IROS 2017 Workshop on Shared Platforms for Medical Robotics Research

Progress Toward a Common API via ROS

Peter Kazanzides

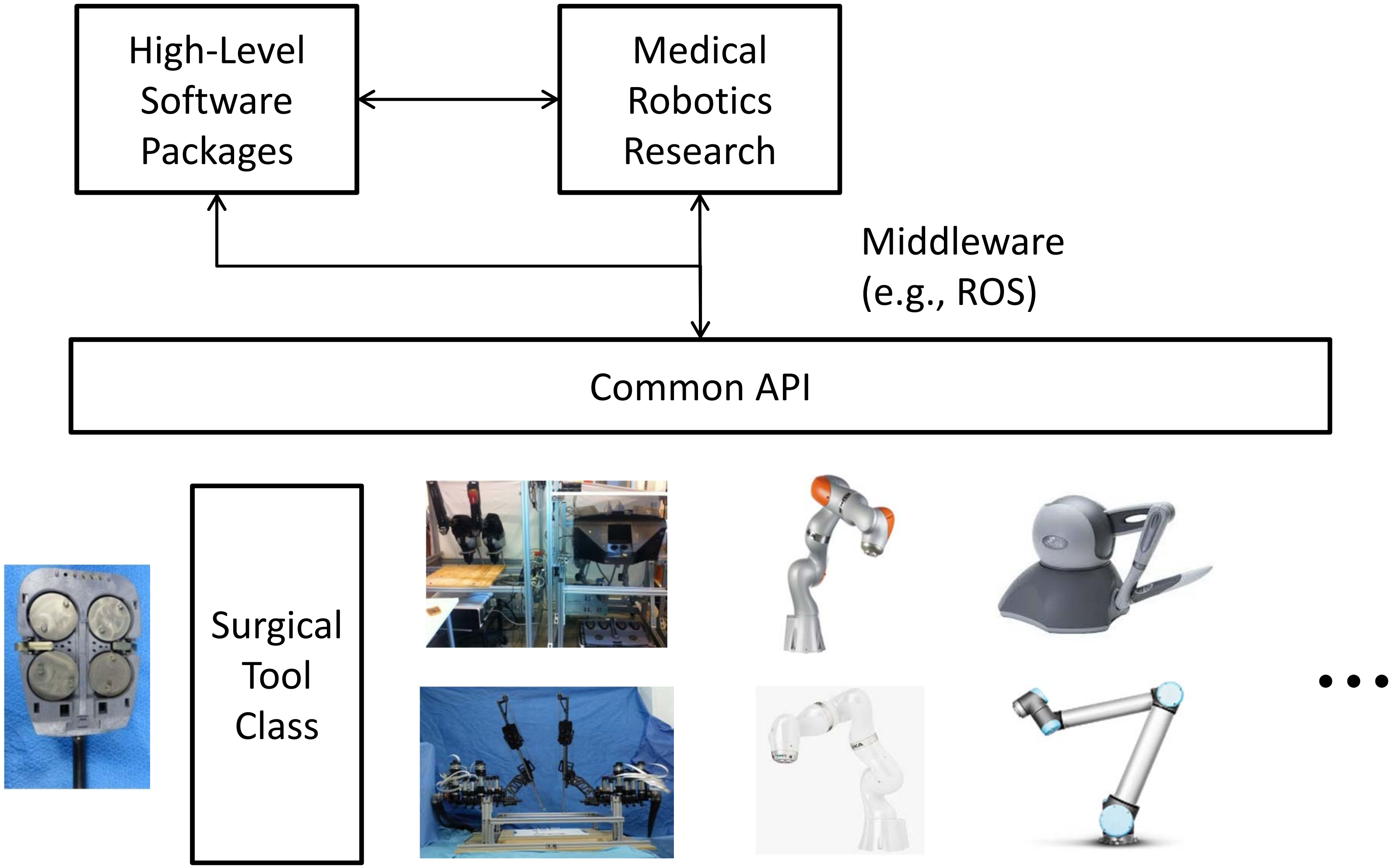
Johns Hopkins University

LABORATORY FOR Computational Sensing + Robotics THE JOHNS HOPKINS UNIVERSITY

W ELECTRICAL ENGINEERING UNIVERSITY of WASHINGTON

AIM WPI

Overview


High-Level Software Packages
Medical Robotics Research
Middleware (e.g., ROS)
Common API
Surgical Tool Class
• • •

