Brandeis Robotics

May 28-29 2024

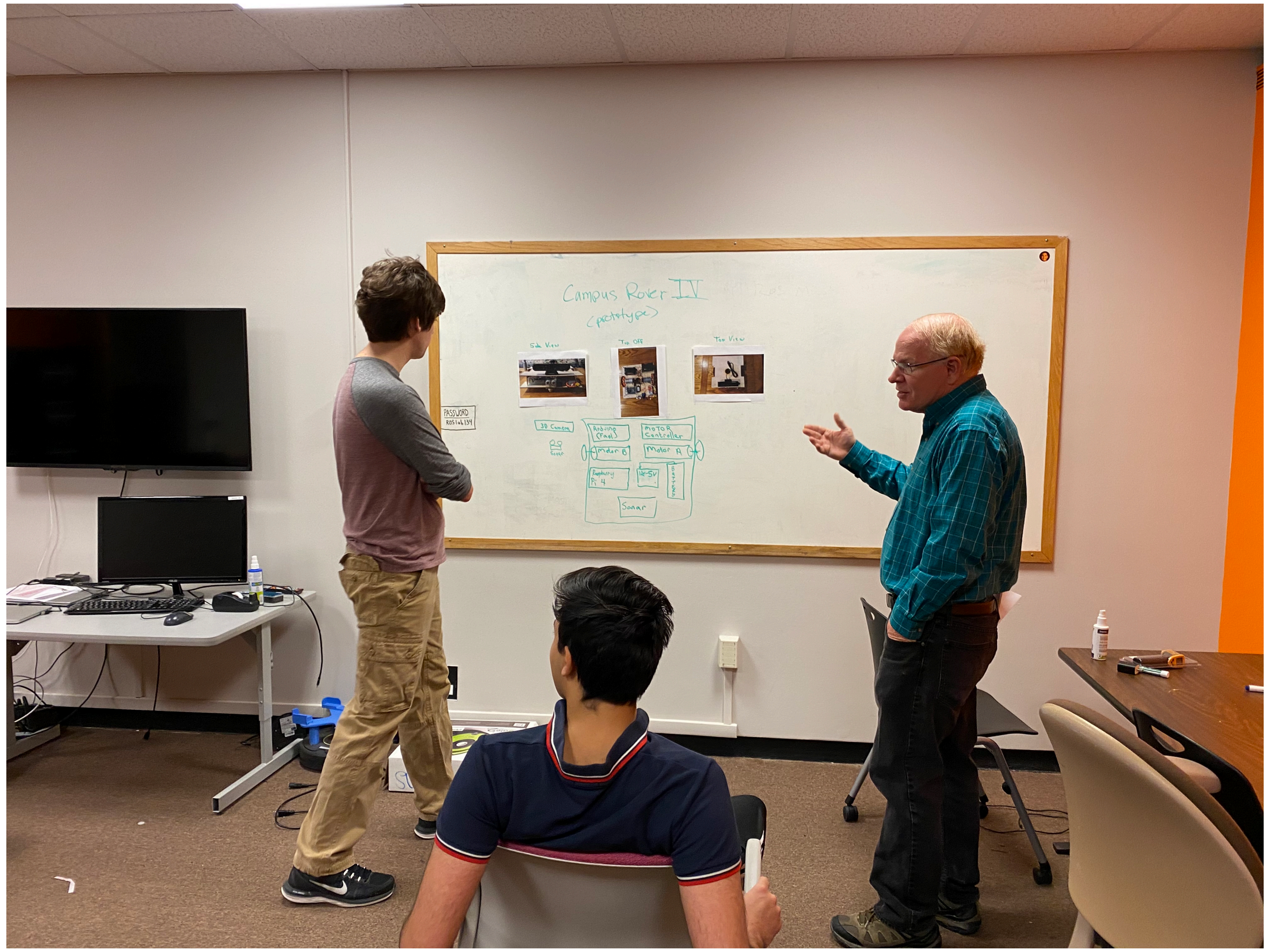
Pito Salas

The Teaching Lab

- Capacity around 20 students (packed)

Teaching Lab

- Staff:
  - Pito Salas (me)
  - Charlie Squires ("Roboticist In residence")



