UNIVERSITÀ DEGLI STUDI DI PADOVA

Robotics in Computer Science Education

Prof. Emanuele Menegatti
Intelligent Autonomous Systems Lab (IAS-Lab)
Dept. of Information Engineering
School of Engineering
University of Padova, Italy

Intelligent Autonomous Systems Lab
IAS-Lab People

Staff

1. Enrico Pagello, Full Professor of Computer Science
2. **Emanuele Menegatti, Associate Professor**
3. Michele Moro, Assist. Professor

Students

1. Filippo Basso, PhD cand.
2. Mauro Antonello, PhD cand.
3. Riccardo Levorato, PhD cand.
4. Roberto Bortoletto, PhD stud.
5. Elisa Tosello, PhD stud.
6. Nicolò Boscolo, PhD stud.

About 10 Master-level students

Post-Docs

1. Luca Tonin, Post Doc
2. Stefano Ghidoni, post-doc
3. Stefano Michieletto, post-doc
4. Matteo Munaro, post-doc

Collaboratorations

- Antonio D'angelo and Claudio Mirolo, Assist. Prof.s at Udine University
- Stefano Carpin, Assoc. Prof. at Univ. of California (USA)
- Hiroshi Ishiguro, Prof. Univ. of Osaka (JAPAN)
- Tamio Arai, Prof. Univ. of Tokyo (Japan)
- Frank Dellaert, Prof. Georgia Teach Inst. (USA)
- Wolfram Burgard, Prof. Univ. of Freiburg (Germany)
- Stefano Soatto, Prof. UCLA
- Jeff Bruke, Assistant Dean UCLA
- Radu Rusu, CEO of Open Perception

UNIVERSITÀ DEGLI STUDI DI PADOVA

Topics & equipment

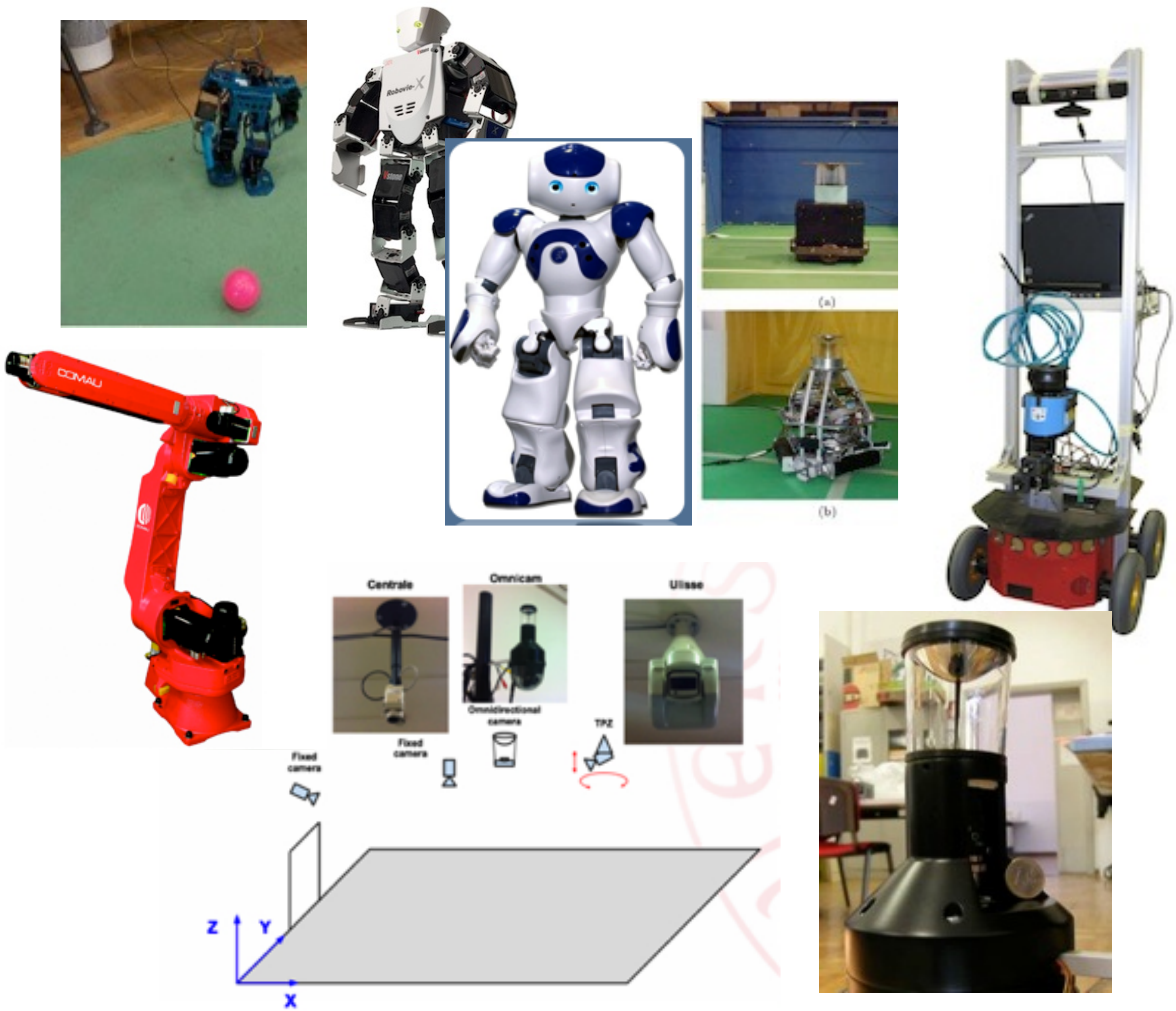
IAS-Lab @ UniPD

Research topics

- Autonomous robotics
- Humanoid robotics
- Camera network applications
- Educational Robotics

Lab equipment

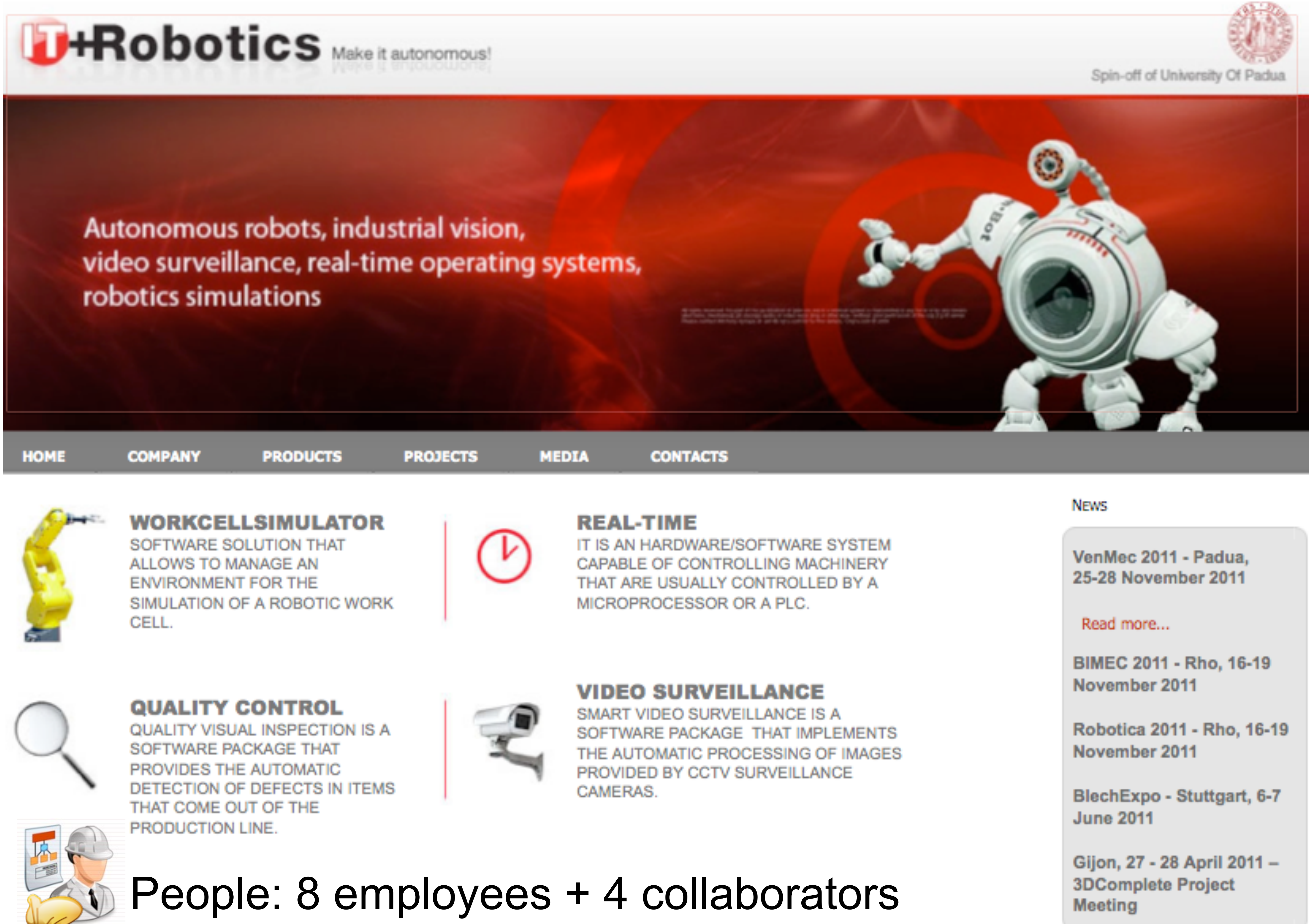
- 3 robot manipulators
- 9 humanoid robots
- 5 mobile robots
- Distributed network of intelligent cameras
- 3D vision cameras



