"Teach One, Teach All" – The Explosive Combination of Instructible Robots connected via Cyber Systems

Matthias Scheutz
Human-Robot Interaction Laboratory
Department of Computer Science
Tufts University
Medford, MA 02155
Email: matthias.scheutz@tufts.edu

***Abstract*—Combining robotic architectures with cyber systems has enormous potential for future robotic applications because it enables the possibility of online sharing of all aspects of the robotic architecture: the knowledge contained in architectural components, the parameterization of these components, the very component algorithms, as well as the architectural layout. In this paper, we discuss the potential of using cyber systems for knowledge sharing and use among multiple robots. We isolate functional requirements for the robotic middleware to enable such knowledge sharing, briefly discuss our own first steps in developing such a system, and show a proof-of-concept demonstration where two robots share and immediately use a new capability acquired through one-shot learning.**

I. Introduction

Researchers in artificial intelligence and robotics have for quite some time advanced ambitious agendas of developing *data-driven* learning algorithms for robotic systems that allow robots to acquire new knowledge, from perceptual learning (such as object learning or SLAM), to skill learning (such as reinforcement learning of behaviors and learning from instruction), to policy-based learning (e.g., of decision policies, etc.). The tacit assumptions behind these data driven efforts are that (1) much of the knowledge required for a well-functioning robotic system can be learned from available data and does not have to be designed by hand, and (2) as robots are becoming more complex, providing "engineered" knowledge is becoming increasingly difficult, time-consuming, and error-prone.

In addition to data-driven approaches that exploit statistical information in large data sets, recent *knowledge-based* efforts have focused on "one-shot" learning (i.e., the ability to learn something new from one exemplar alone utilizing prior knowledge and contextual information). For example, a robot could learn how to perform a new action or rule from observation, or it could be instructed in natural language how to perform it [1]. The main advantages of one-shot learning over statistical approaches are that no large data sets are required as teaching input, that learning is typically much faster and can often be performed during task execution.

Both data-driven and knowledge-based learning paradigms have almost exclusively focused on learning in individual robots (even when these learning approaches were part of a multi-robot setup). Yet, learning in robots overall could be significantly accelerated and improved if robots had the ability to share *any learned information anytime anywhere*. In this paper, we show how in particular one-shot learning in conjunction with the ability to share learned knowledge can lead to massive parallel robot learning at unprecedented speeds.

We start by proving the motivation for sharing knowledge in the context of robot learning and point to its potential for future robotic systems. Next we discuss the architectural and infrastructure requirements for robotic systems to be able to systematically share and use any type of knowledge at any point in time during their operation. We also briefly introduce our ADE infrastructure which provides the necessary capabilities for online knowledge sharing and use for robotic systems. We then demonstrate in a one-shot learning example with two robots how the ADE infrastructure allows for immediate online knowledge sharing and use. The subsequent discussion section elaborates on the potential of this approach for future robotic applicatoins and the conclusion summarize the ideas of combining learning with knowledge sharing for future robots.

II. Motivation

Ubiquitous access to "the grid" (i.e., online computational resources) paired with the availability of massive and diverse data sets "in the cloud" is increasingly transforming research in science and engineering, allowing for hitherto unimaginable forms of scientific discovery. Not surprisingly, these massively parallel distributed cyber systems that can run powerful data mining and machine learning algorithms on large data sets are becoming a major enabling factor for developing more complex robotic systems. Advanced learning algorithms have been employed on large image and movies databases for learning about objects, scenes and activities, or on large natural language corpora for learning various aspects about natural language (from part-of-speech tagging, to parsing and semantics, concepts, etc.). And recent demonstrations of AI systems such as IBM's Watson only hint at the potential of large data sets combined with the right kinds of learning and search algorithm.