Our Robot Gallery

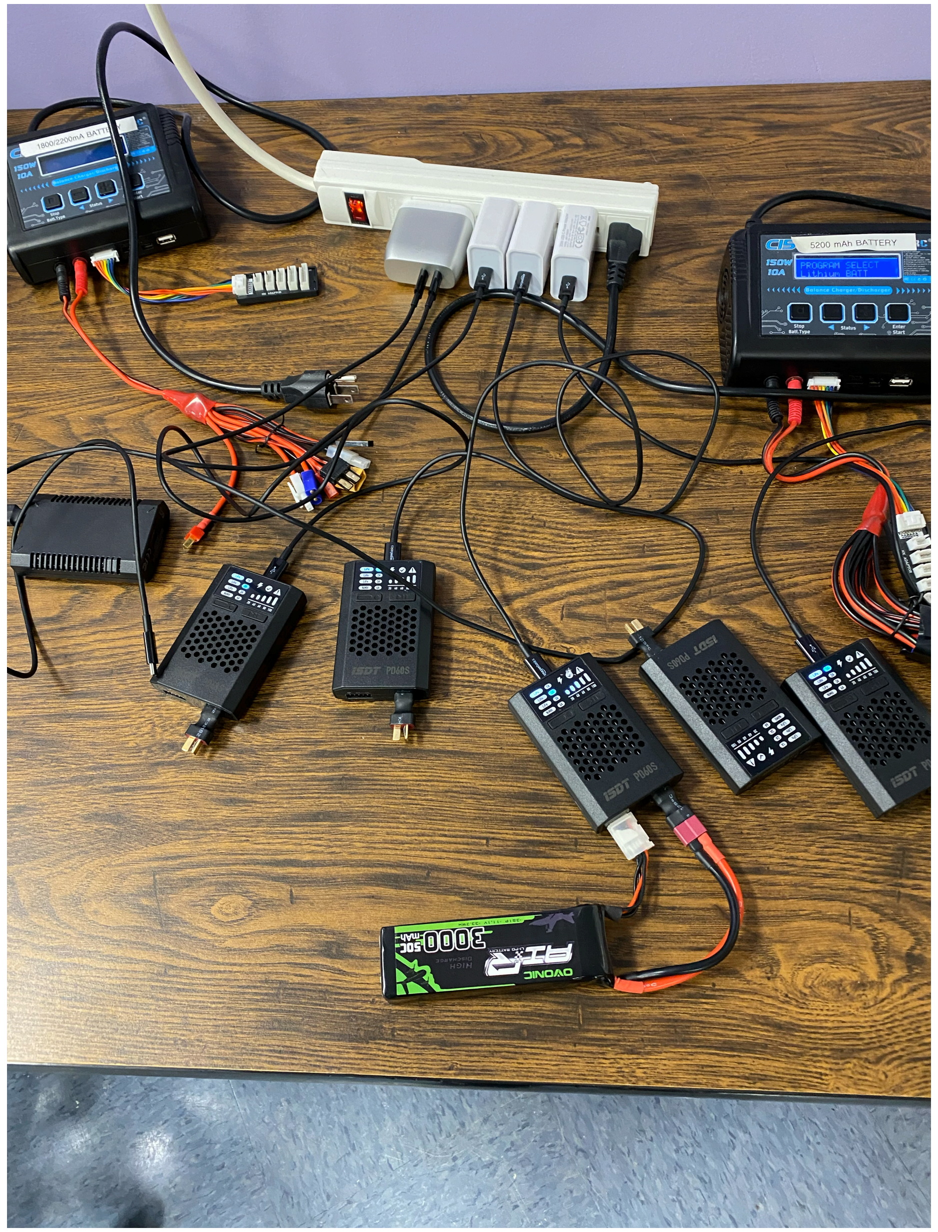
- Robots (Some commercial and some home grown)

WHy home grown?

- Learning/teaching how they work
- Cost
- Outdoor capable
- Customizability

What else we have

- Lots of parts, tools, devices



Course: Cosi 119a Autonomous Robotics

- Structured around "Seven Big Ideas" paper
- An *applied computer science* course
- Theory + Programming assignments + Term Project
- Ever growing scaffolding

Seven Big Ideas in Robotics, and How To Teach Them

David S. Touretzky
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
dst@cs.cmu.edu

ABSTRACT

Robotics is widely recognized as an interdisciplinary mixture of engineering and computer science, but the latter component is not well represented at many undergraduate institutions. The sophisticated technologies that underlie perception, planning, and control mechanisms in modern robots need to be made accessible to more computer science undergraduates.

Following the curriculum design principles of Wiggins and McTighe (*Understanding by Design*, 2^{nd} Ed.), I present seven big ideas in robotics that can fit together in a one semester undergraduate course. Each is introduced with an essential question, such as *"How do robots see the world?"* The answers expose students to deep concepts in computer science in a context where they can be immediately demonstrated. Hands-on labs using the Tekkotsu open source software framework and robots costing under $1,000 facilitate mastery of these important ideas. Courses based on parts of an early version of this curriculum are being offered at Carnegie Mellon and several other universities.

and geometry and linear algebra [13]. But many computer science departments still aren't teaching serious robotics at the undergraduate level. Those that do offer a robotics elective are all too often teaching a high school curriculum where students assemble primitive robots from parts kits and program simple-minded reactive controllers. This paper is a call to action: we *can* make sophisticated robot technologies accessible to undergraduates. Doing so will help them develop into better computer scientists.

