Robots in the Swarm

Integration of a robotic system
into an IoT environment via accessors

Penn Engineering
GRASP LABORATORY
General Robotics, Automation, Sensing & Perception Lab

Berkeley EECS
ELECTRICAL ENGINEERING & COMPUTER SCIENCES

Marcus Pan (UPenn)
Amanda Prorok (UPenn)
Philip Dames (UPenn)
Mark Oehlberg (Berkeley)
Edward A. Lee (Berkeley)

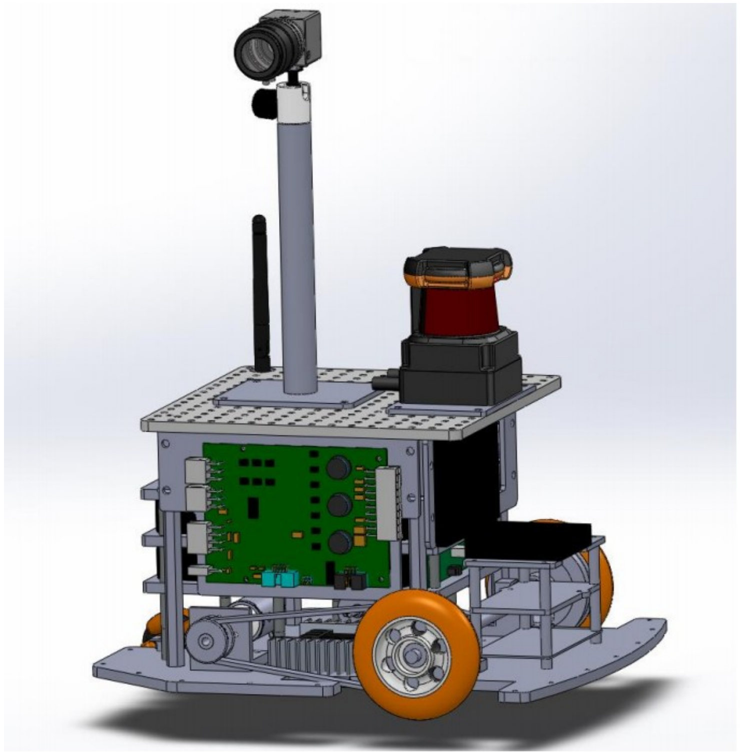
Robots in “A Vision of Swarmlets”

“Consider a scenario in which a startup company produces a robot...that wanders a space, such as a factory floor, that already contains a variety of networked sensors, security devices, and other robots made by other vendors.”

-E. Latronico et al., “A Vision of Swarmlets”

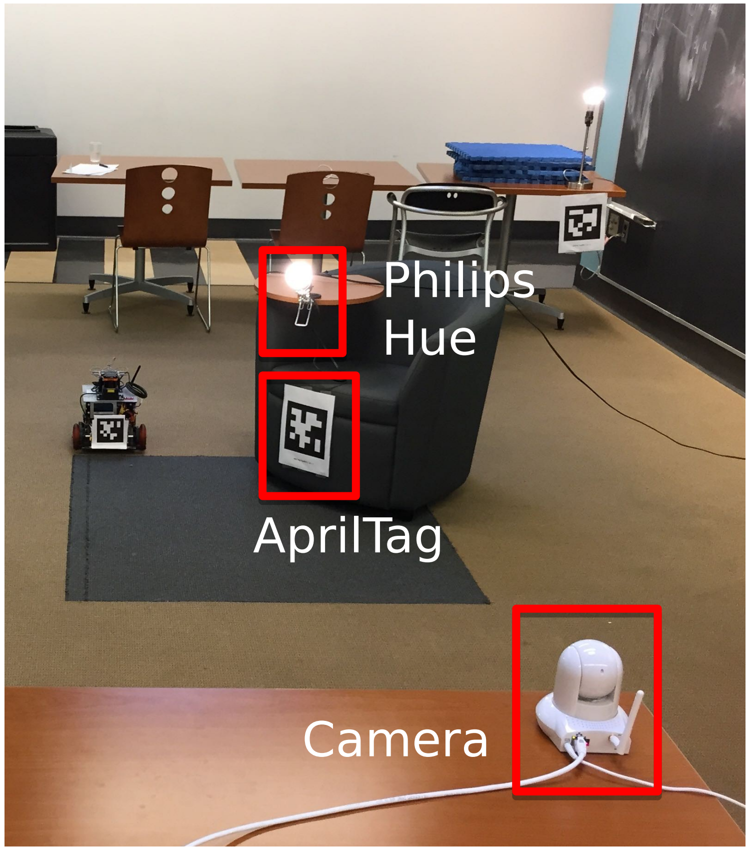
- The IoT has vast applications for robotics
- However, there are various challenges with integrating a tightly controlled robotic system into a variable, diverse IoT environment
- I will present a case study of how Ptolemy can be used to integrate these two environments