In fact, to the extent that data is or will be available, almost every knowledge-based aspect of a robotic system could be learned from "big data": perceivable objects and their

properties from large image data bases such as Flickr; words, concepts, and facts from large corpora and online linguistic resources such as WordNet, VerbNet, ConceptNet, OpenCyc, and others; actions, activities, and events from online videos (such as YouTube); norms, etiquette, and other aspects of human social exchanges from online fora, blogs, Twitter, etc. And the available data can be analyzed in many different ways, from using classifiers to extract categorical information, to finding relations among data items, higher-level rules for generative processes, and others. Even the different ways of how things could be learned can be learned from online information. Critically, several aspects of the grid-based or cloud-based cyber systems make them perfect resources for robots and thus "extensions" of the robotic architecture:

- *Data storage:* Data does not have the be stored locally on robots, hence it is not subject to the local storage limitations onboard robotic platforms; moreover, the same data can be shared among many robotic agents (rather than replicated locally)
- *Processing resources:* Processing does not have to be performed on the robot's local CPUs, hence is again not subject the onboard processing limitations of robotic systems; rather, processing-intense computations can be shared among many robots
- *Data acquisition:* Data can be acquired and processed in parallel, thus allowing robots to utilize data acquired by other robots instead of having to acquire the data themselves; shared data acquisition thus enables to massively parallel collaborative learning among many agents

Note that in addition to providing data storage, processing resources, and an infrastructure for shared data acquisition, the cyber system addresses critical architectural limitations of robotic architectures: *scalability* (of storage and processing resources) and *availability* (of access to the cyber system, its data and processing resources). As a result, designers of robotic systems that can utilize cyber systems would not have to worry about questions of access control, concurrency, fault-tolerance, networking, and data backup, all of which are handled by the cyber system. For robotic architectures would not need to maintain storage-intensive long-term memories locally, nor would they have to use local computational resources for operating on those memories (e.g., for indexing them, for search and retrieval, etc.). Rather, local computational resources on robots could be solely used for operating the robotic platform (taking in perceptions, performing actions, maintaining short-term memories, managing goals, interacting with humans, etc.). Frequently used long-term items could simply be transferred from the cyber system to the local system as needed (e.g., similar to the caching mechanisms for paged virtual memory systems in computers); all other long-term knowledge items could reside in the cyber system, and many computational expensive processes such as planning, reasoning, problem-solving, etc. could occur in the cyber system as well (often times in parallel).

The advantages of such a setting are manifold, but the two most critical advantages are *knowledge preservation* in the light of robotic failures and *knowledge seeding* for initializing new robots: given that all relevant information will always be stored in a typically distributed fashion in the cyber system, little to no knowledge will be lost when an individual robot breaks; and new robots coming online can download and immediately utilize the knowledge built up by their cohort (assuming that the knowledge is represented in a way that allows a robot to utilize it immediately – we will come back to this point later). The cyber system thus will enable machines to acquire new knowledge and skills at completely unprecedented rates, as every robot that learns anything new at all can potentially contribute this knowledge to the collective knowledge stored in the cyber system. And different from humans who, even if this knowledge is available, cannot immediately use it (e.g., because the knowledge is stored in the form of linguistic descriptions that have to be read and understood first), it is possible to develop mechanisms for a robotic architecture that will remove any functional difference from the robot's operating perspective between knowledge stored and processed locally on the robot and knowledge stored and processed in the cyber system.

III. Background and Related Work

Several recent projects have started to investigate ways of utilizing cyber systems for robotics. For example, the EU-funded RoboEarth project[1] was initiated by a multidisciplinary partnership of robotics researchers from academia and industry with the goals to

- create and execute action recipes
- integrate localization and mapping
- perform 3D sensing
- learning control
- track objects dynamically
- mine data from RoboEarth past data

The project aims to show that "connection to a networked information repository greatly speeds up the learning and adaptation process that allows robotic systems to perform complex tasks" and that "a system connected to such a repository will be capable of autonomously carrying out useful tasks that were not explicitly planned for at design time".