Landscape

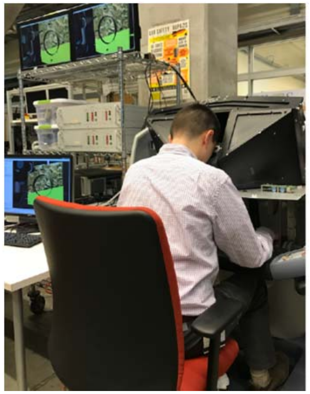
- ROS is de facto standard in robotics
  - Standard middleware (services, topics)
  - Some standard topics/services
  - Standard message types, but many variations
  - Researchers often write "glue" nodes to handle interface mismatches
- Need for further standardization
  - Canonical Robot Command Language (CRCL): primary focus on industrial robots
  - Our focus: semi-autonomous teleoperated and collaborative robots (*Collaborative Robotics Toolkit*)

Approach

- Collect use cases
- Develop API that satisfies use cases
  - GitHub *collaborative-robotics* organization
- API will be compatible with ROS, but not require ROS
  - Enable use of “real-time” middleware and cross-platform middleware
  - Define “topic” (publish/subscribe) and “service” (client/server or RPC) names
  - Define message types (payloads)

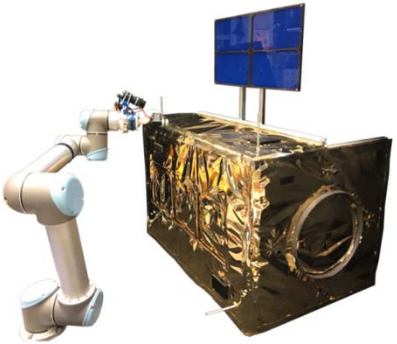
Use Case 1: Teleoperation

- Diverse master and slave devices
- Different communication channels (performance)
- Bilateral teleoperation, force reflection



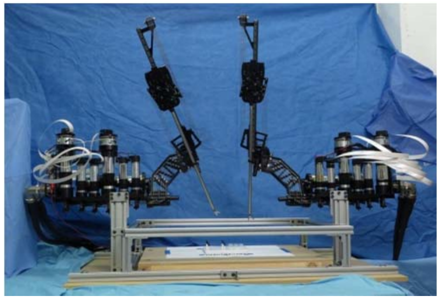
Cartesian position, velocity, incremental position, effort (robot and tool)