UNIVERSITÀ DEGLI STUDI DI PADOVA

Recent Funded Projects @ IAS-Lab

- EU-FoF 2012
  - FibreMap - Automatic Mapping of Fibre Orientation for Draping of Carbon Fibre Parts
- EU-FoF 2011
  - Thermobot - Autonomous robotic system for thermo-graphic detection of cracks
- EU-RfSME 2010
  - 3DComplete - Efficient 3D Completeness Inspection
- EU-FSE 2009:
  - iSP - Innovative Simulation and Programming of robotics workcells
  - iDVS2 - Intelligent Distributed Audio and Video Surveillance System
- EU-FSE 2008:
  - iDVS: Intelligent Distributed Vision System for surveillance and quality inspection
- EU-Comenius2 2006:
  - TERECoP: Teacher Education on Robotics-Enhanced Constructivist Pedagogical Methods
- University of Padua 2009:
  - Mobility, Perception, and Coordination for a Team of Autonomous Robots

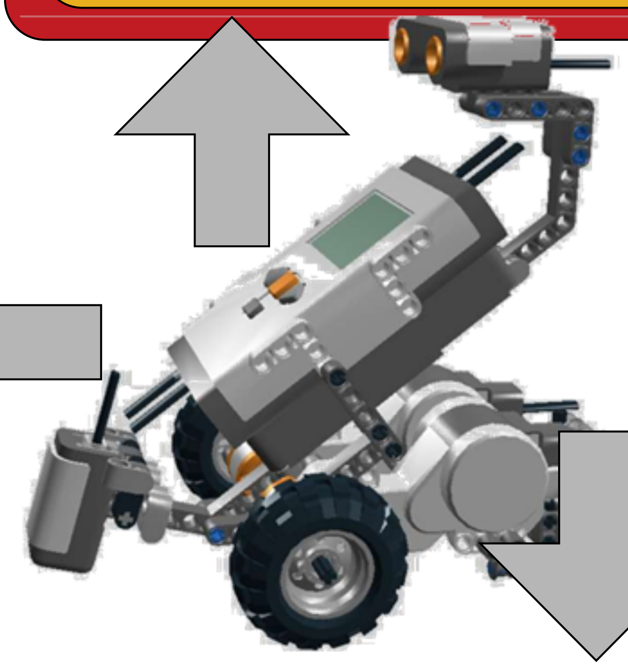


People: 8 employees + 4 collaborators

Money: 600.000 Euro annual turnover

Motivations for Educational Robotics

- Robotics as tool for other disciplines
- Robotics as a subject



- Robotics as extra-curricular activity
- Robotics for inclusion

Germany Case study 1: Roberta Project

Freshman Engineers Build MATLAB Powered LEGO Robots

PROF. TIL AACH, & PROF. ALEXANDER BEHRENS, RWTH AACHEN UNIVERSITY

In a lab "MATLAB® meets LEGO Mindstorms," more than 300 first-year students at RWTH Aachen University build and program their own robots.

Examples of projects:

- robot can translate a brief message into Morse code.
- locate a bottle on a table and determine how full it is.
- robots park themselves autonomously

UNIVERSITÀ DEGLI STUDI DI PADOVA

IAS-Lab expertise with Educational Robotics

UNIVERSITÀ DEGLI STUDI DI PADOVA

TERECOP Project

You are here: TERECoP>Products

Book: Teacher Education on Robotics-Enhanced Constructivist Pedagogical Methods
Editor: Dimitris Alimisis
Published 2009 by School of Pedagogical and Technological Education (ASPETE)
ISBN 978-960-6749-49-0

Robotics in School Education
(Review of research literature and of the existing situation in the 6 participating countries)

Study: A methodology for designing robotics-enhanced constructivist learning for seconda
basic principles, learning objectives and strategies

Study: A methodology for designing robotics-enhanced constructivist learning for seconda
appropriate technology-based environment

Study: A methodology for designing robotics-enhanced constructivist learning for secondary school students: the bus route

Study: Methodology for evaluation and assessment of the learning activities

Pilot training curriculum

Pilot tra

Pilot tra

Pilot tra

TERECoP project
Teacher Education on Robotics-Enhanced Constructivist Pedagogical Methods

In brief:
1.Open Access Book (CreativeCommons lic.)
2. Curriculum for teacher course…
…about introducing robotics in class as a learing tool

"Learning with Robotics"

- Chap. 1
  Educational Robotics and its roots
- Chap. 2
  Introduction to the educational robot
- Chap. 3
  Robotics for STEM
  - 12 lab activities of increasing complexity
- Appendix: NXT-GTD: mark-up language for NXT-G iconic language

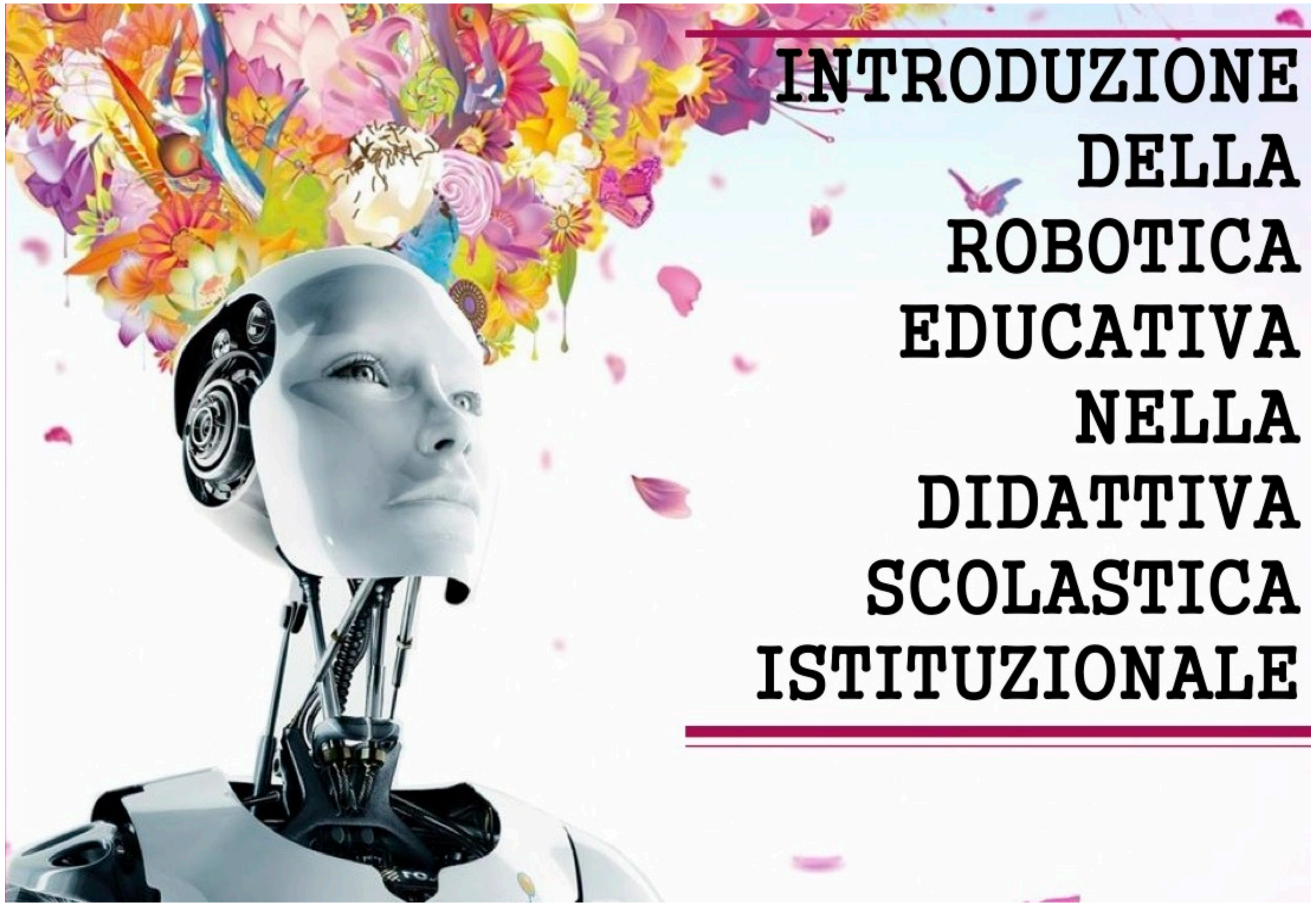
Long-life learning course for school teachers