A Robotic System – The Scarab



- Onboard Computer
  - Ubuntu 14.04, ROS Indigo
- Hokuyo Laser Sensor
  - Distance measurements up to 30m
  - 270° field of view
- Software Algorithms
  - Laser Odometry – maps its surroundings using laser scans
  - Localization – tracks the robot's pose against a fixed map
  - Navigation – plans a path toward a waypoint

An IoT Environment



Consider an environment that has a few Philips Hue light bulbs that are labeled with AprilTags. A Foscam IP camera records a video of this environment. The light bulbs and camera are controlled over a LAN.

Can we add a robot to this environment, make it travel to the light bulbs and turn them on?

Real-world applications:

- Agriculture: robot patrols plant beds and controls light conditions
- Traffic: autonomous vehicles can detect light state of traffic lights

Yes We Can – Using ROS and Ptolemy

ROS

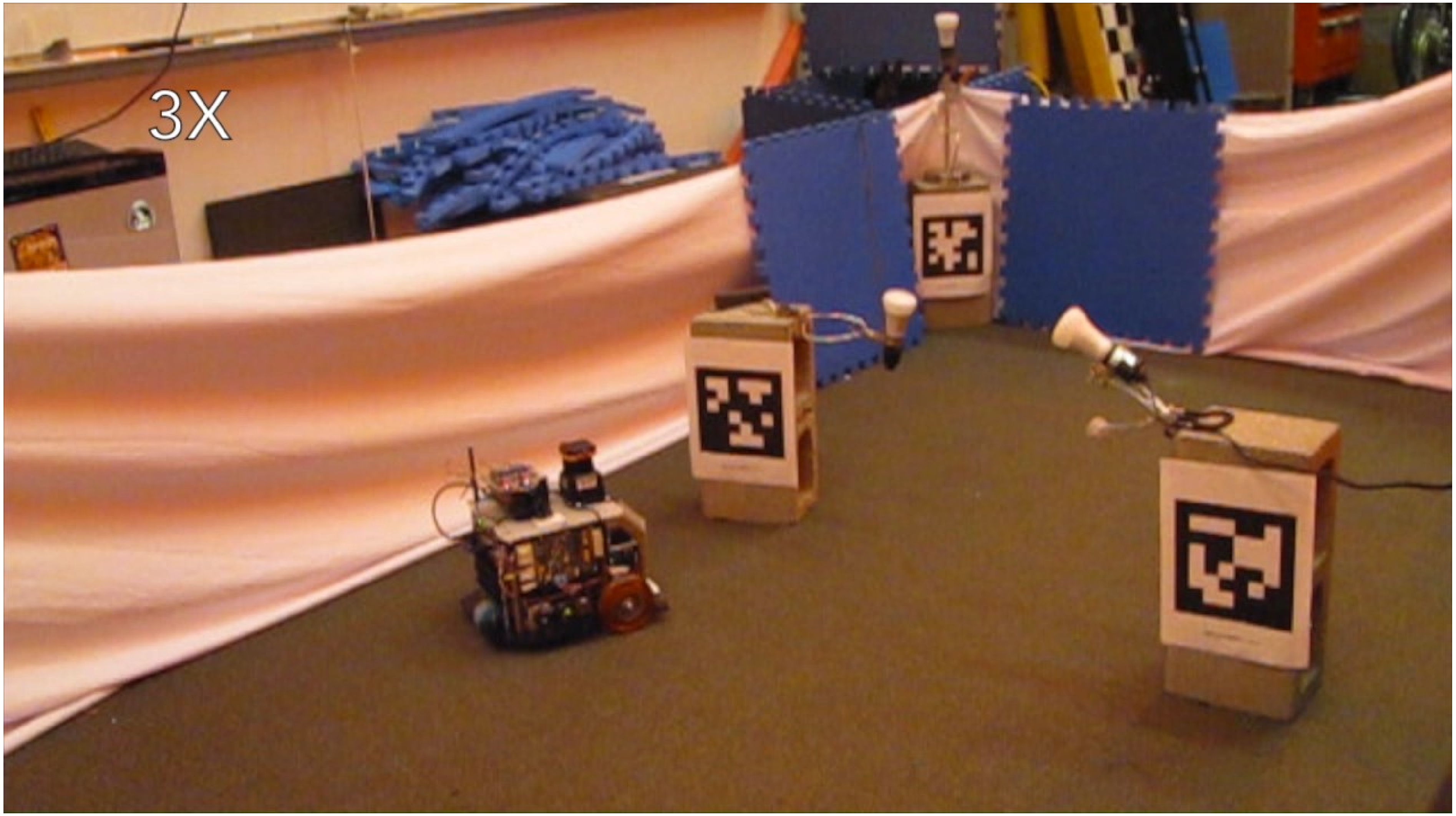
- Open-source software framework for many robotic applications
- Computational nodes communicate over ‘topics’ which are bound to a ROS datatype via TCP/IP
- Provides library called ‘rosbridge’ that exposes active topics to external programs via a web socket interface