→

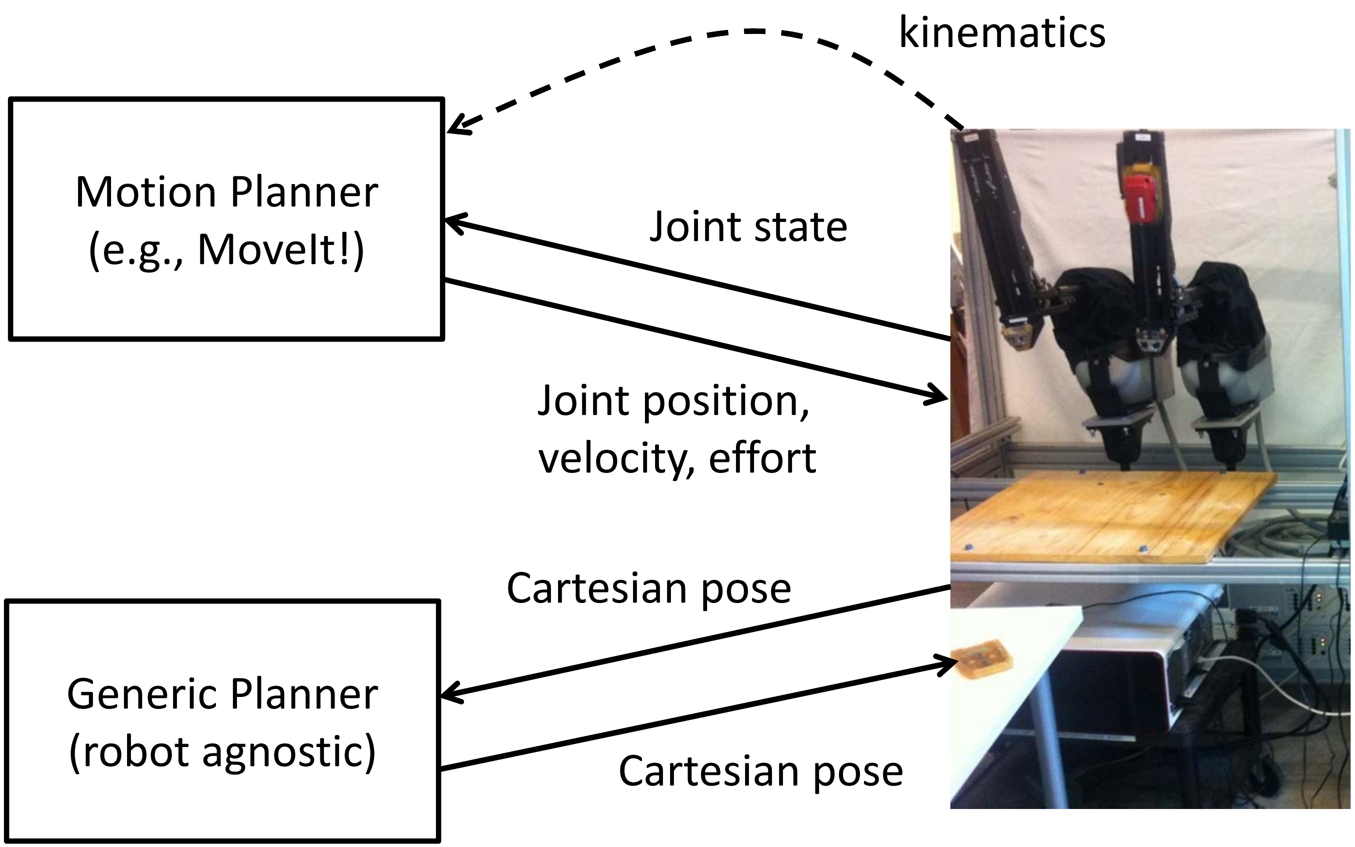


←

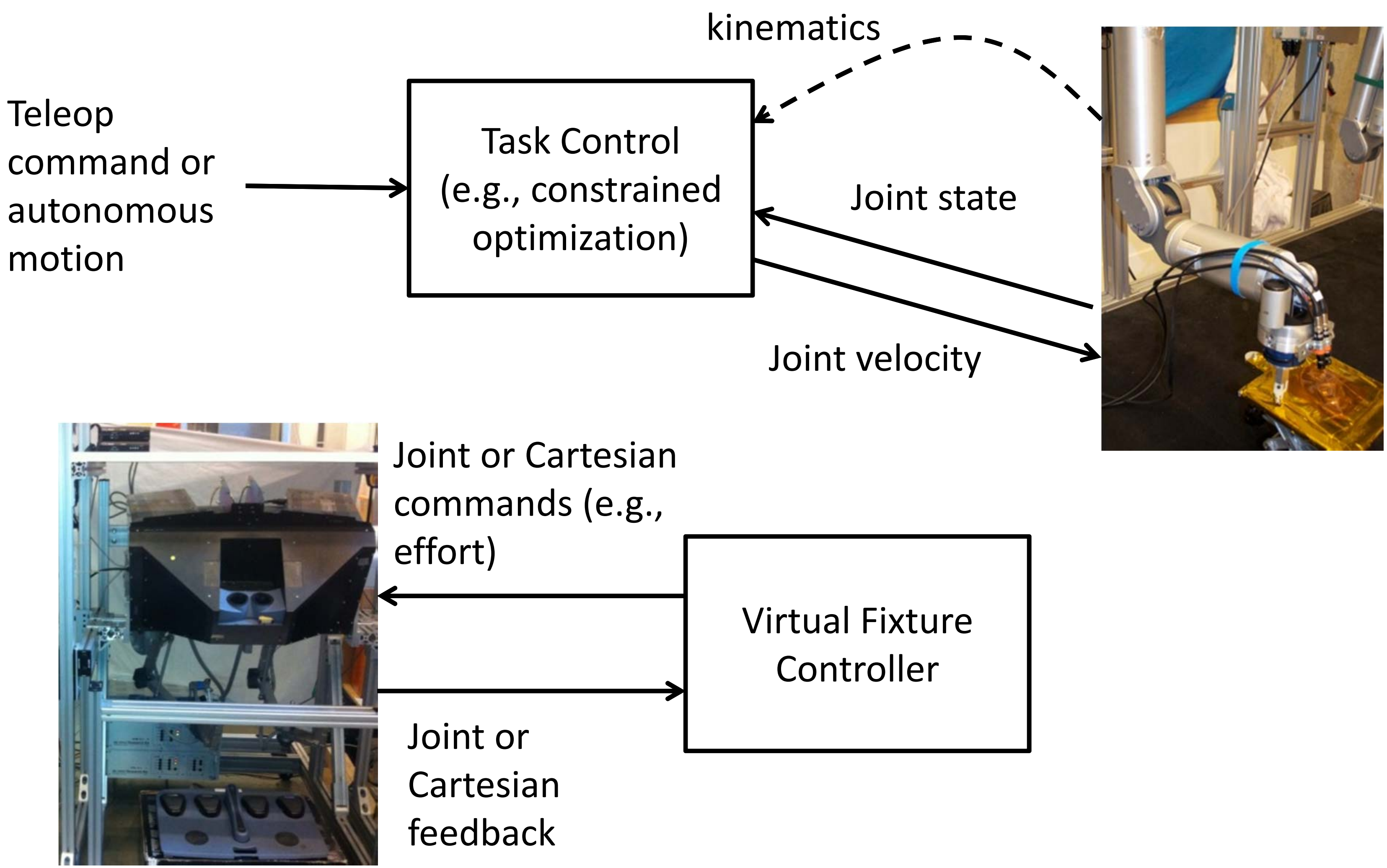
Cartesian state, joint state, generalized forces