"Introducing Educational Robotics in standard curricula in schools"

two editions: 2013 & 2014

50+ teachers from:

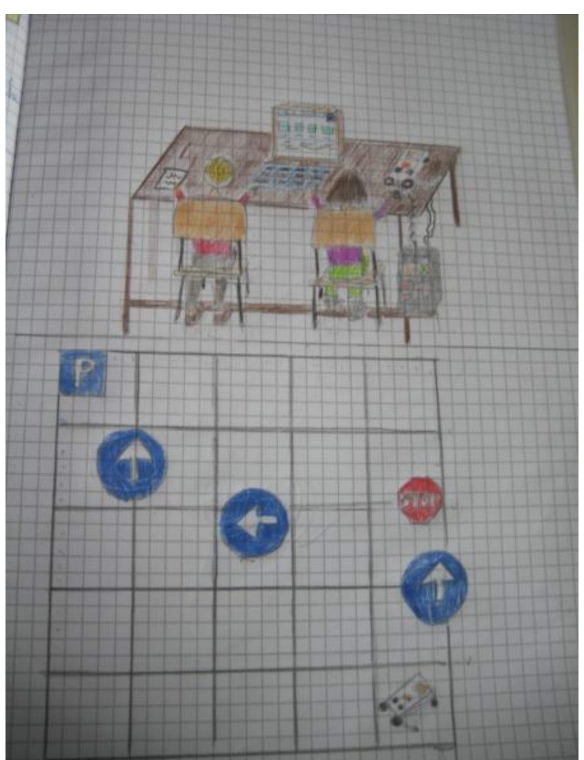
- 1/3rd Primary schools
- 1/3rd Middle schools
- 1/3rd High schools (science and language) & Technical Schools



Educational Robotics at Primary school: 3rd year

1. Building the robot:
- 3D mental representation
- understanding directions
- Interpret the correct sequence of instructions

2. Program the motion on a square with predefined commands:
- Introduction to programming
- Mental representation of motion
- Left & right spatialization



3. Program a path for the robot following road signals::
- The LEGO NXT-G Language
- More complex program environment
- Procedural programming
- Basic flow charts
- Programming constructs

4. Sense obstacles and traffic lights:
- Robot’s sensors (light and ultrasound)
- More complex flow charts
- Programming constructs

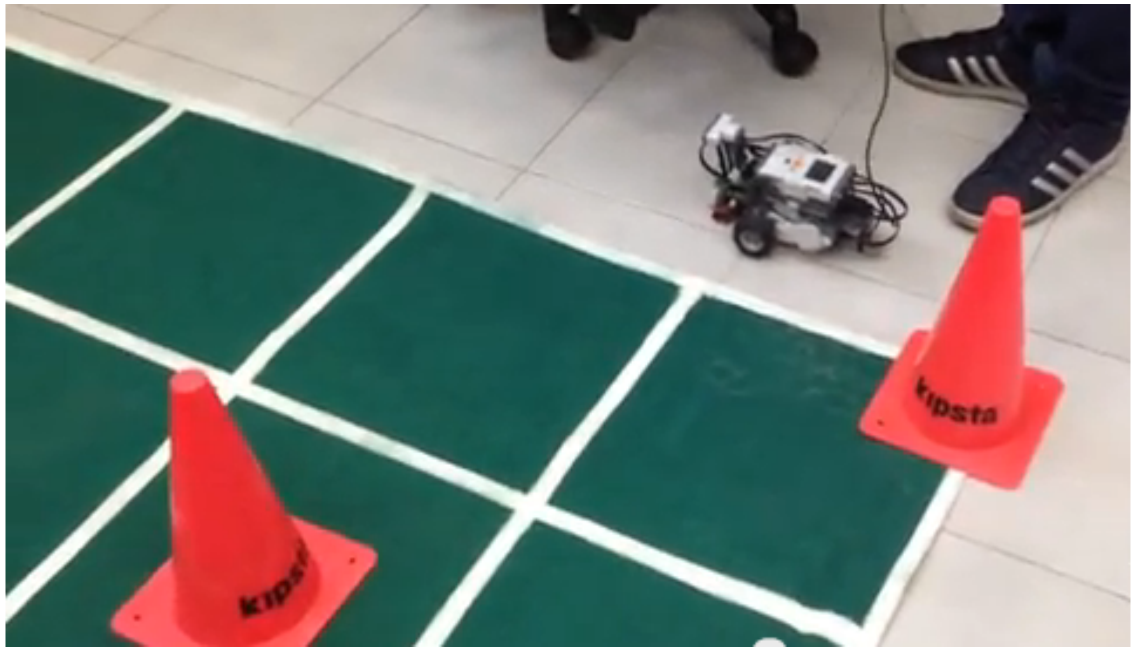
UNIVERSITÀ DEGLI STUDI DI PADOVA

Robotics at The University

- A mix of theoretical lectures in class and practical experiences in the laboratory
- Lectures deal with robotics fundamentals: perception, motion planning, kinematics, and navigation

- In lab, software tools are introduced and students are requested to implement some algorithms presented in the theoretical lessons.