In this paper I present seven "big ideas" in robotics that can fit together in a one semester course for junior or senior level computer science majors. Following the *Understanding by Design* methodology of Wiggins and McTighe [17], each big idea is introduced with an essential question, such as *"How do robots see the world?"* The answers to these questions involve deep computer science concepts, some of which can only be fully mastered in graduate classes. But exposing undergraduates to these ideas – at an appropriate level – can broaden their understanding of computer science and encourage them to learn more. All of these ideas are implemented in the Tekkotsu software framework [11, 12] and can

Current Projects

- Next Generation *BranBot*
- Arm Pick and Place

BranBot

- Outdoor capable
- "Kitified"
- Low cost (~$500)

Pick-and-place

- Robot Arm Control
- Creating new scaffolding for students
- One of the classic robotics and automation challenges
- "A general pick-and-place implementation using inexpensive robotic arms"

README

What

PNP can simulate any factory floor that can be described within the constraints of its JSON config file (`/config/config.json`).

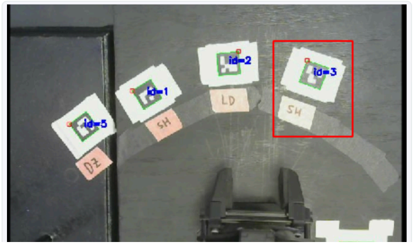
For instance, PNP ships with an example floor that has two tasks: washing that takes 3 seconds, and welding that takes 2 seconds.

```
"tasks": [
    {
        "id": "wash",
        "duration": 3
    },
    {
        "id": "weld",
        "duration": 2
    }
]
```

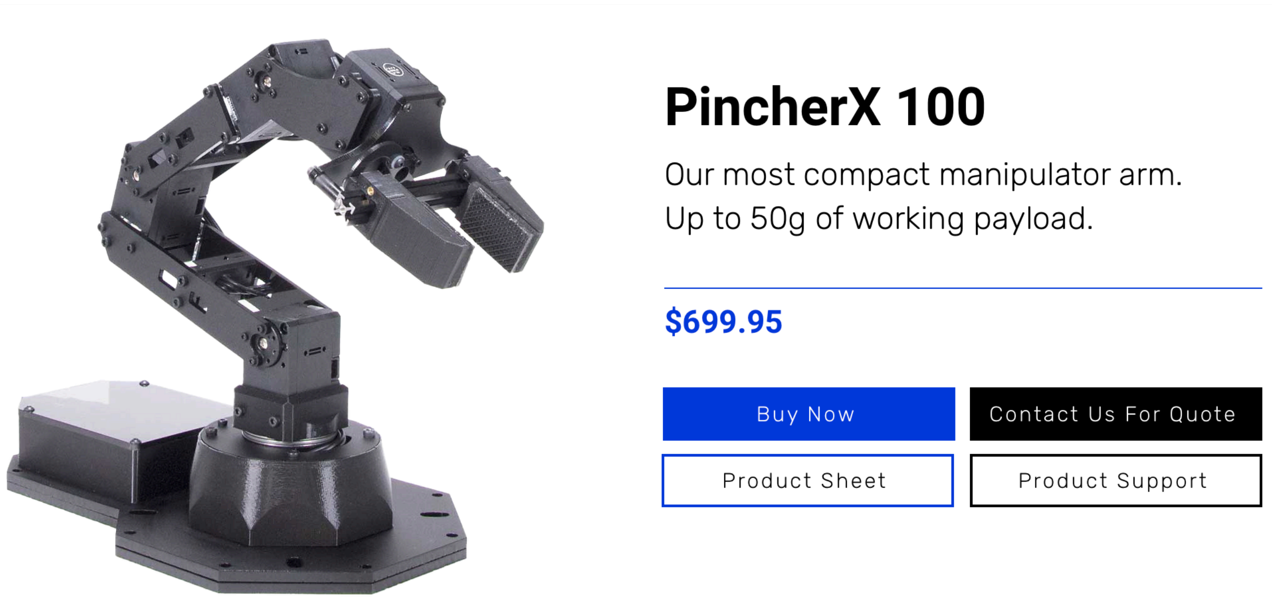
Tasks are executed at stations. The same task can be executed at more than one station. For example, we see below that washing happens at both `station_0` and `station_2` . This means that if a part needs to be washed, but `station_0` is occupied, it can be handled in `station_2` .

```
"stations": [
    {
        "id": "station_0",
        "fid_id": 1,
        "task_id": "wash",
        "free": true
    },
    {
        "id": "station_1",
        "fid_id": 2,
        "task_id": "weld",
        "free": true
    },
    {
        "id": "station_2",
        "fid_id": 3,
        "task_id": "wash",
        "free": true
    }
]
```

Thus, the mapping from tasks to stations can be one-to-many. Stations are represented by fiducial markers. We see above, for instance, that `station_2` is paired with a fiducial of id 3. The physical fiducial is marked in red below.



