Crossing Boundaries: Multi-Level Introspection in a Complex Robotic Architecture for Automatic Performance Improvement

Evan Krause[1], Paul Schermerhorn[2], and Matthias Scheutz[1] | Human-Robot Interaction Laboratory | Tufts University[1] and Indiana University[2]

Abstract

Introspection mechanisms are employed in agent architectures to improve agent performance. However, there is currently no approach to introspection that makes automatic adjustments at multiple levels in the implemented agent system. We introduce our novel multi-level introspection framework that can be used to automatically adjust architectural configurations based on the introspection results at the agent, infrastructure and component level. We demonstrate the utility of such adjustments in a concrete implementation on a robot where the high-level goal of the robot is used to automatically configure the vision system in a way that minimizes resource consumption while improving overall task performance.

Component-Level Introspection Policy

A component has a **set of algorithms** $\mathbf{A = \{a_0, a_1, \ldots, a_n\}}$, where each a_i is capable of independently completing a task.

Each a_i has **tradeoffs between cost, speed, and effectiveness**.

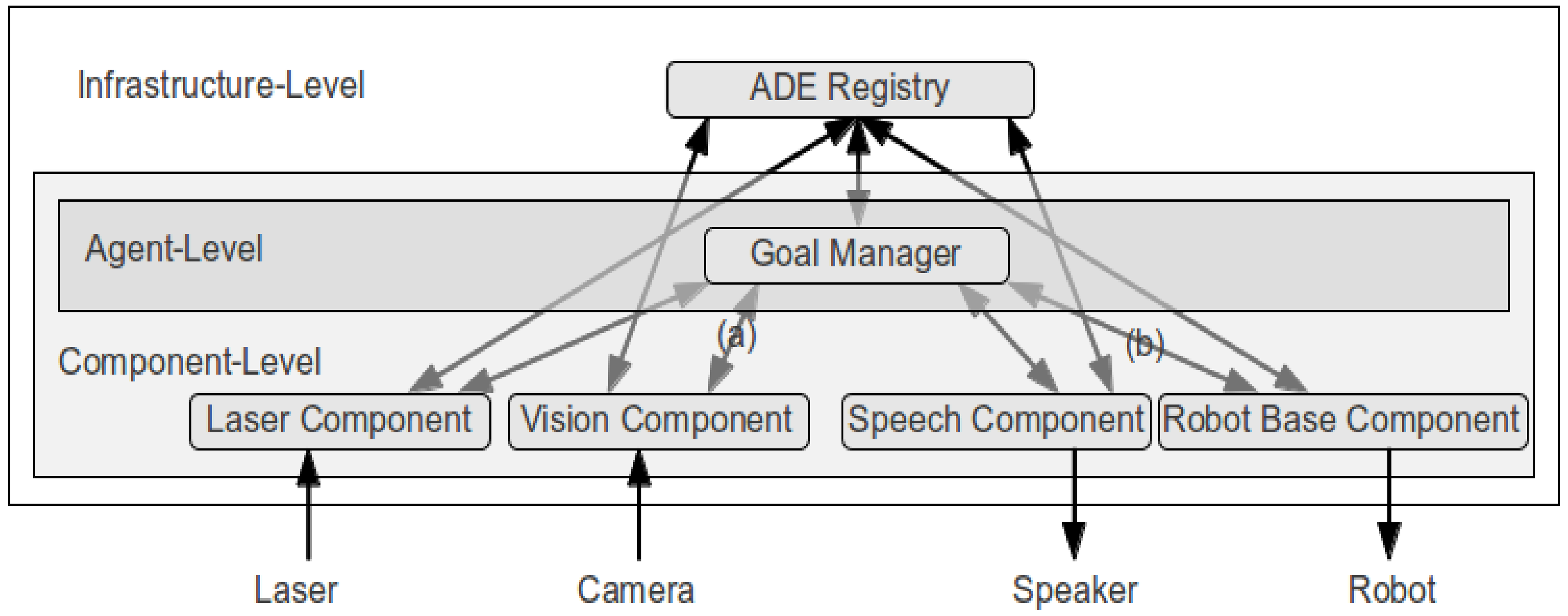
$\mathbf{P_i(s|f)}$ **: the probability of a successful outcome** s for a particular algorithm a_i

where f is a component-specific array of contributing factors, and can include such things as robot velocity, room noise, sensor quality, and/or light levels