Robotics in the last year of Master in Computer Science

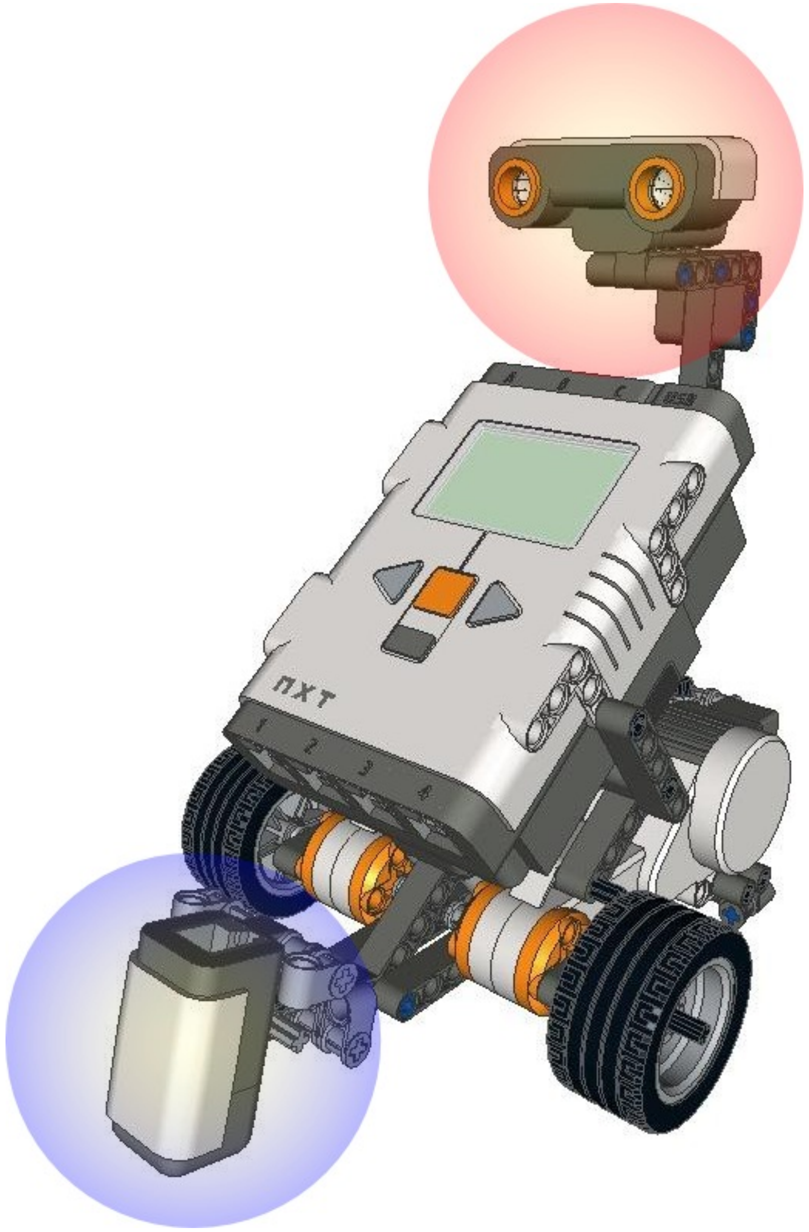


- In our CS master curriculum, just one course about Robotics
- 5 incremental experiences in a constructivist approach to tackle the complexity which is behind the building of autonomous robots

- Didactical objectives
  - teach about autonomous robots
  - Experience with wheeled robots (NXT) and humanoid robots (Robovie-X)
  - Improve programming skills and good programming practice
  - Experience with complex software framework

LEGO Mindstorm NXT

- Lego Mindstorms NXT is a flexible and cheap architecture with known capabilities
- It is widely adopted for undergraduate & master robotics courses
- It is technically documented (for example, you can develop your own I^2C-interfaced sensors and define proprietary BT-supported message-passing protocols)
- It has already been integrated in some known framework (e.g. URBI, MRS, ROS and others)



In Previous years…

- NXT may be programmed in NXC, a procedural C-like language
- Limitations when using NXC
  - low level of structuring, only procedural paradigm
  - strictly targeted to this type of robot
  - modularity is not perceived as a relevant issue
  - tendency to develop ad-hoc and not very general solutions
  - no powerful subsidiary tool available (e.g. simulation, data plotting, debugger etc.)

```
task play() {
  while (true) {
    PlayTone(TONE_A4, MS_500);
    Wait(SEC_1);
  }
}

task drive()
{
  while (true) {
    OnFwd(OUT_A, 50);
    Yield();
  }
}

task main()
{
  Precedes(drive, play);
}
```

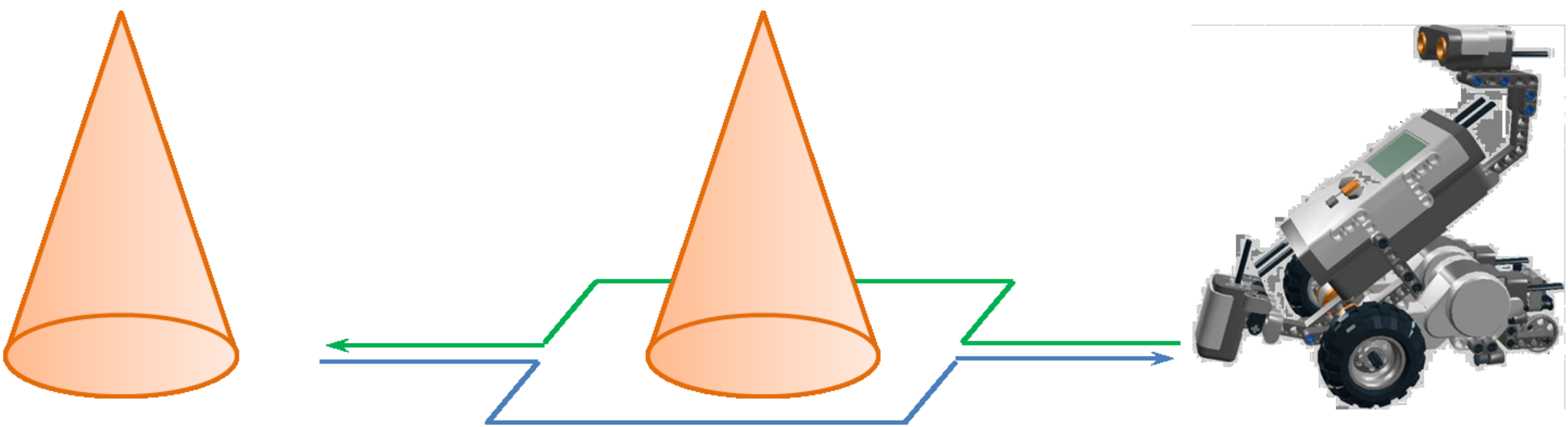
Why ROS?

- Choosing a widely spread framework you get several advantages (and some risks):
  - more robots and more programming language
  - you can exploit powerful added components and tools
  - you start practicing with a ‘real’ development environment, with a lot of documentation and supported by a large community
  - you have to face the complexity of a large framework
  - you have to do it in the temporal perspective of a university lab
- With respect to other large frameworks, ROS is recently becoming a *standard de facto*
- The introduction of ROS did not ask for a complete reformulation of experiments performed in previous years

Experience 1
Obstacle avoidance

The experience

- To avoid obstacles (cones)
- A first obstacle must be avoided, whereas a second one represents a place where to wait for a while and then return to the start position



Robotics objectives

- To get acquainted with robot, motion, sensors and other basic stuffs
- To work with basic ROS modules: master, nodes, fundamental functionalities
- To use a simple visualizer showing the 3D model of the adopted NXT construction (tribot)
- To develop new modules in ROS, exploiting the publisher/subscriber mechanism

