

# EMC 2014 Practical Assignment A-maze-ing PICO

Sjoerd van den Dries  
Eindhoven University of Technology  
Department of Mechanical Engineering



**TU** / **e**

Technische Universiteit  
**Eindhoven**  
University of Technology

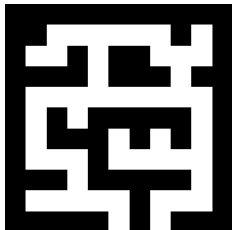
April 25, 2014

Where innovation starts

- ▶ Telepresence Robot from Aldebaran
  - Robot type: **Jazz**
- ▶ Sensors:
  - **Laser Range Finder (LRF)**
  - **170° wide-angle camera**
  - **Sonar**
- ▶ Actuators:
  - **Holonomic base (omni-wheels)**
  - **Pan-tilt unit for head**
- ▶ Computer:
  - **Intel I7**
  - Running **Ubuntu 12.04**
  - **ROS (Robot Operating System)**



- ▶ Let PICO navigate through a maze and find and go to the exit.
- ▶ You have to:
  - try to be as fast as possible
  - but avoid hitting obstacles at all cost!
- ▶ You can use:
  - The Laser Range Finder to detect walls
  - (Optionally: use camera to detect arrows pointing to the exit)
- ▶ Competition day: June 20th



- ▶ **Corridor Competition:** Let PICO drive through a corridor and go through the side exit.
- ▶ You have to:
  - try to be **as fast as possible**
  - but **avoid hitting obstacles** at all cost!
- ▶ You can use:
  - The **Laser Range Finder** to detect walls
- ▶ Competition day: **May 16th**



- ▶ **Linux**-based operating system
- ▶ Use version **12.04** (not 14.04!)
- ▶ 32- and 64-bit (**64-bit recommended**)
- ▶ Easy dual boot installation with e.g., Windows
- ▶ Download: [www.ubuntu.com](http://www.ubuntu.com)
  - Any problems? → [Check the wiki](#).
  - No info? → Ask the ICT Servicedesk or contact us.



- ▶ **Robot Operating System**
- ▶ Open-source meta-operating system for robots
- ▶ Primary goal: support code reuse in robotics R&D
- ▶ Implemented in C++, Python
- ▶ Allows running code on multiple computers
- ▶ We will use ROS **Groovy**
- ▶ [www.ros.org](http://www.ros.org)

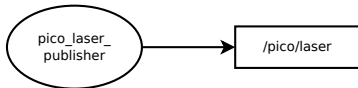


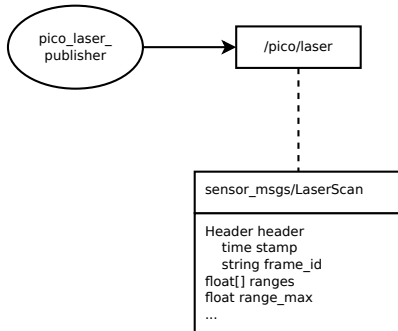
- ▶ **Node**: process that performs computation
- ▶ **Master**: provides name registration and lookup
- ▶ **Messages**: nodes communicate with each other by passing messages
- ▶ **Topics**: named buses over which nodes exchange messages
- ▶ **Services**: request/reply communication
- ▶ **Parameter Server**: allows data storage by key in a central location

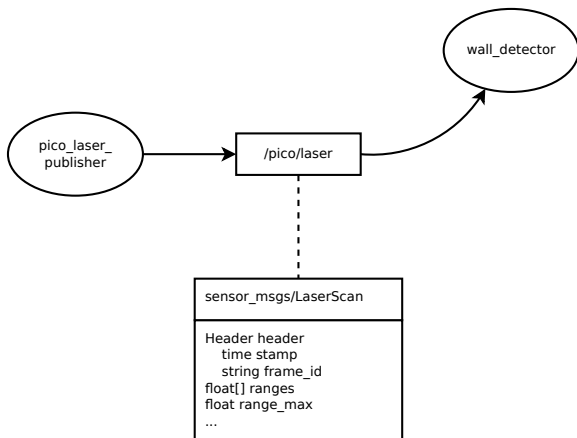


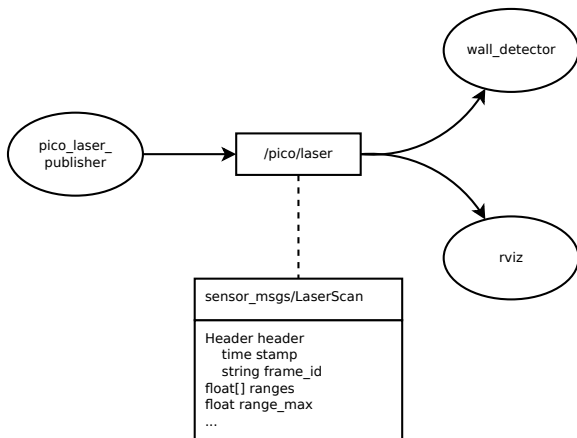
pico\_laser\_  
publisher











- ▶ **Package:** contains one chunk of functionality with clear input and output
  - A module, library, set of tools, ...
  - *E.g.*, `wall_detector`
- ▶ **Manifest:** contains meta data about a package
  - Dependencies
  - Author
  - ...