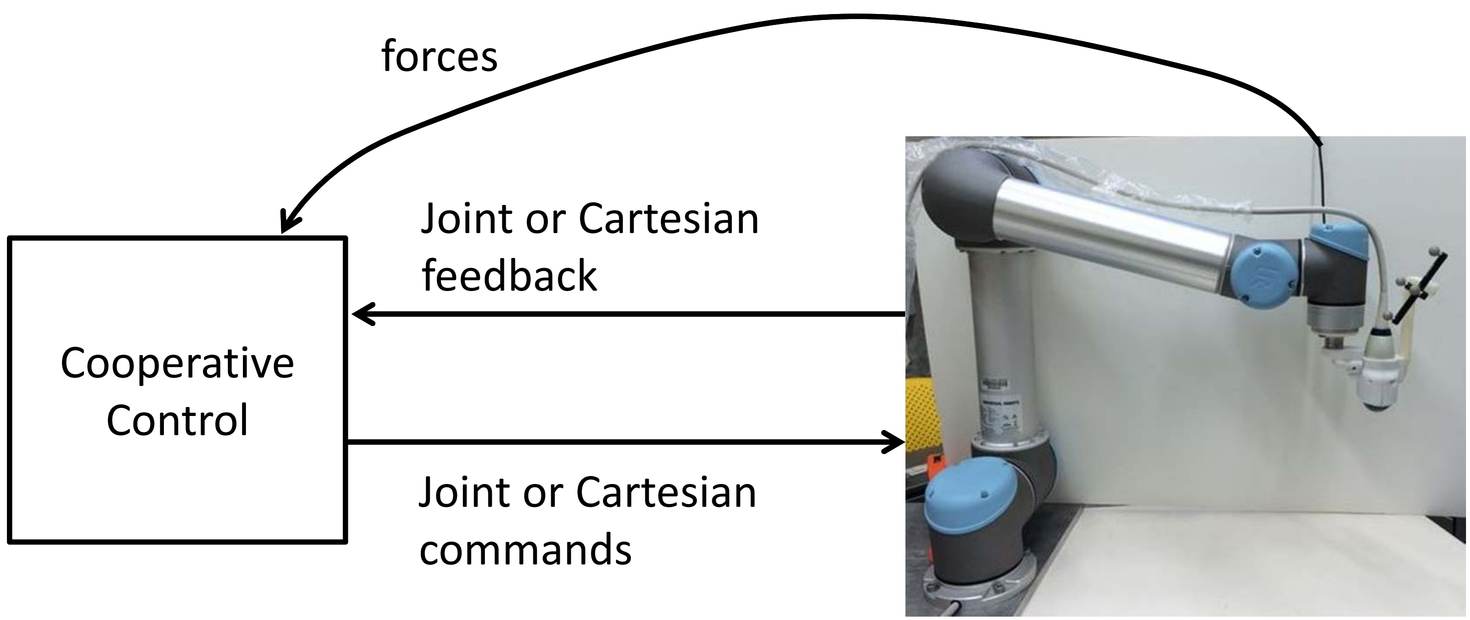
Use Case 2: Autonomous Motion



Use Case 3: Custom Kinematics/Control

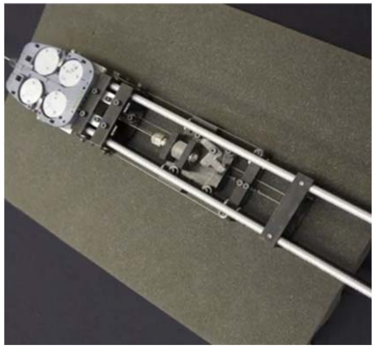


Use Case 4: Cooperative or Compliant Control



Use Case 5: Custom Instruments

- Custom instruments for Raven / dVRK
  - Interface to 4 driving disks



- Powered/sensorized tools/instruments
  - In addition to 4 driving disks
  - Grippers, end-effectors for other robots