$\mathbf{C(a_i)}$ **: the cost** of an algorithm a_i

Policy: **maximize** $\mathbf{P_i(s|f)}$ **and minimize** $\mathbf{C(a_i)}$ to whatever extent possible

Implementation of Introspection Framework in DIARC



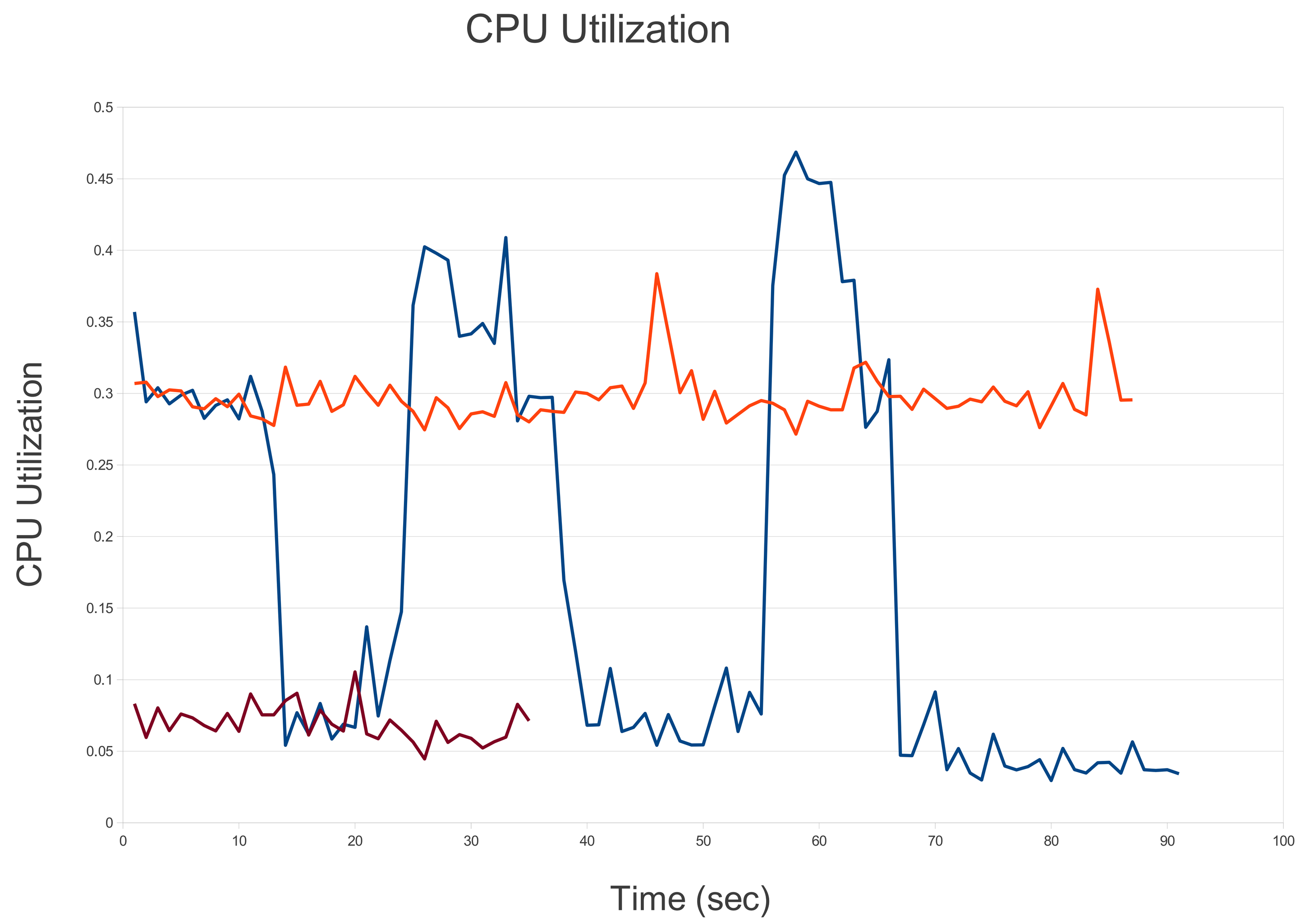
Validation Scenario

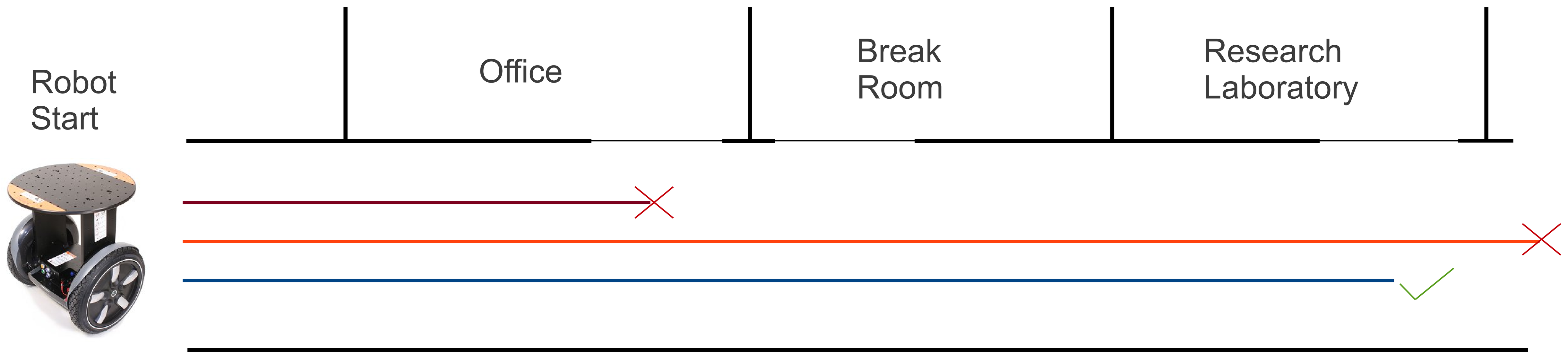
A robot is instructed to move down a corridor to *search for the "Research Laboratory."* The location of the target room is unknown, so the robot must visually identify signs affixed to doors; these signs are all the same shade of blue with the room name in black letters. The task in carried out with three different system configurations.

Configuration 1:
- No component-level introspection
- Blob-only detection
- Stops when the "Office" is incorrectly detected as "Research Laboratory," and never arrives at the target location

Configuration 2:
- No component-level introspection
- SIFT-only detection
- Drives past the target location, failing to detect the "Research Laboratory"

Configuration 3:
- With component-level introspection
- Dynamic switching of detector options
- Successfully detects "Research Laboratory"





Introspection Levels

Component:
Perform highly specialized tasks that may operate independently or in tandem with one or more other components

Component-level introspection monitors the operational conditions necessary for optimal component performance

Infrastructure:
Middleware with knowledge about the underlying operating environment such as available hardware, communication between components, distribution of components across multiple hosts, and resource management

Infrastructure-level reflection is responsible for system health: monitoring the states of system components, restarting or replacing failed components as necessary, migrating components to achieve load balancing [1]

Agent:
High-level knowledge about the world, their own capabilities, and their own goals, and use this knowledge to make decisions and generate actions that lead to the accomplishment of the agent's goals

Agent-level introspection is largely responsible for observing the progress of plans, detecting when plans are acceptable, when they need to be regenerated, or when certain goals are unattainable and need to be abandoned altogether [2]

References

[1] Scheutz, M.; Schermerhorn, P.; Kramer, J.; and Anderson, D. 2007. First steps toward natural human-like HRI. *Autonomous Robots* 22(4):411–423.

[2] Schermerhorn, P., and Scheutz, M. 2010. Using logic to handle conflicts between system, component, and infrastructure goals in complex robotic architectures. In *Proceedings of the 2010 International Conference on Robotics and Automation*.

Acknowledgements

This work was in part funded by ONR MURI grant #N00014-07-1-1049.