Robotics

Project 1

POLITECNICO
MILANO 1863

Paolo Cudrano
paolo.cudrano@polimi.it

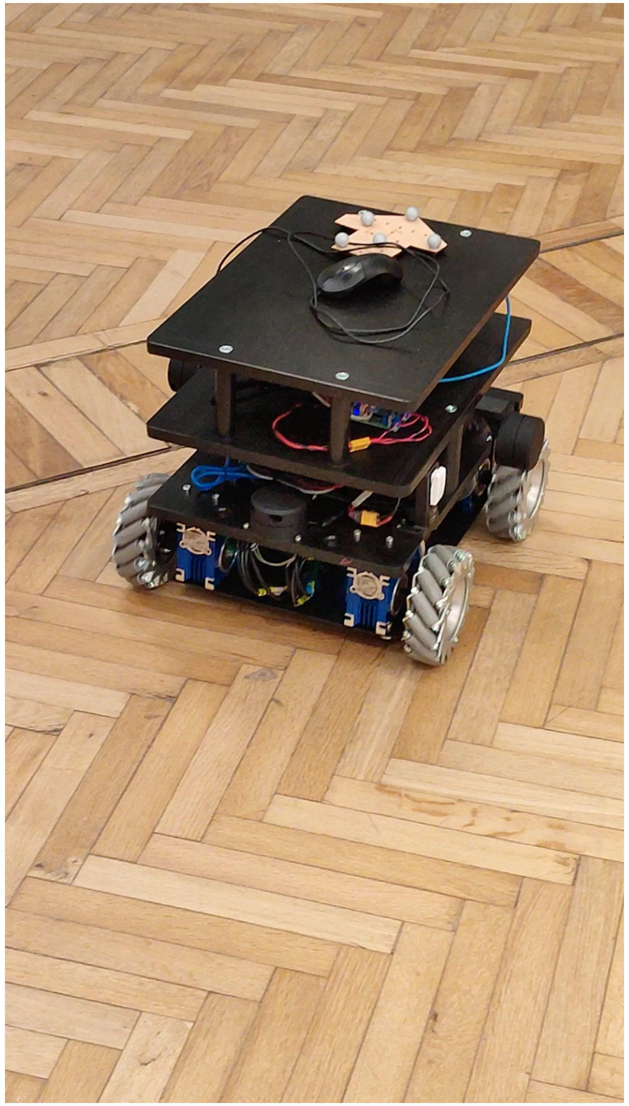
PROJECT 1

ROBOTICS

POLITECNICO
MILANO 1863

THE ROBOT

THE ROBOT



Disclaimer: this video is only for demonstrational purposes. It does NOT show the actual path followed by the robot during our data acquisition.