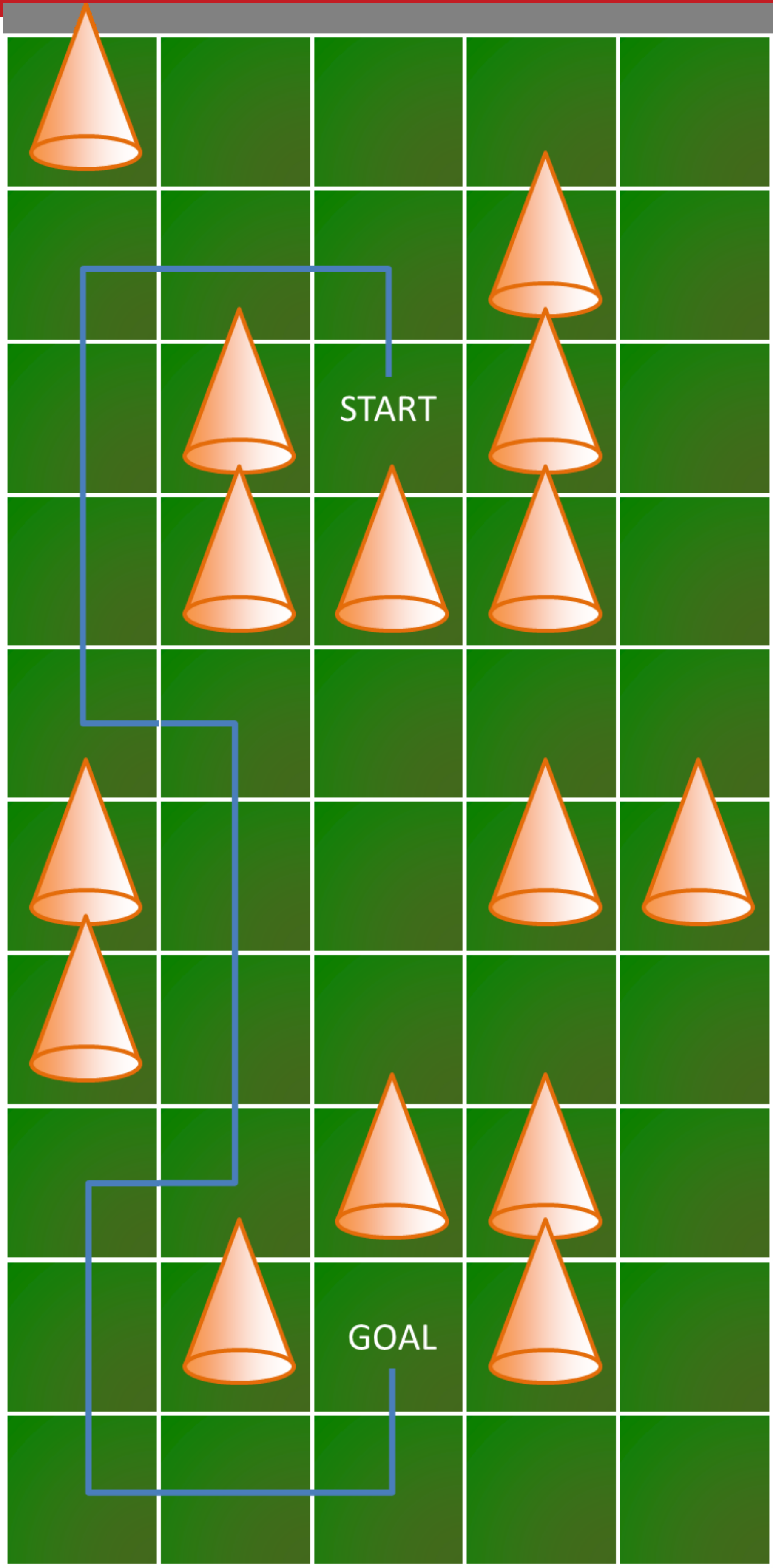
Computer science objectives

- To review basic concepts like data structures and classes
- To analyze the ROS package and message structures
- Publisher/subscriber communication paradigm via callback mechanism
- Though, this first example can be easily implemented in procedural programming, an object oriented solution was also promoted

Experience 2: Path planning

The experience

- To navigate into a map of NxM cells, partly occupied by obstacles
- The grid forces a path with segments parallel to the map edges, ending at the goal cell and avoiding the obstacles
- A second NXT equipped with a touch sensor is placed in the goal cell and used as a button to send a 'stop' to the first robot
- Obstacles may be fixed or movable

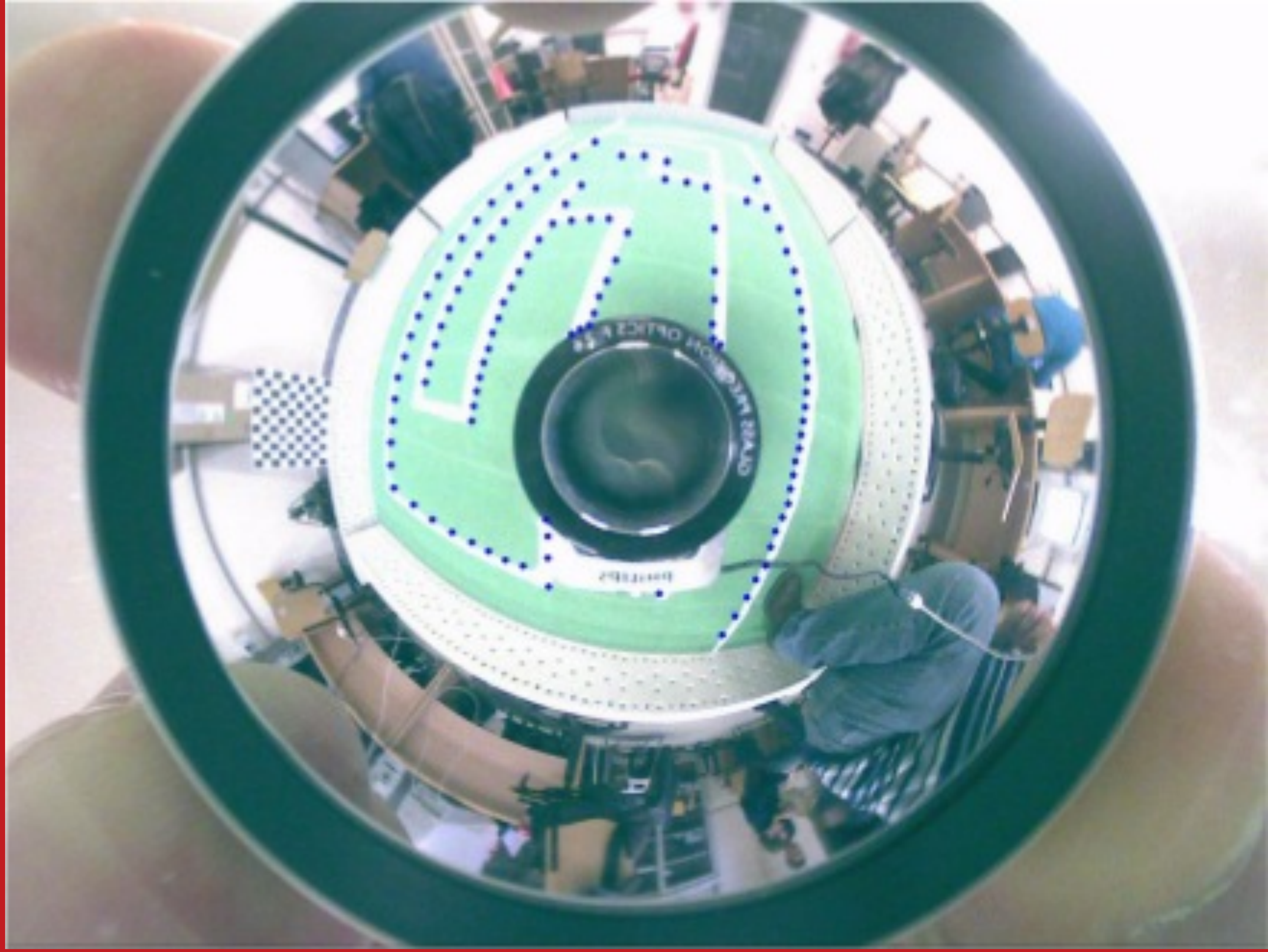


Robotics objectives

- To choose a suitable path planning algorithm, including obstacle avoidance
- To elaborate simple sensor data using ROS modules
- To effectively design the solution as a set of ROS interconnected modules
- To think about some fundamental topics like localizing and mapping, multiple-robot coordination