Ptolemy

- Has a **`WebSocketClient`** accessor that publishes and subscribes to a web socket server
- Two accessors were created that extend **`WebSocketClient`**, **`RosPublisher`** and **`RosSubscriber`**
- ROS datatypes are treated as JSON objects by Ptolemy

It Works

https://youtu.be/CeTlpnGroxY



Control Loop – Ptolemy and ROS

ROS node creates map (occupancy grid) of surroundings using the laser scanner

ROS node subscribes to camera’s video feed of surroundings. AprilTags are detected and localized into camera’s coordinate frame. Tag coordinates are transformed from the camera frame into the robot’s map frame

Ptolemy subscribes to tag coordinates with a **`RosSubscriber`**. It computes an appropriate waypoint, and sends it to the robot via a **`RosPublisher`**

ROS node plans a path and travels toward waypoint, and reports status back to Ptolemy

success

Ptolemy turns on the Hue bulb attached to the AprilTag

waypoint unreachable. (AprilTag localization has +- 0.5 m error)

Ptolemy modifies waypoint

Division of Work – Ptolemy and ROS

Red work done in Ptolemy, Blue work done in ROS

Supervisory Control
- Monitoring robot waypoints

Interfacing with external Sensors/Actuators
- Turning on Hue bulbs
- Subscribing to IP camera video feed
- AprilTag detection and localization

Robot algorithms
- Laser odometry
- Robot motion
- Path planning

- Ptolemy is good at high-level control and I/O
  - Easy to program control loop in JavaScript
  - Library of accessors to communicate over different protocols
- ROS is good at robotics
  - Stable, strongly typed C++ environment
  - Software libraries for CV, matrix multiplication

A Cleaner Division of Work Would Be:

Red work done in Ptolemy, Blue work done in ROS

Supervisory Control

- Monitoring robot waypoints

Interfacing with from Sensors/Actuators

- Turning on Hue bulbs
- **Subscribing to IP camera video feed ****
- **AprilTag detection and localization ****

Robot low-level algorithms

- Laser odometry
- Robot motion
- Path planning

This can be done with further development of accessors for robotics

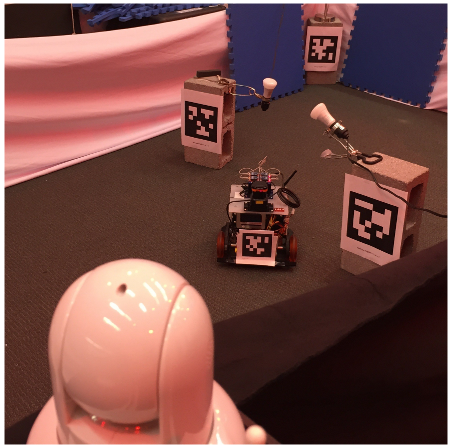
More Accessors for Robotics: 1. AprilTags