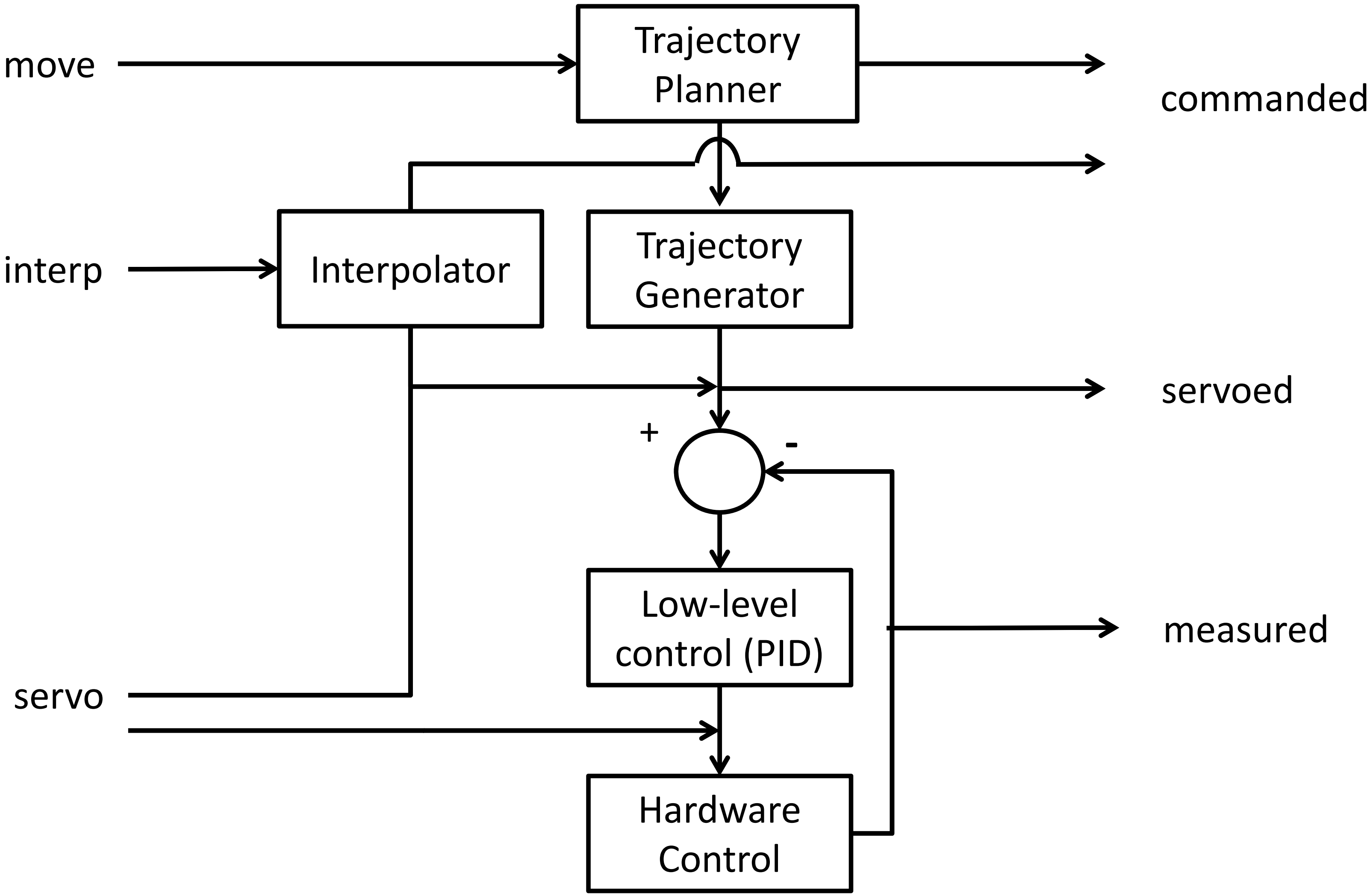
Guiding Principles

- Based on realistic use cases
- As simple as possible (e.g., stream of positions easier than PVT)
- Logical and consistent naming conventions (somewhat like part-numbering convention)
  - All robots not required to implement all commands, but should use consistent name
- Short enough to type into interpreter (e.g., Python, Matlab)
- In most cases, support both publish/subscribe (ROS topics) and client/server (ROS services)

Categories of Commands

- Robot state feedback (Joint/Cartesian position, velocity, effort)
- Robot motion control (Joint/Cartesian motions)
- Robot status feedback and control (e.g., homed, powered on, error)
- Other: configuration, capabilities, …

Feedback and Control



Convention

Control level	• servo: direct real-time stream (pre-emptive) • interp: interpolated stream (pre-emptive) • move: plan trajectory to goal (not pre-emptive? queued?) • (also need to specify multiple goals)
Feedback	• servoed: current setpoint to low-level control • commanded: most recent interp or move goal • measured: sensor feedback • measuredN: redundant sensor feedback (N=2, 3, …)
Space	• j: joint • c: Cartesian
Type	• p: position • i: incremental position • v: velocity; (t: twist) • f: generalized force (e: effort, w: wrench) • s: state (position, velocity, effort) feedback