- ▶ **Package:** contains one chunk of functionality with clear input and output
  - A module, library, set of tools, ...
  - *E.g.*, `wall_detector`
- ▶ **Manifest:** contains meta data about a package
  - Dependencies
  - Author
  - ...
- ▶ **Stack:** collection of packages that provides functionality as a whole
  - *E.g.*, `navigation`
- ▶ **Stack manifest:** contains meta data about the stack

- ▶ Enormous amount of **open-source libraries** and tools available!
  - Approx. **2000 packages** and counting!
  - Perception
  - Navigation
  - Manipulation
  - Visualization
  - Etc ... Etc ...

- ▶ Enormous amount of **open-source libraries** and tools available!
  - Approx. **2000 packages** and counting!
  - Perception
  - Navigation
  - Manipulation
  - Visualization
  - Etc ... Etc ...
- ▶ Thriving community
  - Approx. **100 research groups contributing** (registered!)
  - Mailing lists, forums, ...
  - Spin-offs:
    - **ROS industrial**



- ▶ We will use C++ as programming language
- ▶ One of the two core ROS languages
  - Packages `roscpp` and `roslib`
- ▶ C++ is object-oriented C
  - “C with Classes”
  - Encapsulate data and functionality within objects
- ▶ More on C++ and ROS [next week](#)

- ▶ **Integrated Development Environment**
  - Advanced code editor
- ▶ **Many advantages over 'simple editors':**
  - Syntax highlighting
  - Code completion
  - Visual compiler feedback
  - Static code checking
  - Refactoring tools
  - Parenthesis matching
  - ...



- ▶ Version Control System:
  - *'Manages files and directories, and the changes made to them, over time'*
- ▶ Use to store and maintain your code on the server

- ▶ **Version Control System:**
  - *'Manages files and directories, and the changes made to them, over time'*
- ▶ Use to store and maintain your code on the server
- ▶ **Basic commands:**
  - `svn checkout <URL>`
  - `svn update`
  - `svn add <FILENAME / DIRECTORY>`
  - `svn commit -m '...message...'`
  - `svn status`

- ▶ We created a dedicated SVN repository for EMC
- ▶ SVN-account **per group**

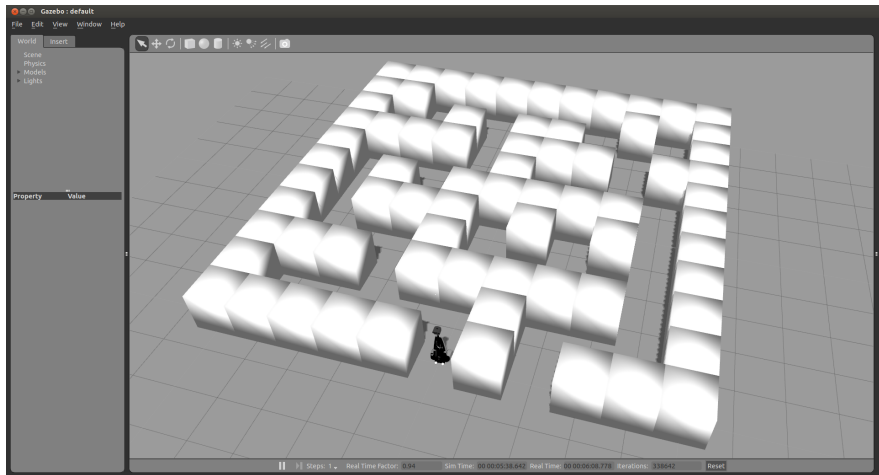
- ▶ We created a dedicated SVN repository for EMC
- ▶ SVN-account *per group*
- ▶ <https://roboticssrv.wtb.tue.nl/svn/emc/2014>
  - general *(accessible to all)*
    - pico\_gazebo
    - pico\_example
    - ...
  - groups *(accessible per group)*
    - emc01
    - emc02
    - ...