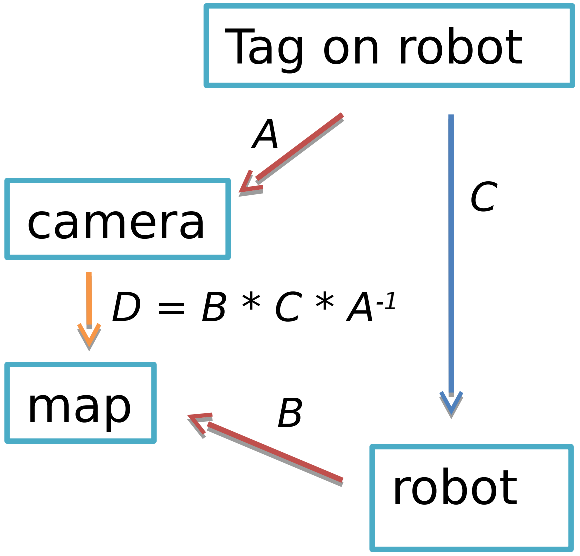
- Current AprilTag accessor outputs pixel coordinates of AprilTags
- If tag size and camera parameters are known, pose of AprilTag (a translation and rotation matrix) can be computed easily with OpenCV, using the **`solvePnP()`** function :

```
for each tag:
     s = tagSize/2;
     objectPoints = [(s, s, 0), (-s, s, 0),
     (s, -s, 0), (-s, -s, 0)];
     imagePoints = [(tag_UR_pix_coords), (tag_UL_pix_coords),
     (tag_LR_pix_coords), (tag_LL_pix_coords)];
     solvePnP(objectPoints, imagePoints, cameraMatrix,
     distortionCoeffs, rotVector, transVec);
```

Reference:
https://github.com/RIVeR-Lab/apriltags_ros/blob/hydro-devel/apriltags/src/TagDetection.cc#L76-L112

More Accessors for Robotics: 2. Localization





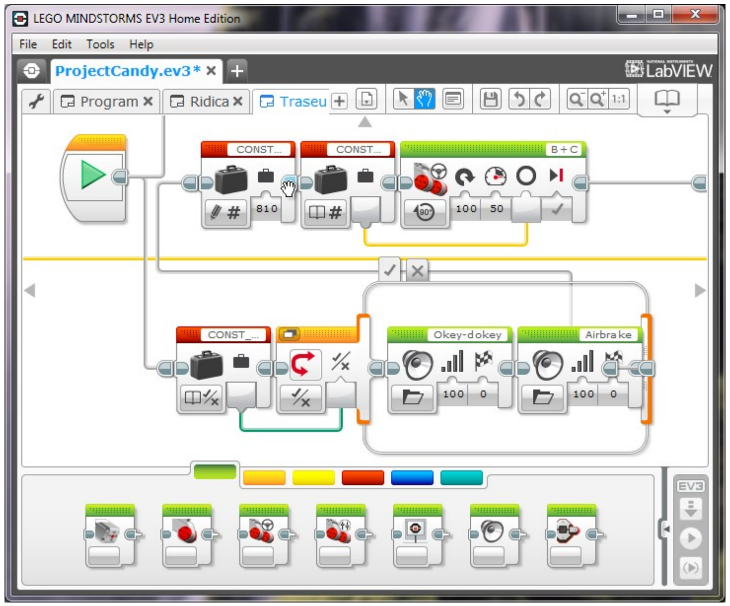
Ptolemy doesn’t have matrix multiplication libraries to do coordinate transformation. This would be useful to localize AprilTags from the camera’s frame to the robot’s map frame.

Pose matrix = [Rot | Trans]

A. Pose of tag on robot is known in camera’s coordinate frame
B. Pose of robot is known in map’s coordinate frame
C. Set pose of robot tag in robot’s frame
D. Get transformation from camera frame to robot’s map frame

Conclusion

- Ptolemy can provide supervisory control to a robotic system
- It abstracts away complex robotic algorithms into I/O blocks
- It opens the door to vast array of sensors through accessors
- Can it be used as an IDE to program sophisticated robots in an IoT environment?



Lego Mindstorms IDE