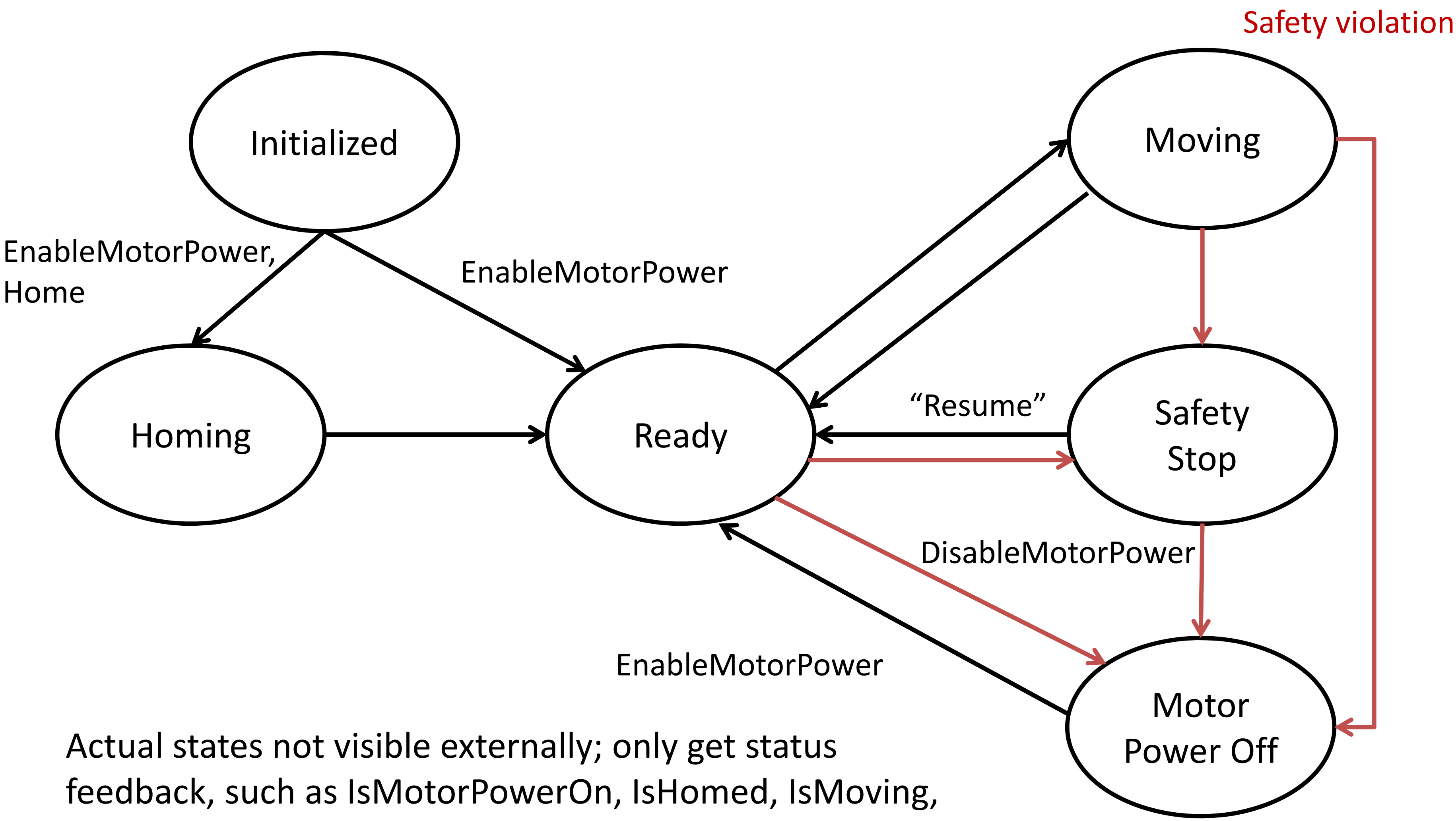
Examples

- move_jp: Plan trajectory and move to joint position
- servo_jv: Real-time update of joint velocity
- interp_ci: Interpolated increment from Cartesian position
- servo_jf: Real-time update of joint force (torque)
- measured_js: measured position, velocity, force
  - ROS JointState payload
- measured_cp: measured Cartesian pose (position)
  - ROS PoseStamped payload

Robot Status

- Not attempting to standardize robot state machine (too much variability)
- Instead, standard commands to change/query status (modes)