THE ROBOT

Omnidirectional robot

Mecanum wheels

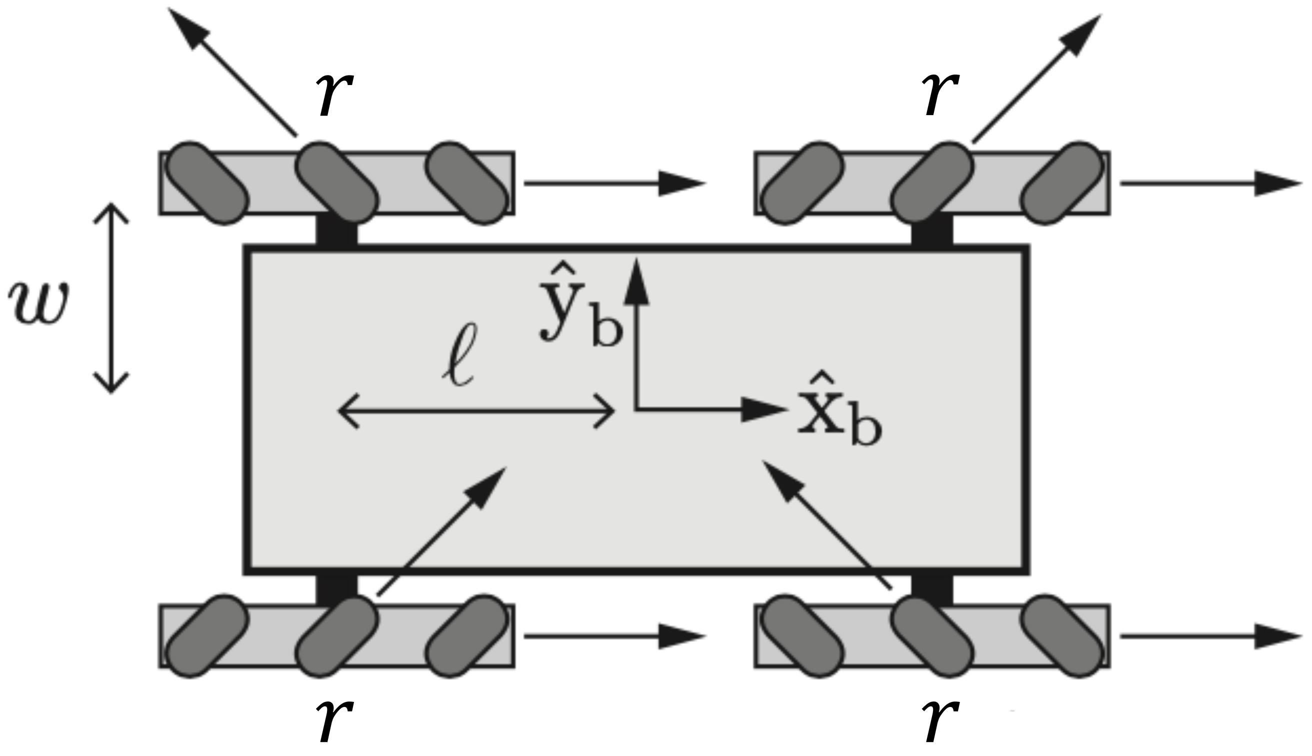
- 4 wheels with rollers at 45°

Encoders on each wheel

- RPM (very noisy)
- Ticks (more accurate)

Geometric parameters:

- Wheel radius (r)
- Wheel position along x ($\pm l$)
- Wheel position along y ($\pm w$)



THE PROJECT

Given

- Wheels encoder state
  - RPM (noisy)
  - Ticks (more accurate)
- Nominal robot parameters r, l, w
- Ground truth pose of the robot (acquired with OptiTrack)

THE PROJECT

Goals

I. Compute odometry using appropriate kinematics
   - Compute robot linear and angular velocities v, ω from wheel encoders
   - Compute odometry using both Euler and Runge-Kutta integration
     - ROS parameter for initial pose
   - Calibrate (fine-tune) robot parameters to match ground truth

II. Compute wheel control speeds from v, ω

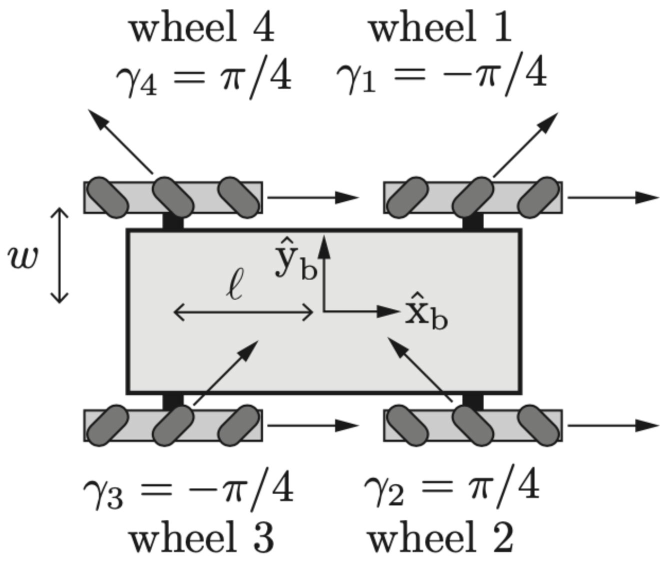
III. Add a service to reset the odometry to a specified pose (x,y,θ)

IV. Use dynamic reconfigure to select between integration method

I. COMPUTE ODOMETRY

- Compute velocities with mecanum wheels kinematics
  1. Write down the formula to compute v, ω from wheel speeds *
  2. Adapt formula to use wheel ticks instead of RPM (more precise)
  3. Compute a rough estimate of v, ω (with given robot parameters fixed)
  4. Publish v, ω as topic `cmd_vel` of type `geometry_msgs/TwistStamped`

* Do some reading on mecanum wheels.
In particular, you can use
"Modern robotics: Mechanics, Planning, and Control" by Kevin M. Lynch and Frank C. Park
Free version here