Computer science objectives

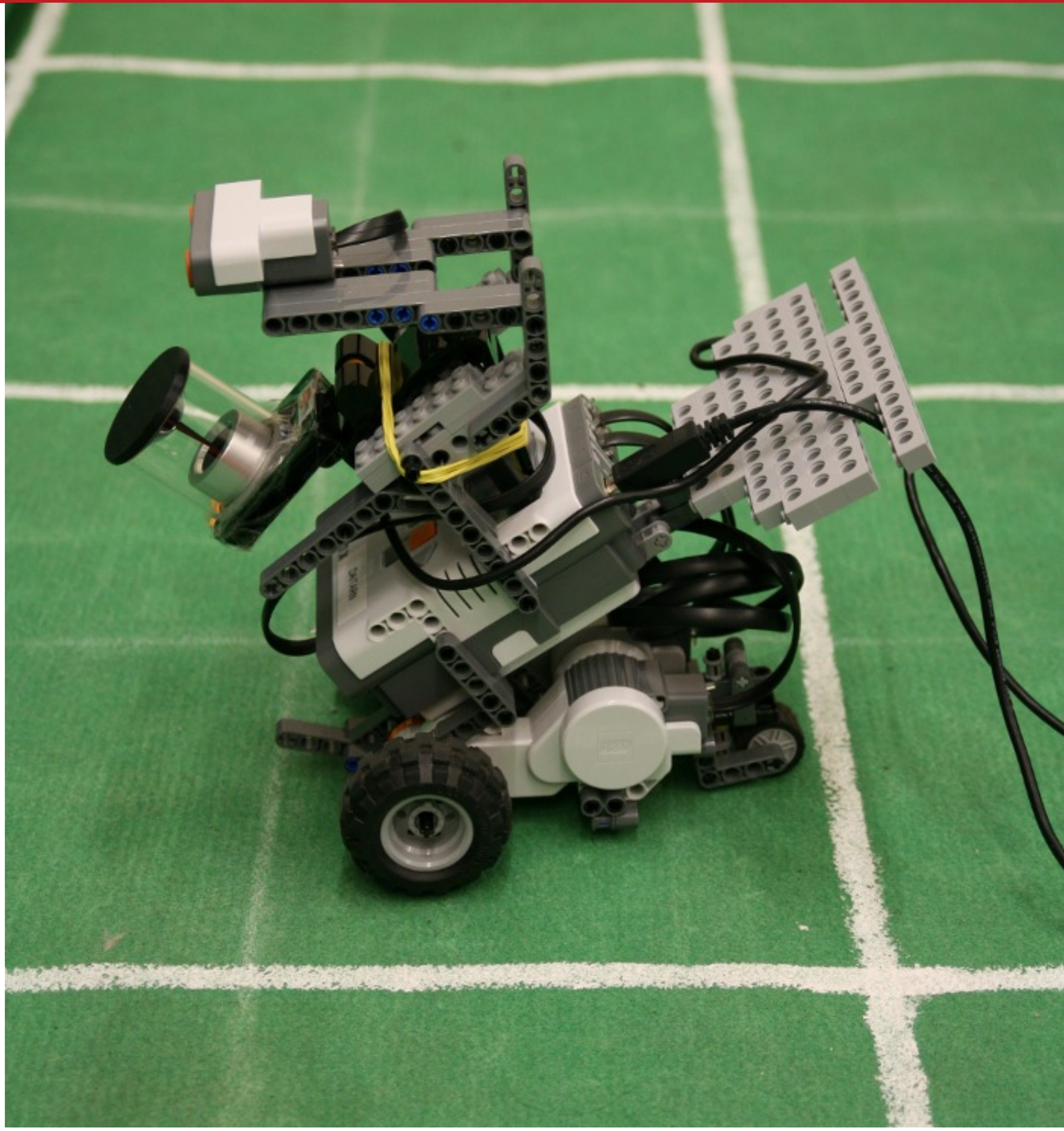
- To appreciate the value of a well-structured software design
- To map real elements (the robot, the map) onto software objects
- To implement the data structure for a dynamic map
- To solve the problem of the two similar, but different in behavior, communicating robots through a suitable level of abstraction of the robot class



Experience 3
Perception using computer vision

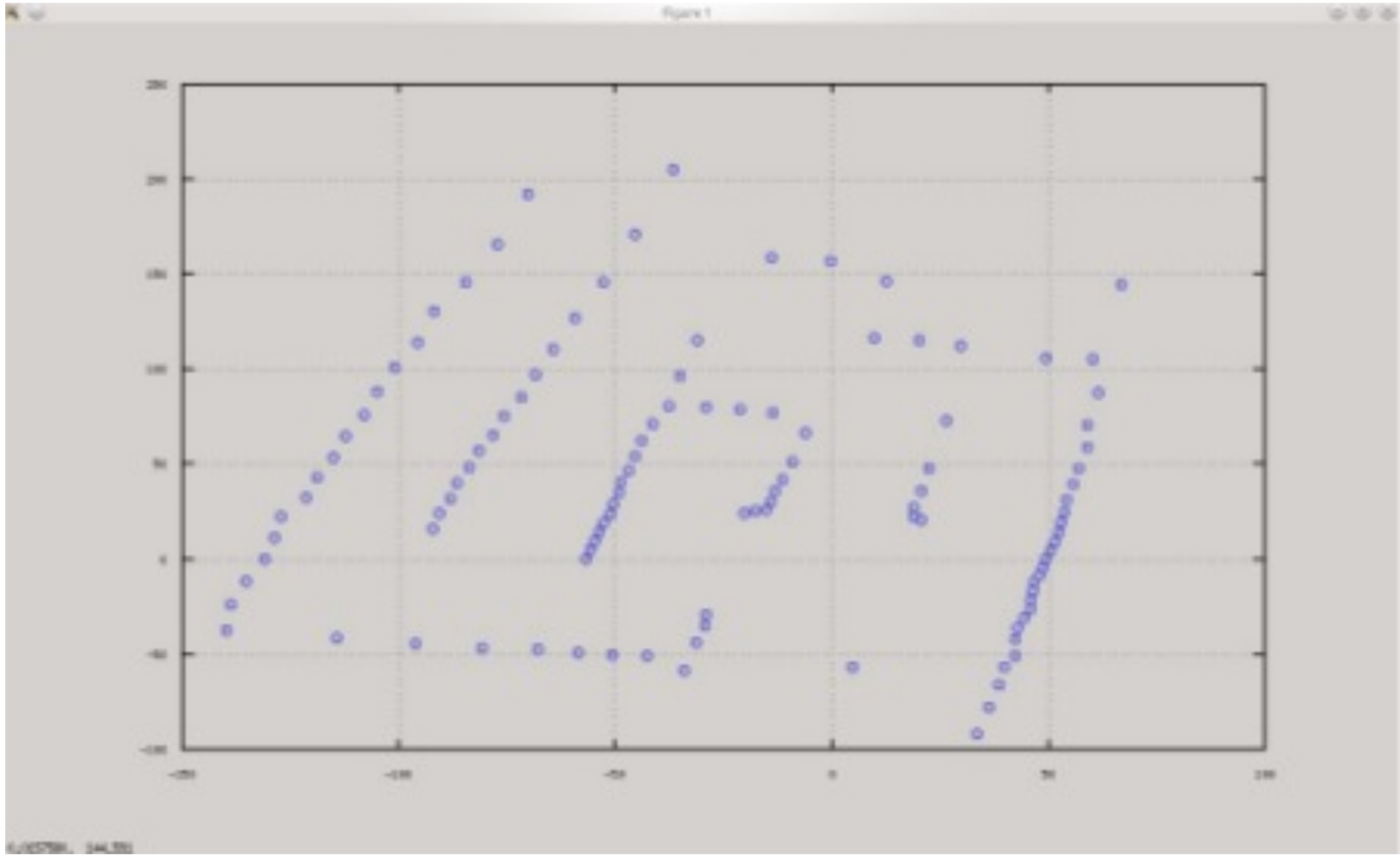
The experience

- The light sensor is replaced by an omnidirectional camera)
- Use of a camera calibration algorithm
 (Calculate height and inclination of the mirror's axis)
- First step: to reproduce the same output of the light sensor module
- Second step: exploit the full potential of the camera (odometry)



Robotics objectives

- To choose the most suitable algorithm(s)
- To face the tradeoff between complexity and accuracy
- To face the complexity of elaborating omnidirectional images



- To exploit the ROS designing approach (i.e. to develop an image acquisition module connected to a image processing module)

