

Intelligent Robotics

Project and simulator

Thibaut Cuvelier

16 February 2017

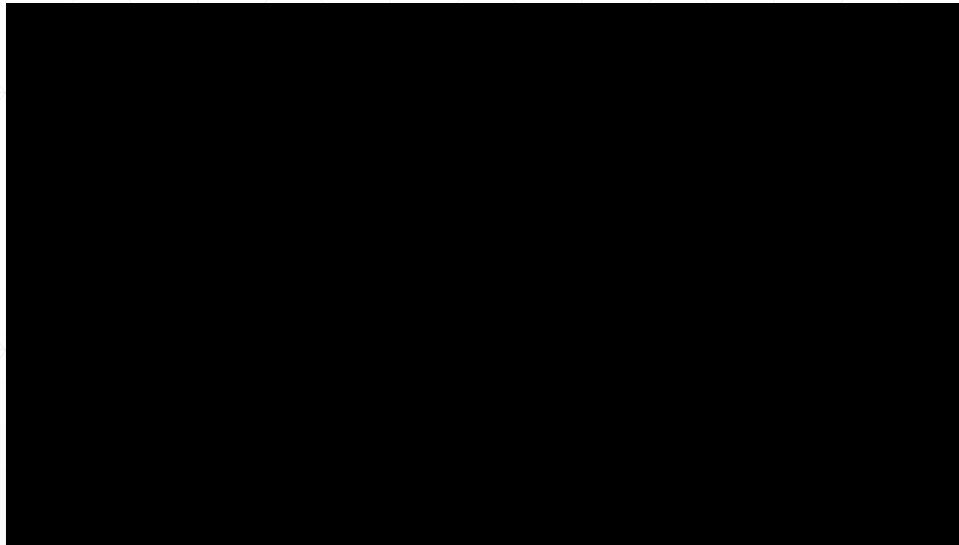
Today's plan

- Project details
- Introduction to the simulator
- MATLAB for the simulator

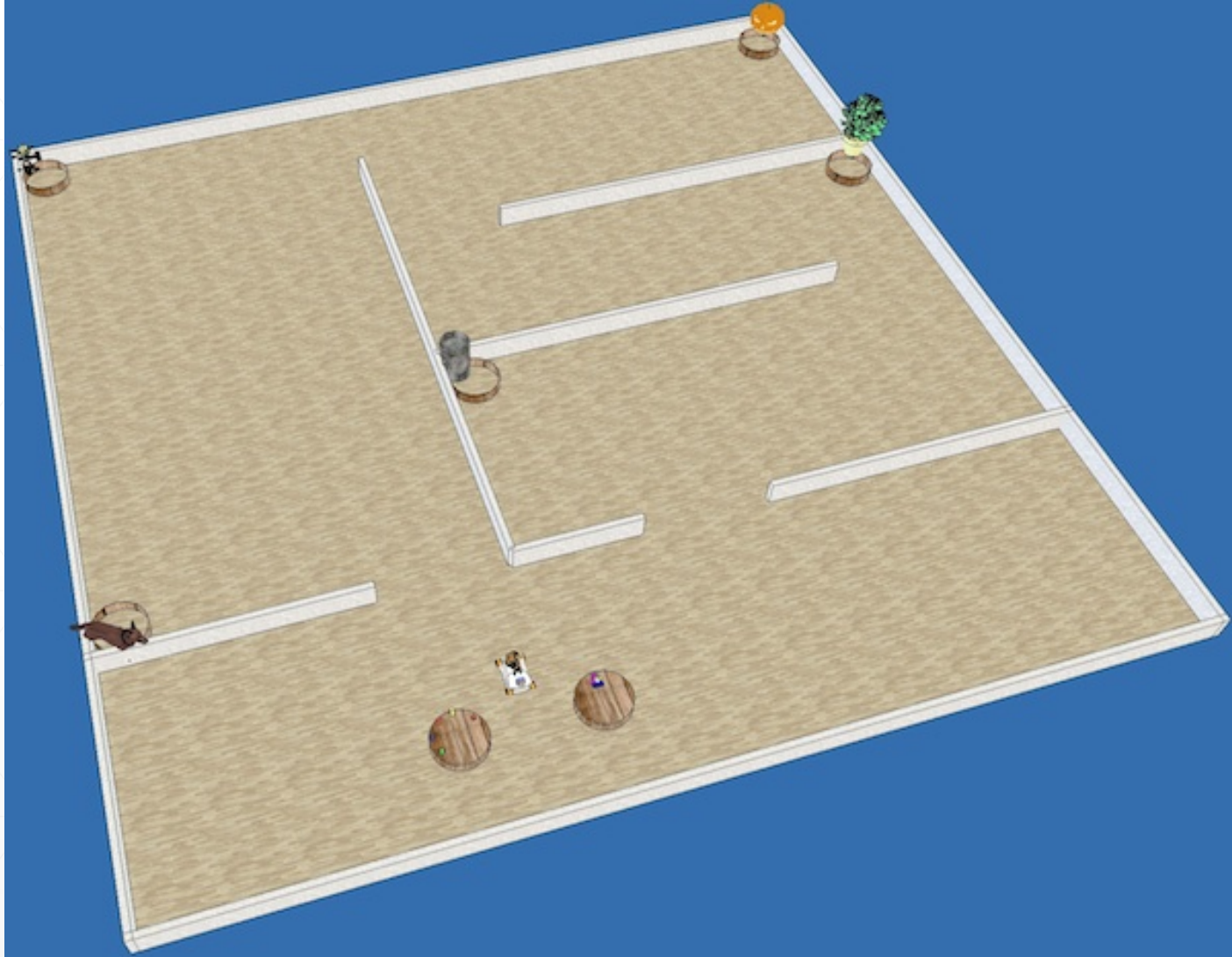
- <http://www.montefiore.ulg.ac.be/~tcuvelier/ir>