I. COMPUTE ODOMETRY

- Compute velocities with mecanum wheels kinematics
  1. Write down the formula to compute v, ω from wheel speeds
  2. Adapt formula to use wheel ticks instead of RPM (more precise)
  3. Compute a rough estimate of v, ω (with given robot parameters fixed)
  4. Publish v, ω as topic `cmd_vel` of type `geometry_msgs/TwistStamped`
- Compute odometry using v, ω
  1. Start with Euler, add Runge-Kutta later
  2. Add ROS parameter for initial pose (x,y,θ)
  3. Publish as `nav_msgs/Odometry` on topic odom
  4. Broadcast TF `odom->base_link`
- Calibrate robot parameters (r, l, w) to match ground truth pose

II. COMPUTE CONTROL

- Compute wheel speeds (RPM) from v, ω
  1. Reverse the formula obtained at the previous step (I.1)
  2. Read v, ω from `cmd_vel` and apply the obtained formula
  3. Publish the computed wheel speed as custom message on topic `wheels_rpm`
     - The custom message has prototype:

```
Header header
float64 rpm_fl
float64 rpm_fr
float64 rpm_rr
float64 rpm_rl
```

f/r: front/rear
l/r: left/right

Check that the results match the recorded encoders values, apart from some noise

- You could use `rqt_plot` or `plotjuggler`

III. RESET SERVICE

- Define a service to reset the odometry to any given pose (x,y,θ)

IV. INTEGRATION METHOD SELECTOR

- Use dynamic reconfigure to select the odometry integration method
- Use an enum with 2 values: Euler, RK

DATA

3 ROS bags, with topics:

- Encoder message containing RPM and ticks for the 4 wheels
  - Topic `/wheel_states` of type `sensor_msgs/JointState`
- Ground truth pose of the robot (acquired with OptiTrack)
  - Topic `/robot/pose` of type `geometry_msgs/PoseStamped`

DATA

Additional information:

- Wheel radius (r): 0.07 m (could be a bit off)
- Wheel position along *x* (l) : 0.200 m
- Wheel position along *y* (w) : 0.169 m
- Gear ratio (T): 5:1

FILES

Everything you need (bags, these slides)

is in our **shared folder** (link on the course website)

TIPS FOR DEBUG

You can use `rviz` to visualize the given ground truth pose and your odom and TF.

In this case, you might want to define a static TF transform to align the frame `odom` (yours) and `world` (used by OptiTrack).

For the robot calibration, notice that:

- bag1 only performs linear motions (no rotations)
- bag 2 only performs forward motions and rotations (no y translation)
- (bag 3 is freestyle ☺)

ADDITIONAL INFO

The ground truth pose is measured with an Optitrack system, which is based on cameras. For this reason, this information might sporadically not be available due to occlusion, or it might present some lag. This should not affect your project, just be aware of it.

GENERAL REQUIREMENTS

The project must be written in C++

- NO Python unless previously discussed

You can use any number of nodes.

You must provide **1 single launch file** to start **everything** needed except the bag (i.e., all nodes, parameters, etc.).

Remember that we should be able to run your code on our machines

- E.g., DO NOT use absolute paths
- Test with Ubuntu 18.04 + ROS Melodic
- (If possible, test on a colleague's machine)

SUBMISSION

You should deliver your project via email, attaching 1 compressed file as follows:

- Create a .tar.xz archive containing:
  - a text file with instructions (details in next slide)
  - the source code for every package you created (entire package folders, including their `CMakeLists`, `package.xml`, etc.)
    - Please DO NOT send us your entire catkin environment (with `build` and `devel` folders); we only need the packages you created inside of your `src` folder
  - Notice: include all the files you think are important, as we should be able to properly recreate your workflow with what you send us
- Name the archive with the Student ID ("Person Code") of each team member
  - E.g. 10450001-10450002-10450003.tar.xz
- Send this archive (only **once per team**) via e-mail to **all of us**: Paolo Cudrano, Simone Mentasti and Matteo Matteucci. Email subject must be “FIRST ROBOTICS PROJECT 2022”

SUBMISSION

The instruction file must be a .txt or .md file and **must** contain (at least):

- Student ID ("Person Code"), name and surname of all team members
- small description of the files inside the archive
- name and meaning of the ROS parameters
- structure of the TF tree
- structure of any custom message
- description of how to start/use the nodes
- info you think are important/interesting

TEAMS AND DEADLINE

You can work in teams of **max 3 students** (unless previously discussed)

Deadline: April 29 2022, 23:59 CEST (Rome time zone)

QUESTIONS

Questions:

- Send an email to Paolo Cudrano (paolo.cudrano@polimi.it)
  - Cc: Simone Mentasti (simone.mentasti@polimi.it)
  - DO NOT write only to Prof. Matteucci