Specific conceptual suggestions have been made for "cloud robotic architectures" [2] that would allow for ad-hoc "cloud networks" among multiple robots with peer-to-peer connections as well as connections between robots and existing cyber systems. The focus, however, was mostly on the challenges involved in the networking infrastructure.

Other work has successfully demonstrated robot learning through written instructions utilizing freely available online information. Nyga and Beetz, for example, demonstrated how a robot could learn to follow recipes written in natural language on wikihow.com. The robot used of a variety of online resources including the WordNet lexical database, the FrameNet action database, the Stanford Parser and wikihow.com, as well as Amazon Mechanical Turk for acquiring labels [3].

The *KeJia* project has also made progress in allowing robots to learn from written natural-language data [4]; when

[1] See http://www.robotearth.org/.

the OK-KeJia robot detects a gap in its knowledge base (whether conceptual, procedural or functional), it attempts to acquire openly available information to fill the gap (e.g., from the OMICS database).

However, there is currently no project that has demonstrated how multiple robots by way of utilizing cyber systems can share and use knowledge, in particular, knowledge acquired through one-shot learning during task execution in the way described here.

IV. Integrating Robotic Architectures with Cyber Systems

For robots to be able to utilize cyber systems in the above envisioned ways, several basic hardware and software provisions are necessary. On the hardware side, network connectivity to the cyber system is required through any type of networking interface (tethered Ethernet or Wifi) that will allow robots to connect to the cyber system (if not all the time, then at least intermittently). On the software side, the operating system running on the robot's computing equipment needs to support networking (including basic IP features such as packet routing, packet forwarding and adhoc networking, e.g., to be able to build and use adhoc networks among multiple robots within wifi capabilities but without WAN connections to be able route packets through the adhoc wifi network to a resource with WAN connectivity).

Given these basic hardware and OS networking capabilities, the critical functional features for online knowledge sharing and shared knowledge use need to be provided by the robotic infrastructures in which the robotic control architectures are implemented [5]. We first briefly review architectural concepts in order to determine what can or cannot be shared among multiple robots, and then discuss the mechanisms needed for the different levels of sharing.

A. Sharing potential in robotic architectures

Robotic architectures, as any agent architecture, consist of two main parts: the *functional components* (consisting of various algorithms for data processing together with parameters of those algorithms) and *their structural organization* ("architecture layout") and the *knowledge* for those components represented in data structures on which the component algorithms can operate. The functional components and architectural layout typically remain the same across multiple platforms, tasks, and missions, while the knowledge for those components are typically task and mission dependent.

The extent to which a robotic architecture can be shared, i.e., both the knowledge contained in the architecture as well as parts of the architecture itself, depends on how the architecture is realized in the implementing robotic middleware. For example, if the architecture is "monolithic" in that all components are implemented in one computational process (as with classical cognitive architectures), then only architectural parameters, but not individual algorithms can be shared, in addition to knowledge contained in the architecture. If, on the other hand, the architecture is "modular" in that at least some components, if not all, are implemented in a distributed fashion in the middleware (as is typical for recent complex robotic architectures), then even component algorithms can be shared if the architecture has a way to handle temporary unavailability of components (as is also typically the case in recent robotic architectures). We thus have the following list of potential knowledge items that could be shared among multiple robots:

- individual components for NLP, SLAM, episodic memory, etc. (i.e., the "algorithms")
- configuration data for those individual components (i.e., the parameterization of the algorithms)
- data stored in individual components (i.e., the knowledge for the architecture such as dictionaries, skills, maps, rules, etc.)
- configuration of groups of components (i.e., the structure of the architecture)

Since we are not restricting sharing to data stored in individual components, but also allow for sharing of architectural components, parameters, and layout, virtually every aspect of the control system is now a resource that has the potential to be shared and used, thus allowing for several intriguing features such as "continuous automatic learning and upgrades".