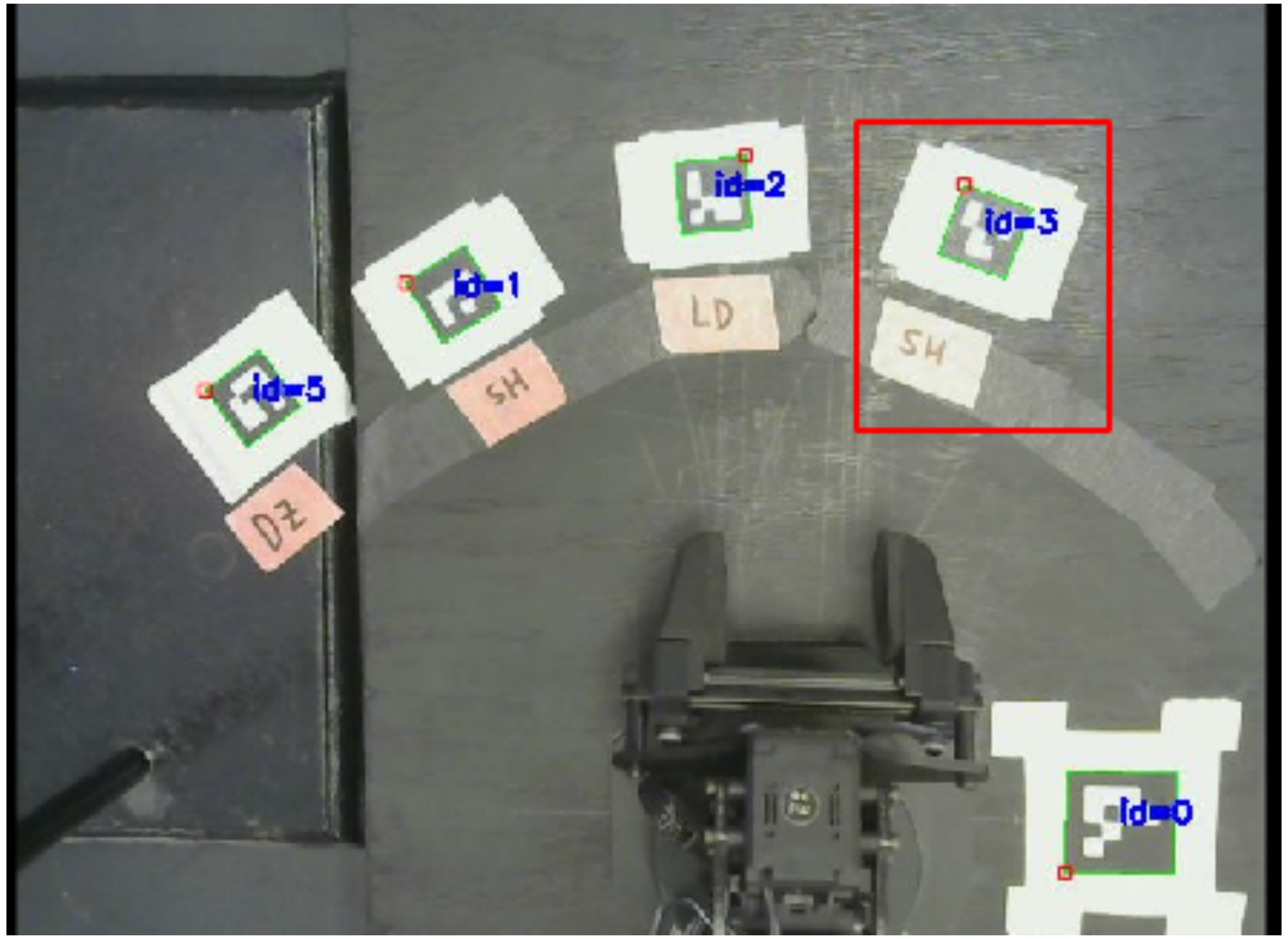
Challenges



1. The least expensive-non-toy Robot arm we could find
2. Four degrees of freedom

Overview

1. Target "payloads" have to be layed out in a semi-circle
2. Identified by fiducials



Controlling the factory floor

- Cargos, Stations, Processes
- e.g.
  1. Cargo "bracket" requires Process "weld"
  2. Station "welder-1" can perform Process "weld"
  3. Process "weld" requires

Demo

End