CHALLENGE

ROBOTICS

POLITECNICO
MILANO 1863

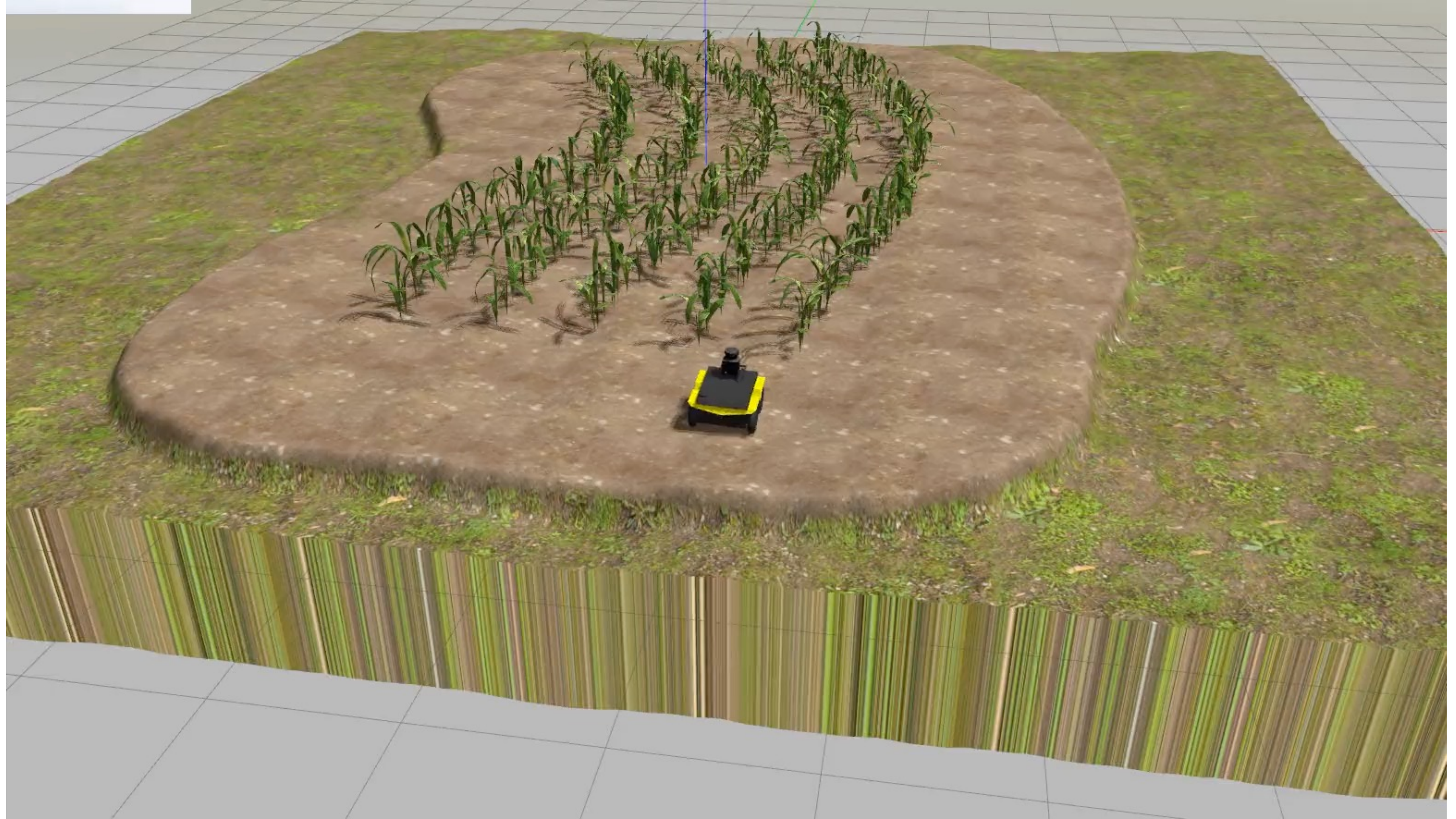
FIELD ROBOT EVENT 2022

We would like to participate with a team of students to the International Field Robot Event 2022, June 14–16, 2022

https://www.fieldrobot.com/event/

Navigation and mapping in a maize field

- Virtual robot
- Real robot ? (TBD)



HOW TO PARTICIPATE

Interested students should meet with us in order to discuss further details.

You can let us know your interest by sending an email to **all of us** with subject "Robotics - FRE 2022".

We should receive these requests by **April 8**
(Not a strict deadline, but we need some time to get organized...)

Important: to avoid inconveniences in the future, we strongly suggest interested students to still prepare and submit Project 1.