About the project

youBot



Your goal: deal with the groceries



Milestones

- The project is divided in a series of milestones
 - No need to do all of them!
 - Make choices based on what you prefer
- Broadly:
 - (A) Navigation
 - (B) Object manipulation
 - (C) Vision
 - (D) Manipulation with vision
 - (E) Calibration

Look on
website for
deadline.

First deadline: ~~March 23~~

- Milestone A1: your robot moves and builds a map
 - You should prepare a small demo of your exploration
 - Roughly five minutes per group
- What we expect
 - “Basic” exploration: no need for something complicated
 - No time constraint: no need to complete exploration in 5 minutes
 - You can do more if you wish
- Schedule conflicts?
 - Contact us

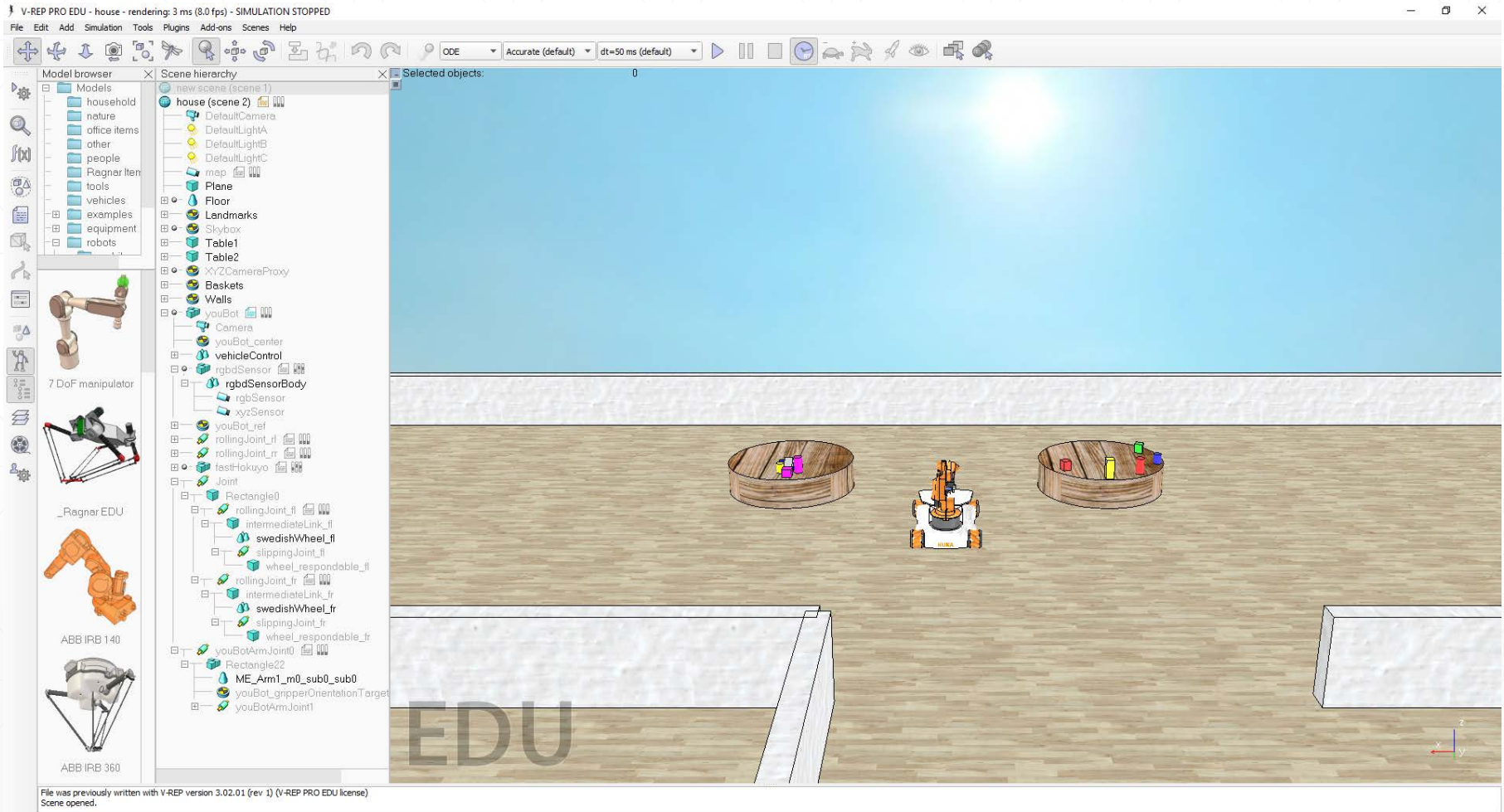
Look on
website for
deadline.