Continuous automatic learning means that by way of sharing essential architecture storage components such as long-term memories, dictionaries, image and rule databases, and others, each robot will profit from any other robot learning new knowledge, because any such knowledge item will be directly stored and thus contributed to this common knowledge repository. Note that learning can occur in parallel and independent consolidation processes running in the cyber system can remove duplicate knowledge items to streamline the database. Moreover, indexing schemes much like those use by web-based search engines can provide fast data access and proactive data transfer (based on task state and mission context) will often allow robots to "download" knowledge items into their local architectural components before they are needed (e.g., if a robot knows that a map of a building will be required for performing a search and rescue operation, it can download that map if it is available into its SLAM component before entering the building; otherwise, it will build a new map and contribute it to the repository).

Continuous automatic upgrades refers to the continuous improvement of the performance of robotic architectures by way of adjusting parameters, upgrading component algorithms, and altering component layouts (by replacing existing connectivity and/or adding/removing components). For example, it is possible in architectures with distributed components to "upgrade" individual components at run-time when newer algorithms are available on other robots as along as the architecture supports "dynamical component substitution" (i.e., swapping of components while the system is running) [6]. This feature allows robots to automatically stay up-to-date whenever newer algorithms and features become available by way of running architectures of deployed robots, while removing the need for robot operators to manually initiate upgrade processes. Note that even in the monolithic case it might still be possible to "upgrade the architecture" at run-time by way of replacing the whole architecture at once (which may or may not work in a given context, e.g., if the system cannot be shut down during the upgrade).

B. Requirements for knowledge sharing use

The middleware in which the robotic architectures are implemented must provide several features to allow for the above types knowledge sharing among architectures:

- *Cyber system connection:* The prerequisite for all data exchanges between the robotic architecture (RA) and the cyber system (CS) is the ability of the RA to establish, maintain and repair a connection to any type of CS that could serve as a data repository and computing resource (e.g., the "Amazon Cloud Drive" and the "Amazon Elastic Compute Cloud", or simply a set of compute clusters with local disk storage); this includes access control and authentication of the RA in the CS, discovery of available data and computing services offered by the CS for the particular RA, and discovery of other connected RAs and their capabilities (to be able to establish direct connections with other RAs).
- *Data type descriptions:* To be able to share data, *data types* have to be negotiated for all levels of data, including component-based data (e.g., a common map representation for SLAM, common word representations for parsers, common RGB-D representation for point clouds, etc.) as well as architectural components (e.g., algorithms, parameters, and component configurations) for which ultimately an architecture description language is required; moreover, a hardware/software capability specification language is required to formalize the necessary capabilities on the RA's side to be able to consume a particular knowledge item (e.g., what kinds of manipulator and sensory capabilities are necessary for executing a particular kinds of script, what kind of processing is required to be able to consume 3D point clouds, etc.).
- *Data transfers:* Once an RA is connected to the CS and the kinds of data types that can be handled by both sides have been negotiated, data transfer can be initiated by both the RA and the CS (e.g., the RA might send a newly recognized object together with its 3D point cloud to the CS for storage, the CS might send an action plan from an CS-based planner to the robot for execution, etc.); moreover, in addition to single requests for data items or processing, continuously updating data streams might need to be established (e.g., video streams for recording the robot's activities, or force sensor streams for collecting data to improve the robot's manipulation control, etc.).
- *Knowledge search:* The RA can initiate searches for knowledge items pertaining to any aspect of the robotic architecture in the CS, including search for knowledge items to be used in existing components in the RA (e.g., a new executable skill for an action execution component, a new object representation for the vision system, a new production rule for the reasoning system, etc.) as well as components themselves to be used for replacing them (e.g., a new policy-based planning component replacing a deterministic classical planner).
- *Instantiation and processing request:* The RA can utilize the CS computational infrastructure to instantiate and run new architectural components on an "as-needed" basis (e.g., a route planning algorithm in a complex environment might not have to run on the RA, but could be performed in a path planning component running in the CS which can utilize the parallelism of the CS); results obtained from such runs can, in turn, be stored in the CS for future use by other robots (e.g., another robot requesting the same route); this allows for a distributed operation of an RA where parts run on the robot while other parts run in the CS.
- *Online component substitution:* Both the RA and the CS need to be able to replace running components by equivalent or better components without interrupting the operation of both systems; the ability to substitute one component for another component ...

