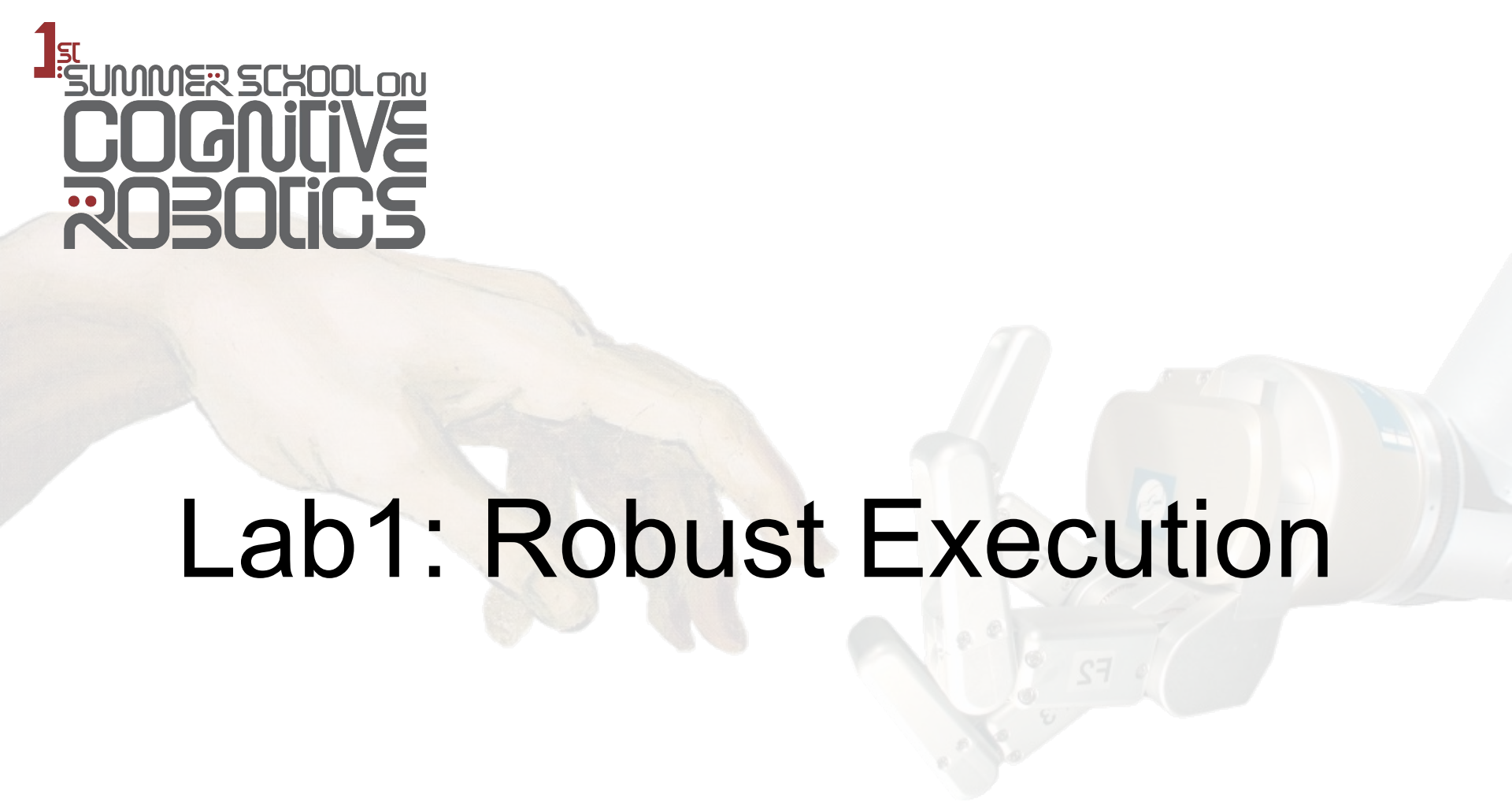
1st Summer School on Cognitive Robotics

Lab1: Robust Execution



NSF
Artificial Intelligence
www.elsevier.com/locate/artint
BOEING
MIT AEROASTRO
CSAIL
MERS Model-based Embedded & Robotic Systems group CSAIL@MIT
MIT