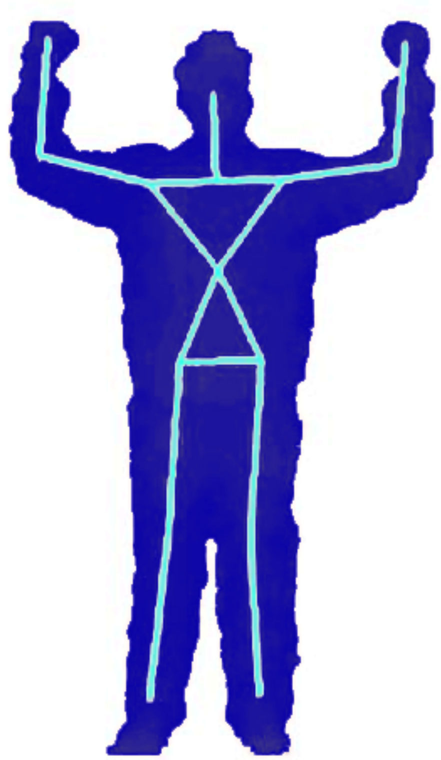
Computer science objectives

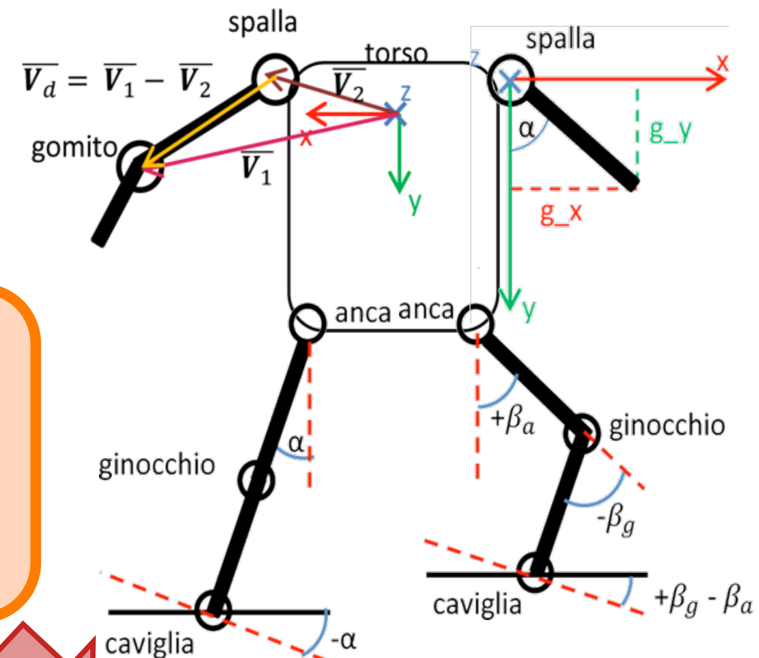
- To extend the previous sensor module design (exploiting code modularity, class inheritance, good design patterns)
- Students who had already developed a good software design in the previous experience, could more simply apply a principle of reuse
- The Others have to reimplement the software and are fostered to reimplement everything