The above mechanisms then allow for different types of knowledge sharing and learning: *active* versus *passive sharing* (where in the active case, an RA sends information directly to another RA, while in the passive case it simply places the information in the cyber system) and also *active* versus *passive learning* (where in the active case, the RA learns something new by using onboard and CS-based resources, while the passive case the RA simply searches for information in the CS.

V. The ADE Middleware

We have developed the "Agent Development Environment" (ADE) [7], [6] which is a fault-tolerant multi-agent system (MAS) [8] that serves as a distributed implementation environment for complex robotic architectures. Analogous to current robotic infrastructures (such as ROS [9] and several others), ADE provides a basic communication and computational infrastructure for parallel distributed processes that implement various functional components of an agent architecture (e.g., the interfaces to a robot's sensors and actuators, the basic navigation and manipulation behaviors, path and task planning, perceptual processing, natural language parsing, reasoning and problem solving, etc.).[2] Different from other robotic infrastructures, ADE was recently extended to also include mechanisms for connecting to cyber systems and for grid computing [10].

In a running ADE system, all participating ADE agents start up independently and connect to each other as prescribed in the architecture diagram to allow for information flow among architectural components. A special agent, the *ADE registry*, implements basic "white and yellow page services" (i.e., specialized "broker agents" that help other agents to find each other) to allow other ADE agents to (1) register and join a distributed ADE system, (2) advertise their services to other components, and (3) connect to other components to be able to use their services. Every ADE system consists of at least one ADE registry and any number of ADE agents with potentially different functionality that must register with a registry on start-up. A multi-robot ADE system, for example, typically has one registry for each robot where each registry is responsible for the ADE agents implementing that robot's architecture.

[2] A detailed conceptual and empirical comparison of robotic infrastructures up to 2006 can be found in [5].

```
Human: Aandy, please pick up the medkit.
Aandy: I do not know how to pick the medkit up.
Human: Cindy, can you please pick up the medkit?
Cindy: I do not know how to pick the medkit up.
Human: Let me show you.
Cindy: Ok.
    [multimodal instruction begins]
Human: Put your hand above the medkit with your palm up like this.
Cindy: OK
Human: Close your hand.
Cindy: OK
Human: Lift your hand.
Cindy: OK
Human: And that's how you pick it up.
Cindy: OK
    [end of instruction]
Human: Please tell Aandy how to pick up the medkit.
Cindy: Certainly.  I am doing that right now.
    [Cindy transfer the newly learned action to Aandy]
Cindy: Transfer complete.
Human: Aandy, now can you do it?
Aandy: OK.
    [Aandy received the script and can now perform the pickup action]
```

Fig. 1. A demonstration interaction between a human and two robots where the human teaches the robot an new activity which one robot (Cindy) learns in "one shot" and shares directly with another robot (Aandy) via the ADE middleware. A video of the interaction can be found at http://hrilab.tufts.edu/movies/teachoneteachall.mov.

The registries are all connected to each other which not only provides mechanisms for fault-tolerant computing,[3] but also a means for enabling agent discovery and agent-to-agent communication across multiple robotic architectures. In addition, specialized ADE agents such as the "ADE grid manager" and the "ADE grid engine" are provided for interacting with cyber systems to be able to schedule, distribute, start, and monitor the execution of distributed computational processes (such as simulations or planning processes) [10]. These grid components are started via their own registries in cyber systems (e.g., a compute cluster or a cloud computing infrastructure) and allow robots to submit expensive computations that can be performed in massively parallel computing environments offline. The results are then sent back to the robots for onboard online use. In addition, special database agents are provided to provide an interface to standard database systems and allow for efficient storage and access of computational results from grid computations. These databases can also serve as a common repository for storing knowledge items that robots want to share via the cyber system.