Tutorials and Labs

1st SUMMER SCHOOL on COGNITIVE ROBOTICS
MERS Model-based Embedded & Robotic Systems group CSAIL@MIT

Day 1: Robust Execution
Introduction 1: Architectures for Autonomy
Tutorial 2: Self-Monitoring, Self-Diagnosing Systems
Tutorial 3: Temporal Networks for Dynamic Scheduling

Lab: Enterprise/ROS Familiarization and Robust Execution

Day 2: Motion Planning
Tutorial 4: Sampling-based Motion Planning
Tutorial 5: Single-Robot and Multi-Robot Path Planning with Quality Guarantees
Tutorial 6: Trajectory Optimization for Underactuated Robots

Lab: Incorporating Trajectory Planning for Autonomous Vehicles

Day 3: Activity Planning
Tutorial 7: Classical Planning@Robotics: Methods and Challenges
Tutorial 8: Planning in Hybrid Domains
Tutorial 9: Planning of Concurrent Timelines

Lab: Incorporating Activity Planning

Day 4: Perception and Manipulation
Tutorial 10: Multi-vehicle Routing with Time Windows
Tutorial 11: Generative Models for Perception
Tutorial 12: Fundamentals of Robotic Manipulation and Grasping

Lab: Manipulation and Multi-vehicle Routing

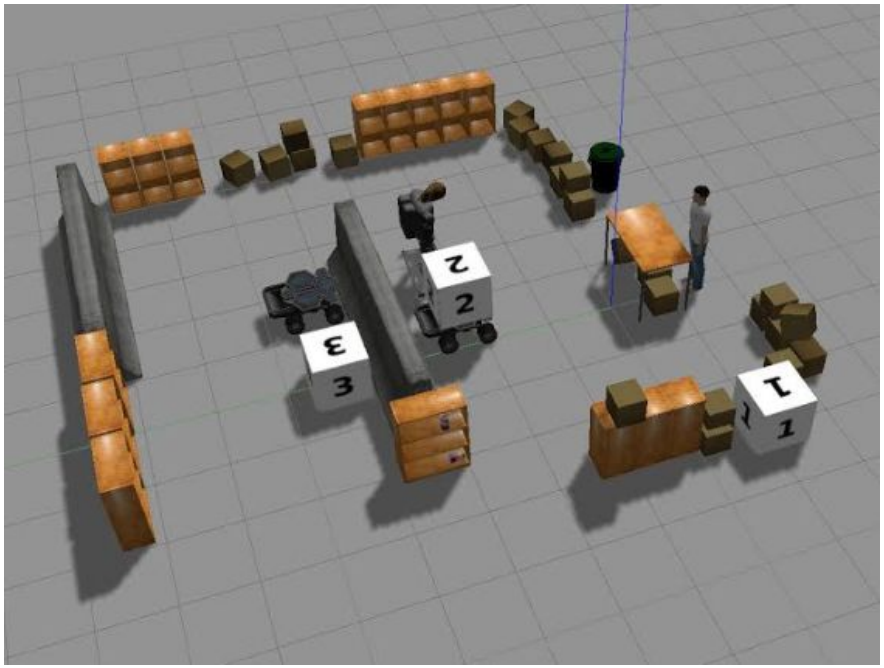
Day 5: Planning with Uncertainty and Risk
Tutorial 13: Probabilistic Planning
Tutorial 14: Localization and Mapping
Tutorial 15: Risk-bounded Planning and Scheduling