Small Humanoid Experience

RGB-D Sensor

Skeleton tracking

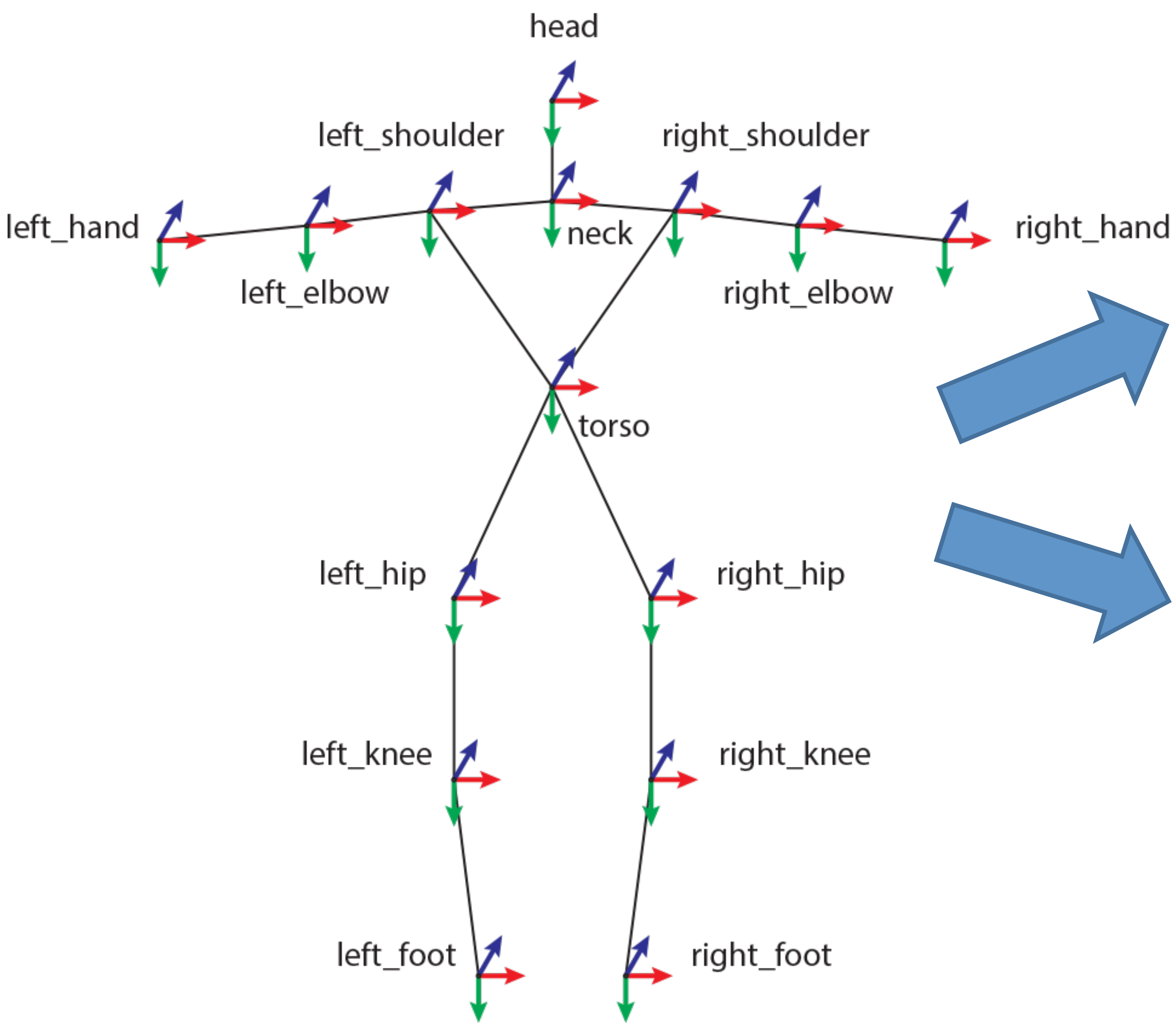


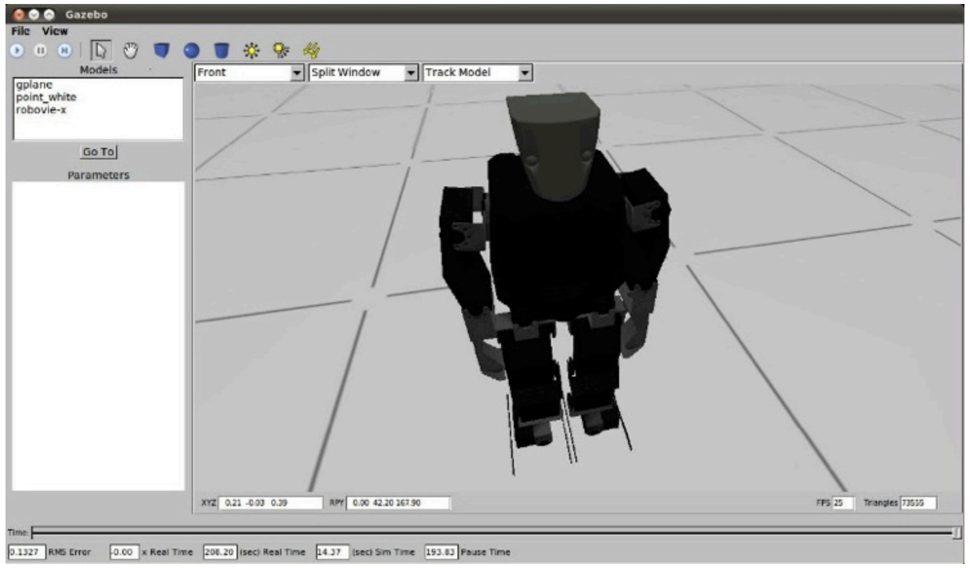


Joint mapping

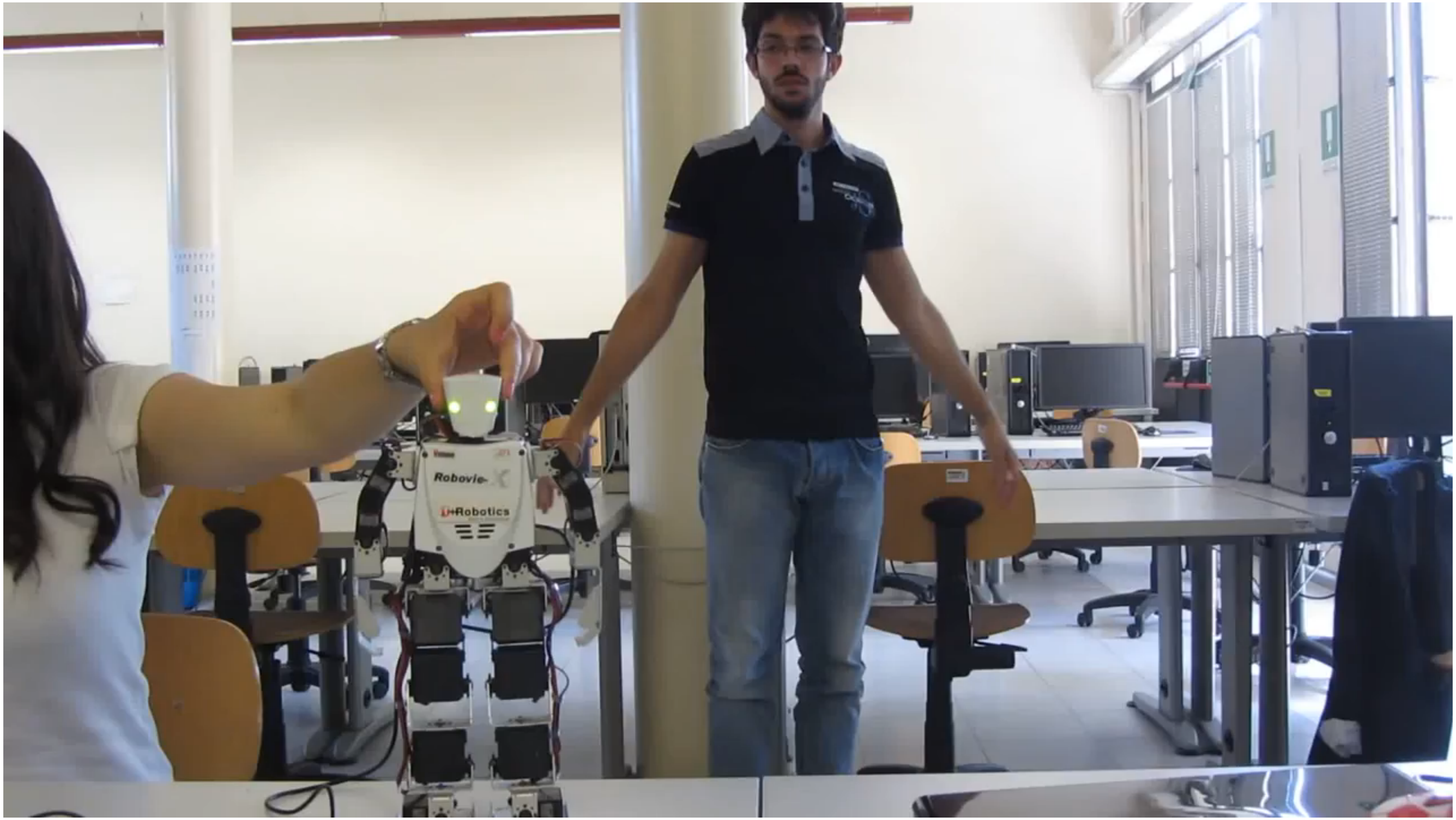
Robot control

Exp 1: Motion remapping





Exp 1: Motion remapping



Computer science objectives

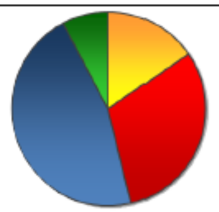
- Get acquinted with RGB-D sensors and large data structures (RGB-D images as 3D point clouds)
- Experience the power of the publisher/subscriber communication paradigm
- Implement a publisher/subscriber system

Exp 2: Robot stabilization

Computer science objectives

- Implement algorithm with feedback and error correction
- Experience with soft real-time systems
- This final result can be achieved in time only if good software prectices have been implemented and good software reuse is exploited

Legend: Not at all   A little   Enough   Very much

Q1.	During the experiences of the robotics course I exploited knowledge acquired in other courses that I had not put into practice.	

Evaluation of didactical impact

Evident advantages when using ROS

- Students are forced to organize their software into modules, reuse their data structures and classes, exploit class inheritance
- They experienced the role of the message sharing mechanism in a robotics framework
- When problems are correctly formalized and the software design of the solution is appropriated….

…students feel the difference!

Analysis of the students' software

- The software produced by 16 students (in groups of two) was analyzed to extract quality indicators in terms of:
  - potential of **code reuse** (functions/methods applicable in different situations)
  - fusion of common or similar characteristics into general **data structures**
  - modeling of real entities into suitable **abstracting classes**

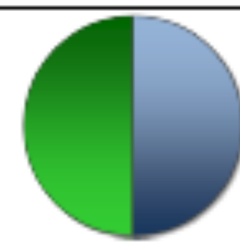
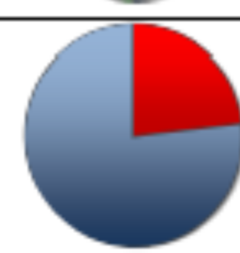
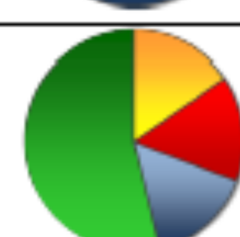
	1^{st} exp.	2^{nd} exp.	3^{rd} exp.
Code reuse	75%	100%	100%
Structured data	38%	88%	100%
Classes	63%	88%	88%

Students' satisfaction

- A questionnaire distributed to the students with four levels of evaluation (Very much, Enough, A little, Not at all)
- Aspects evaluated:
  - exploitation of students' programming background skills and software engineering
  - effort spent in developing the experiences
  - closeness with future job activities
- For every question we represent the overall results with a pie chart

Students’ satisfaction

Legend: Not at all | A little | Enough | Very much

Q1.	During the experiences of the robotics course I exploited knowledge acquired in other courses that I had not put into practice.	
Q2.	Programming skills I developed in previous courses were sufficient for working on the ROS framework.	
Q3.	I gained new programming experience because I had to work with a complex framework as ROS is.	
Q4.	In order to efficiently work with ROS, I have been forced to divide my software into modules, which in turn made it easier to reuse it.	
Q5.	In my future job I will be asked to work with modular software structures similar to ROS.	
Q6.	By working with ROS I have got in touch with a well structured software, that has been a source of inspiration for new ideas on how to develop my software.	
Q7.	By working in groups I improved my ability to split tasks among people.	
Q8.	By working in groups we were able to reach results I would not have reached alone.	

Conclusions

The results of the questionnaire show that choices made while designing the course had a good impact over several important aspects:

- The introduction of ROS has resulted an incitement to use OOP concepts
  - ROS framework effectively forced students to adopt a more modular approach
- The overhead undergone to learn a new framework is compensated by the later ease to develop code of increasing complexity
- Students gave value to team working
- Working with a known 'real' industrial framework is a rewarding effort for master students
- Students appreciated the hands-on approach of the course, in spite of the increasing of the work load
- They also appreciated the way experiences gradually increase in complexity
- Good response both regarding how students' expectations were met and the improvements in robotics and programming skills.

Conclusions

- Need for curricula in project based learning with team work
- Student should be invited to explore, observe, manipulate, ask questions, formulate hypothesis, collect data and biuld his/her own knowledge on the topic (constructivism)
- Teachers are enforced by their tutorial role and role of "learning guide"
- Useful to initiate external collaborations with companies and scientific dissemination initiatives.

Università
degli Studi
di Padova

Robotics in Computer Science Education

Prof. Emanuele Menegatti
Intelligent Autonomous Systems Lab (IAS-Lab)
Dept. of Information Engineering
School of Engineering
University of Padova, Italy

Karlsruhe October 24 , 2014