VI. Demonstration Scenario

To demonstrate the existing capabilities in ADE for enabling the connection between robotic architectures and cyber systems and how this connection can be used for sharing newly learned knowledge, we will briefly discuss a proof-of-concept example where one robot ("Cindy") is taught a new skill, asked to share with another robot ("Aandy"), and the other robot is subsequently asked to perform the new skill. Specifically, we will consider a human-robot interaction scenario where a human instructor is teaching a robot the activity of "picking up a medical kit" using a mixture of natural language and physical demonstration. We assume that neither robot knows in general how to pick up any object with a handle, and that, in particular, neither robot knows how to pick up the medical kit (even though it already knows what a medical kit is). The following multi-modal multi-robot dialogue interaction is a proto-typical example of how a robot could (1) be taught to pick up medical kit that is placed in front of it on a table, (2) assemble a "pick-up-medkit" script from this one interaction ("one-shot learning"), (3) then immediately share the script with another robot, which, in turn, can execute it right away when asked. Both robots are in the example are running our DIARC cognitive robotic architecture which is implemented in the ADE middleware.

Figure 1 shows the interaction between the human instructor and the two robots. Note that for the Cindy robot to be able to learn to move one of its hands into the handle with its palm up in a way that will allow it to subsequently perform the action on different types of objects placed in different ways in different positions, the robot needs to abstract over the exact position of the handle on the particular medkit and generate a representation that is general enough to capture the final position of the hand inside the medkit without being too specific about the particular handle and the particular position of the instructor's hand.[4]

[3] Since all registries share essential system information, any registry can take over if another registry fails for whatever reason (e.g., a hardware fault on the computer the registry is running on causing all ADE agents located on that machine to fail) by noticing the fault and automatically restoring the failed subsystem, if possible at all.

[4] The learned "action script" generated by the Cindy robot (which is then shared with the Aandy robot) captures the relevant aspects of the learned activity and can be executed by both robots (the details of how the robot can generate the script are not the focus of this paper and thus described elsewhere [11]).

VII. Discussion

It seems clear from everything said so far that once robotic architectures are connected to and integrated with cyber systems, unprecedented massive online learning is possible where robot can share and improve virtually every aspect of their system. To just get a sense of the amount of such learning, suppose that 10% of the currently 20 million service robots could connect to a cyber system and exchange learned information in the way demonstrated above. Furthermore, suppose each individual robot is on average taught one new knowledge item per day and collectively there are, again on average, 1000 distinct knowledge items in the whole population (i.e., there is a 95% overhead in teaching the same knowledge on any given day). In that case, sharing the information will give each robot access to all 1000 items on day one, and potentially access to another 1000 items on day two, and so forth, thus speeding up the knowledge acquisition by three orders of magnitude. Moreover, in many cases, robots will not have to be taught anything because all necessary information is already available in the cloud. This will come in handy, for example, in the case of factory robots where skilled human workers can teach multiple robots different activities at the same time and all robots in the factory (whether they were taught or not) will know them and know how to use them right away. Another example might be the cohort of household robots where once one robot has found a solution for how to stack a particular type of dishwasher with different types of plates effectively, all other robots will know how to do it without any human instruction or supervision. And very much like current computers first and foremost connect to their manufacturers web site to download the latest software when they boot up for the first time, future robots could connect the cyber system to retrieve configure their architecture and obtain all the knowledge required to perform their tasks.[5]

There are other capabilities, in addition to to sharing knowledge, that are enabled by connecting robotic architectures with cyber systems. For example, robots could share sensory information via streaming data (e.g., one robot could use data streaming from another robot's camera to augment its own vision system or to monitor the state of the environment in the other robot's location; multiple robots could share the same speech recognizer running on the robot co-located with the operator, etc.).

