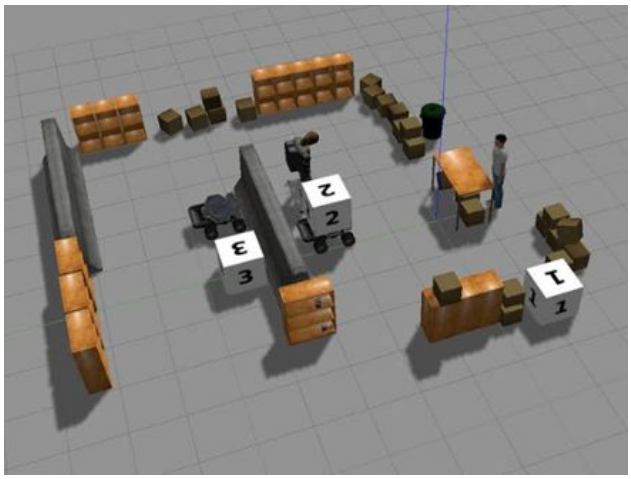
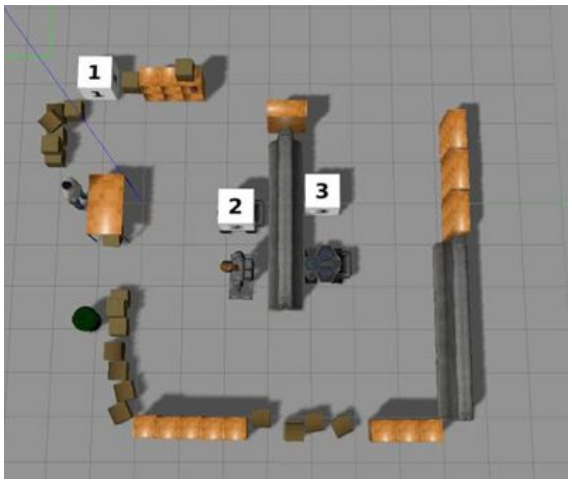
Day 1: Enterprise/ROS Familiarization and Robust Execution

Description:
Through the summer school you will be using the Enterprise/ROS architecture, combined with various planners, to control robots. Enterprise manages decision making, while ROS manages mobility. The goal of this lab is twofold. The first goal is to start familiarizing you with the Enterprise/ROS architecture (henceforth Enterprise) and its different components. The second goal is to give you a first experience with the Enterprise system, applied to a simple, single-robot scenario. In the *first section* of the lab, we will walk you through a step-by-step setup of the code base on your laptops. In the *second section,* you will be writing an RMPL script, to program a robot to finish a set of tasks in a simulated environment.

Environment setup:
Refer to instructions for environment setup in the ~/Documents/labs/lab1 folder.

Scenario
In the following simulation environment, there are three numbered blocks, 1,2 and 3. The mad scientist (you) wants to collect data from the blocks to conduct his experiment, so he wrote a script and sent off his robot to do it for him.





Requirement
Write an RMPL script that achieves the following tasks:

1. Navigate the robots to the numbered blocks, 1, 2 and 3. (order does not matter)
2. Take a photo of each block.
3. Go back to the scientist when the above tasks are done.

The possible actions are:

Action	Arguments	Description
move-2d	v (summit), to-x (real), to-y (real)	Move the summit v to the location specified by (to-x, to-y).
rotate	v (summit), angle (real)	Rotate the summit by angle degrees in counter-clockwise direction.
take photo	v (summit)	Take a photo.

The following is an example for the RMPL script you will need to create.

```
(defmethod main ((world world))
  ;; write a script to run the actions sequentially
  ;; let binds summit-1 to the summit object in the world
  (let ((summit-1 (summit world)))
     ;; move the summit to (-5.0, 1.0)
     (move-2d  summit-1  -5.0  1.0)
     ……..)))
```

The position of the scientist and the numbered blocks are given below:

object	x	y
block-1	-3.0	-1.1
block-1	0.37	2.3
block-1	-0.11	4.26
scientist	0.0	-1.5
home-location	-5.0	0.0

Instructions

Make sure your VM is installed correctly and "update-vm" is run successfully. Then, in the ~/Documents/labs/lab1 folder, we have provided you with the following files:

- lab1-defs.rmpl: Contains the definitions for objects and functions. You should **not** need to modify this file (unless you want to).
- lab1.rmpl: Contains the RMPL script to be executed. You need to modify this file.

Complete the lab1.rmpl file.

Once you have completed the rmpl file, compile the RMPL file into TPN (temporal plan network) file, which is an executable plan format for the robot, using the following command:

```
rmpl2tpn -o example.tpn -i lab1-defs.rmpl lab1.rmpl
```

The command will output the example.tpn in the same folder.

Launch the simulated environment by running the following command:

```
roslaunch cogrob day-1.launch
```

Then open another terminal and execute the TPN plan in simulation by running the following command:

```
rosrun mpex_tools execute_plan example.tpn
```

When the simulation is done, terminate the above two processes by Ctrl-C. Relaunch them if you want to run the simulation again.