- ▶ You will have to work with the **real robot**, but we **only have one**. Therefore:

- ▶ You will have to work with the **real robot**, but we **only have one**. Therefore:
- ▶ **PICO Simulation:**
  - Build in **Gazebo simulator**
  - Simulates:
    - Sensors
    - Actuators
    - Environment (maze)
    - Physics
  - Integrates well with **ROS**







## ▶ RViz

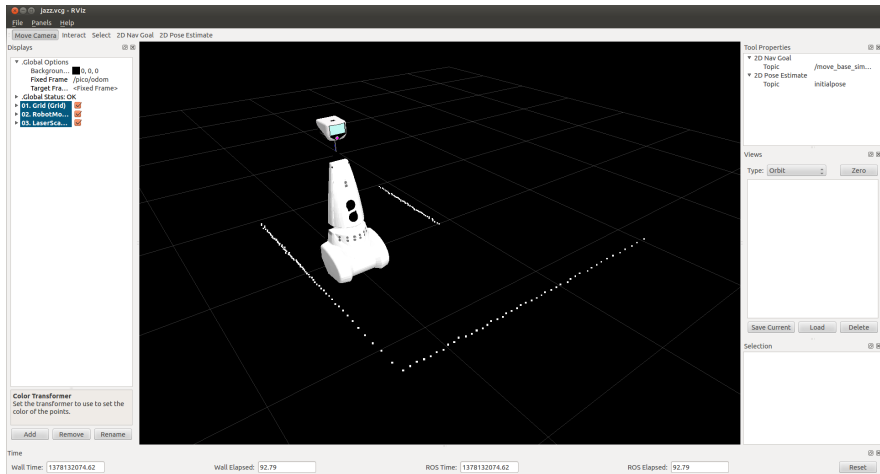
- 3D visualization tool for ROS
- Visualizes:
  - Robot model
  - Sensor data
  - Odometry information
  - Markers
  - ...

## ▶ RViz

- 3D visualization tool for ROS
- Visualizes:
  - Robot model
  - Sensor data
  - Odometry information
  - Markers
  - ...

## ▶ Gazebo + RViz:

- **Gazebo** determines how the world **is**
- **RViz** shows how the robot **perceives** it



▶ EMC Wiki:

- [http://cstwiki.wtb.tue.nl/index.php?title=Embedded\\_Motion\\_Control](http://cstwiki.wtb.tue.nl/index.php?title=Embedded_Motion_Control)
- Info on practical assignment, installation, getting started
- Log-in: student account

- ▶ **EMC Wiki:**
  - [http://cstwiki.wtb.tue.nl/index.php?title=Embedded\\_Motion\\_Control](http://cstwiki.wtb.tue.nl/index.php?title=Embedded_Motion_Control)
  - Info on practical assignment, installation, getting started
  - **Log-in: student account**
  
- ▶ **Group pages on EMC Wiki:**
  - Each group its own page
  - **Update at least weekly**

- ▶ EMC Wiki:
  - [http://cstwiki.wtb.tue.nl/index.php?title=Embedded\\_Motion\\_Control](http://cstwiki.wtb.tue.nl/index.php?title=Embedded_Motion_Control)
  - Info on practical assignment, installation, getting started
  - Log-in: [student account](#)
- ▶ Group pages on EMC Wiki:
  - Each group its own page
  - [Update at least weekly](#)
- ▶ Overall use:
  - Everyone can [edit](#)
  - If you see a mistake: [correct it](#)
  - If you had a problem and know how to fix it: [add it](#)

- ▶ Assignment: solve maze with PICO robot



- ▶ Assignment: [solve maze](#) with [PICO](#) robot
- ▶ OS: [Ubuntu 12.04](#)
- ▶ Platform: [ROS](#)
- ▶ Programming language: [C++](#)
- ▶ Code editor: [Qt Creator](#)
- ▶ Version control: [SVN](#)
- ▶ Simulation: [Gazebo](#)
- ▶ Visualization: [RViz](#)
- ▶ Documentation: [Wiki](#)

1. Create groups
  - 4-5 students per group
2. Check the wiki:
  - [http://cstwiki.wtb.tue.nl/index.php?title=Embedded\\_Motion\\_Control](http://cstwiki.wtb.tue.nl/index.php?title=Embedded_Motion_Control)
3. Add your group info to the wiki
4. **One person:** send an email to `s.v.d.dries@tue.nl`:
  - Containing group number
  - Put other group members in CC
5. Follow the **tutorials** on the wiki:
  - Ubuntu, C++, ROS, SVN, Qt Creator, simulator

SVN-account information and tutor name will be sent to you  
It is *your* responsibility to get in touch with your tutor