Final deadline: ~~end of May, beginning of June~~

- Presentation of your whole project
 - A small report will be required
 - Oral presentation of your project
 - Think about videos if your code decides not to work
 - You may be asked to test your robot on **another** map
- More details later

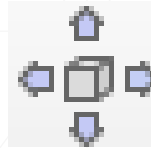
Introduction to the simulator

Basic overview

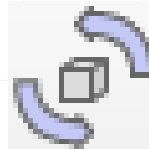


Common manipulations

- Move an object:



- Rotate an object:



- Scaling an object:



- **Caution:** any modification while the simulator is running is lost at the end of the simulation!

MATLAB for the simulator

Many functions available in the simulator

- You will be writing MATLAB code to interact with the robot
 - Set the speed of the wheels
 - Take a picture
 - Move the gripper
 - Get the position of the robot or one of its components
- See examples:
 - Complete example
<https://github.com/dourouc05/trs/blob/master/youbot/youbot.m>
 - More focused and much shorter examples:
<https://github.com/dourouc05/trs/tree/master/youbot/focused>

Many functions available, but not infinitely many

- Not all functions can be called:
<http://ulgrobotics.github.io/trs/project.html#api>
 - For example: forbidden to move an object into the gripper
 - But you can use forbidden functions for your tests, of course
- A few functions are not allowed for given milestones
Sometimes, not all arguments are allowed
 - For example, B4: you cannot use VREP IK to move the arm
 - Very natural restrictions

Many functions available outside the simulator

- When running the installer, Peter Corke's robotics and vision toolbox is automatically installed
 - Many useful functions for the project
 - Reference frame transformations, navigation, trajectories...
 - **Pay attention:** not always working as you would expect!
- If you have it: MATLAB Robotics Toolbox

Programming tips

- **Use an infinite loop**
 - Simulation goes on continuously for each iteration in this loop
 - Take actions at each iteration:
 - Set the speeds for the wheels
 - Plan your path through the room
 - Take a picture
 - Analyse a picture
 - ...

```
while true
  % ...
end
```


Programming tips

- **Use a state machine, such as:**
 - State 1:** Explore the map
 - State 2:** Go to the tables
 - State 3:** Pick an object
 - State 4:** Move to the corresponding basket
 - State 5:** Drop the object

Back to state 2 until all objects are dealt with
- You can of course decompose further, embed state machines within some states, etc.

Debugging tips

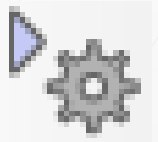
- When you work on an image or a point cloud
 - First run a simulation and save the image/cloud to a file
 - Then develop your algorithm
 - Finally try the integrated code
- Don't tune a parameter, run the simulator, tune, run, etc. (Great loss of time!)
- The samples show how to store an image and a point cloud
- You can also use MATLAB's save and load functions

How does the simulator work?

- The simulator uses a physics engine
 - Must be allowed to run often enough for realistic simulations
- Two impacts:
 - The simulator physics engine has an iteration every **50 ms**
 - Risk of overshooting
 - Don't approach waypoints at a too high speed
 - Don't rotate too fast

How does the simulator work?

- The simulator uses a physics engine
 - Must be allowed to run often enough for realistic simulations
- Two impacts:
 - Your code **should run within 50 ms**
 - Otherwise: physics desynchronised from your measurements
 - Use already optimised functions! (Or optimise your code)
 - You can also precompute a few things
 - If not enough: multiplication factor, non-real-time mode



How does the simulator work?

- The simulator uses a physics engine
 - Must be allowed to run often enough for realistic simulations
 - Two impacts:
 - Sometimes, **strange robot behaviour**
 - Gripper closed, object falling
 - Robot wheels straight, but robot following a bended curve
 - Mostly due to numerical errors in the simulation
- Dynamic steering!

Questions?

Installation

- Supposing MATLAB is installed
- Install V-REP PRO EDU:
<http://www.coppeliarobotics.com/downloads.html>
- Install V-REP bindings for MATLAB: step 3 of
<http://ulgrobotics.github.io/trs/setup.html#install>
- Clone or download the course's Git repository:
<https://github.com/dourouc05/trs>
- Run the script `startup_robot.m`
 - Installs Peter Corke's toolbox
 - Sets MATLAB's path
 - Must be run each time you restart MATLAB!