Typical Robot Controller State Machine

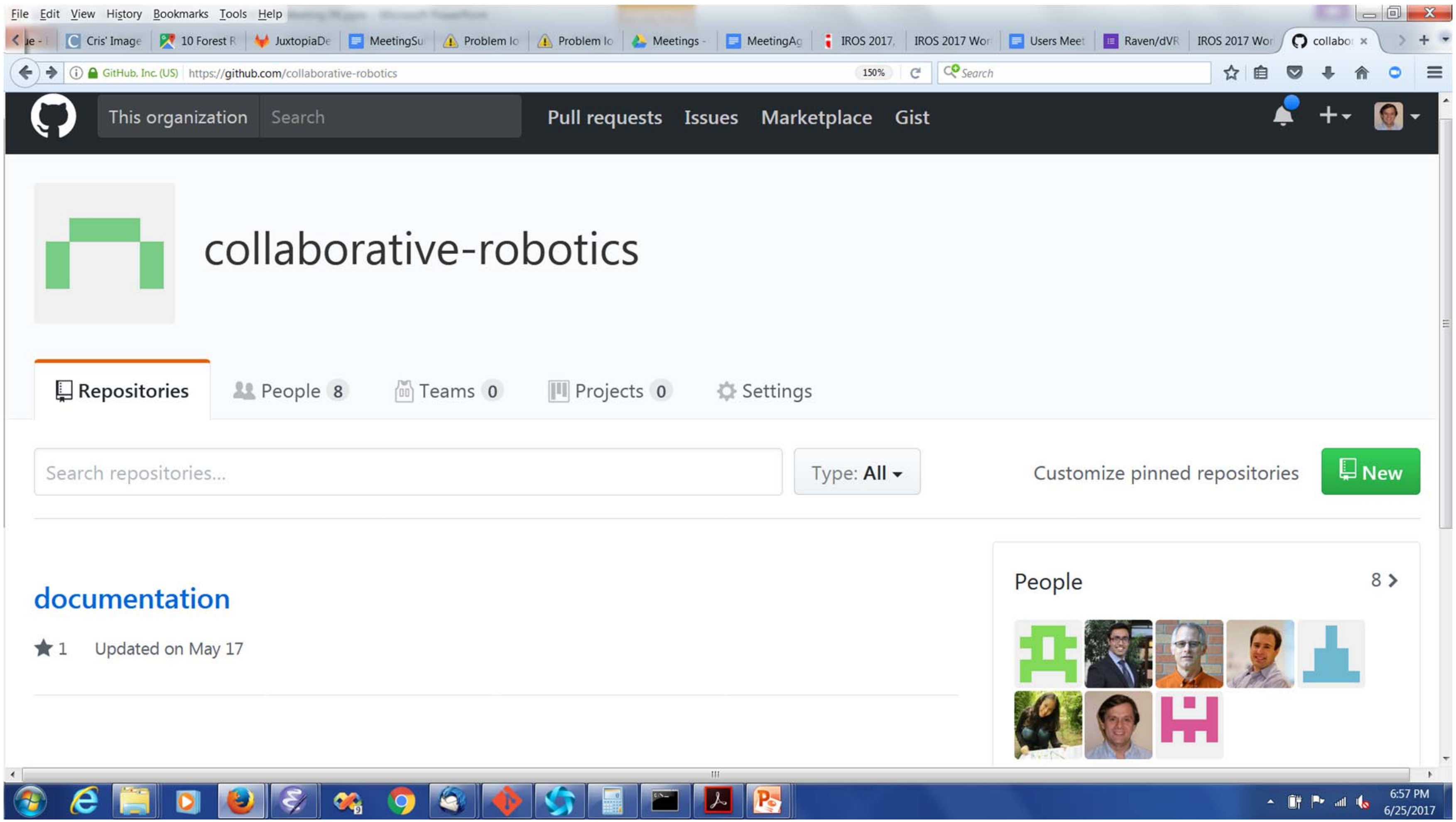


Actual states not visible externally; only get status feedback, such as IsMotorPowerOn, IsHomed, IsMoving, etc. Still need to handle IsToolPresent.

Robot Status Feedback

- Possible status values:
  - IsMotorPowerOn, IsToolPresent, IsHomed, IsReadyForMotion, IsMotionActive, IsProtectiveStop, …
- Can use 1 character (byte) to represent status; e.g., use the ASCII codes for ‘1’, ‘0’, and ‘?’

Looking for Contributions and Feedback



Acknowledgments

- NSF NRI: Software Platform for Research in Semi-Autonomous Teleoperation
  - JHU: IIS 1637789
    - Peter Kazanzides, Russell Taylor, Anton Deguet, Jie Ying Wu, Simon Leonard, Balazs Vagvolgyi
  - WPI: IIS 1637759
    - Greg Fischer, Adnan Munawar
  - UW: IIS 1637444
    - Blake Hannaford, Yangming Li, Melody Su, Kyle Lindgren