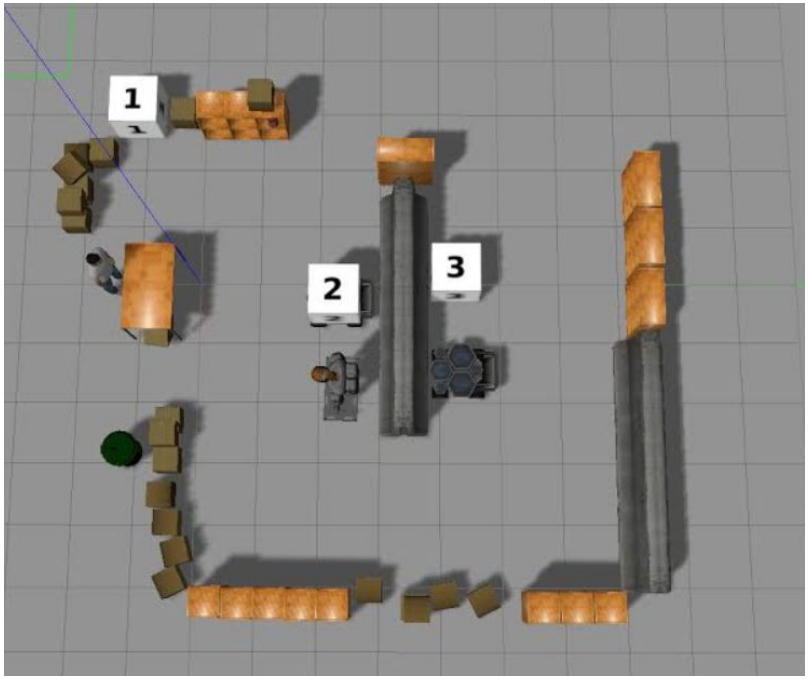
Lab: Challenge. And all comes together ….

Objective

- Start familiarize with using the Enterprise/ROS architecture
- Start familiarize with RMPL
- Use RMPL as a scripting language to program the robot to execute a simple scenario in simulated environment
- Understand the motivation for robust execution

Scenario





Scenario

- In the simulation, 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…
- Goals:
- 1. Move the robots to go to each numbered block (order does not matter)
- 2. Take a picture of each block
- 3. Go back to the scientist when tasks are done

Program Language

- RMPL (Reactive model-based programming language)
- Classes of objects
- Methods
- Main program

Classes

- Specify type and properties for objects
- Summit
- Constructing the world with objects
- 1 summit object

```
(defclass summit ()
        ())
```

```
(defclass world ()
       ((summit
         :type summit
         :final t
         :initform (make-instance 'summit))))
```

Methods

- Actions to be taken
- move-2d
- rotate
- take-photo

```
(defgeneric move-2d (v to-x to-y))

(defmethod move ((v summit) (to-x real) (to-y real))
        (declare (primitive t)))
```

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.

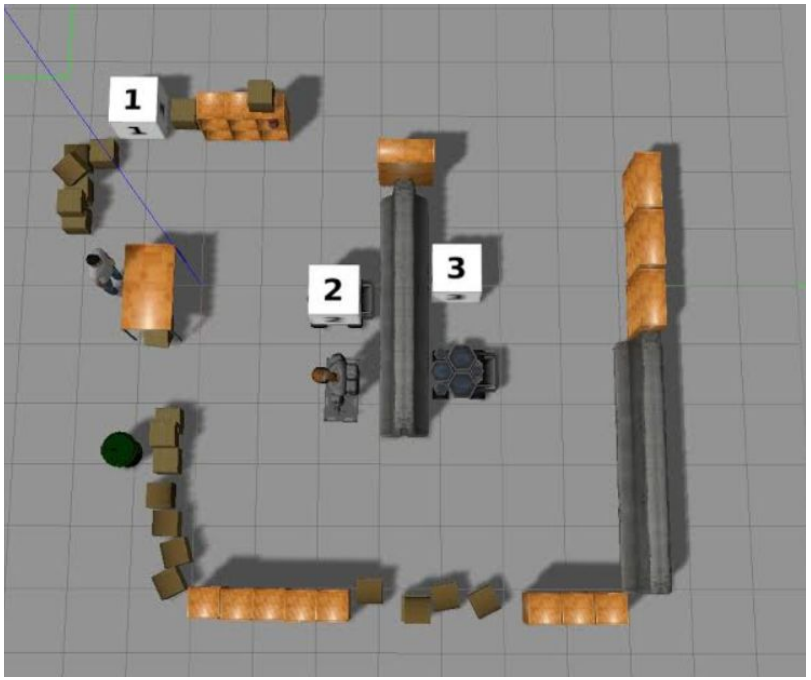
Main program

- Control program to be run

```
(defgeneric main (world))

(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)
            ……..)))
```

World Information



Location:

- Block-1 (x: -3.0 , y: -1.1)
- Block-2 (x: 0.37 , y: 2.3)
- Block-3 (x: -0.11 , y: 4.26)
- Scientist (x: 0.0 , y: -1.5)
- Home-location (x: -5.0, y:0.0)

Lab location