VIII. Conclusion

We have argued that combining robotic architectures with cyber systems has enormous potential for all kinds of future robotic applications. We specifically focused on the possibility of online sharing all aspects of the robotic architecture, including the knowledge contained in components, the parameterization of the components, component algorithms, as well as the architectural layout. We discussed the computational mechanisms required of implementation infrastructure of robotic architecture to enable interactions with cyber systems and briefly described a first attempt at implementing such capabilities in the ADE infrastructure. We also provided a proof-of-concept demonstration showing that the mechanisms available in the ADE system allow robots to immediately shared newly learned information. We then briefly discussed the potential knowledge sharing in industrial and service settings and also pointed to other functional capabilities enabled by connecting robotic architectures with cyber systems. Finally, it is worth mentioning that aside from the technical challenges that still remain to be solved to enable truly general knowledge sharing among robots there are also important questions about the human perception and acceptance of such "super-human" capability of future robots, especially when these robots have to work with humans in teams (we recently completed the first such evaluation study [12]).

[5]Exactly how this system will then be set up will depend on various factors, including financial and market-specific factors. E.g., one could envision an "app store" for robots where robots can shop for skills that they then can purchase, see http://www.robotappstore.com/, or one could imagine a separate cyber system for each robot cohort produced by a particular manufacturer.

IX. Acknowledgments

This project was in part funded by ONR grants N00014-11-1-0289 and N00014-14-1-0149. Thanks to Evan Krause, Gordon Briggs, Thomas Williams, Cody Canning, Matthew Dunlap and Jeremiah Via for their work on the action learning demo scenario.

References

[1] M. Scheutz, R. Cantrell, and P. Schermerhorn, "Toward humanlike task-based dialogue processing for human robot interaction," *AI Magazine*, vol. 32, no. 4, pp. 77–84, 2011.

[2] G. Hu, W. P. Tay, and Y. Wen, "Cloud robotics: Architecture, challenges and applications," *IEEE Network*, pp. 21–27, May/June 2012.

[3] D. Nyga and M. Beetz, "Everything robots always wanted to know about housework (but were afraid to ask)," in *Proceedings of IROS*, 2012, pp. 7–12.

[4] X. Chen, J. Xie, J. Ji, and Z. Su, "Toward open knowledge enabling for human-robot interaction," *Journal of Human-Robot Interaction*, vol. 1, no. 2, pp. 100–11, 2012.

[5] J. Kramer and M. Scheutz, "Robotic development environments for autonomous mobile robots: A survey," *Autonomous Robots*, vol. 22, no. 2, pp. 101–132, 2007.

[6] M. Scheutz, "ADE - steps towards a distributed development and runtime environment for complex roboti c agent architectures," *Applied Artificial Intelligence*, vol. 20, no. 4-5, 2006.

[7] P. Schermerhorn and M. Scheutz, "Natural language interactions in distributed networks of smart devices," *International Journal of Semantic Computing*, vol. 2, no. 4, pp. 503–524, 2008.

[8] F. Bellifemine, A. Poggi, and G. Rimassa, "Jade - a fipa-compliant agent framework," in *Proceedings of the 4th International Conference and Exhibition on the Practical Application of Intelligent Agents and Multi-Agents*, London, April 1999, pp. 97–108.

[9] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *Proceedings of ICRA Workshop on Open Source Software*, 2009.

[10] M. Scheutz and J. Harris, "An overview of the simworld agent-based grid experimentation system," in *Large-Scale Computing Techniques for Complex System Simulations*, D. Werner, K. Kurowski, and B. Schott, Eds. Wiley, 2011.

[11] M. Scheutz, E. Krause, and G. Briggs, "Towards one-shot activity learning from a mixture of natural language dialogues and demonstrations."

[12] T. Williams, P. Briggs, N. Pelz, and M. Scheutz, "Is robot telepathy acceptable? Investigating effects of nonverbal robot-robot communication on human-robot interaction."