Special Issue: Robotics Collaborative Technology Alliance (RCTA) Program

Regular Article

An Intelligence Architecture for Grounded Language Communication with Field Robots

Thomas M. Howard[1], Ethan Stump[2], Jonathan Fink[2], Jacob Arkin[1], Rohan Paul[3], Daehyung Park[3], Subhro Roy[3], Daniel Barber[4], Rhyse Bendell[4], Karl Schmeckpeper[5], Junjiao Tian[6], Jean Oh[6], Maggie Wigness[2], Long Quang[2], Brandon Rothrock[7], Jeremy Nash[7], Matthew R. Walter[8], Florian Jentsch[4] and Nicholas Roy[3]

[1]University of Rochester
[2]Army Research Laboratory
[3]Massachusetts Institute of Technology
[4]University of Central Florida
[5]University of Pennsylvania
[6]Carnegie Mellon University
[7]Jet Propulsion Laboratory, California Institute of Technology
[8]Toyota Technological Institute at Chicago

Abstract: For humans and robots to collaborate effectively as teammates in unstructured environments, robots must be able to construct semantically rich models of the environment, communicate efficiently with teammates, and perform sequences of tasks robustly with minimal human intervention, as direct human guidance may be infrequent and/or intermittent. Contemporary architectures for human-robot interaction often rely on engineered human-interface devices or structured languages that require extensive prior training and inherently limit the kinds of information that humans and robots can communicate. Natural language, particularly when situated with a visual representation of the robot's environment, allows humans and robots to exchange information about abstract goals, specific actions, and/or properties of the environment quickly and effectively. In addition, it serves as a mechanism to resolve inconsistencies in the mental models of the environment across the human-robot team. This article details a novel intelligence architecture that exploits a centralized representation of the environment to perform complex tasks in unstructured environments. The centralized environment model is informed by a visual perception pipeline, declarative knowledge, deliberate interactive estimation, and a multimodal interface. The language pipeline also exploits proactive symbol grounding to resolve uncertainty in ambiguous statements through inverse semantics. A series of experiments on three different, unmanned ground vehicles demonstrates the utility of this architecture through its robust ability to perform language-guided

Received: 02 November 2020; revised: 10 April 2021; accepted: 10 May 2021; published: 30 March 2022.

Correspondence: Thomas M. Howard, University of Rochester, Email: thoward@ece.rochester.edu

spatial navigation, mobile manipulation, and bidirectional communication with human operators. Experimental results give examples of component-level behaviors and overall system performance that guide a discussion on observed performance and opportunities for future innovation.

Keywords: robot teaming, human robot interaction, cooperative robots

1. Introduction

The promise of field robots to automate dangerous, dirty, difficult, and/or dull tasks in semi-structured and unstructured environments has been evident for several decades (Krotkov and Blitch, 1999). Examples include automating underground mine operations (Scheding et al., 1999, 1997; Roberts et al., 2000; Marshall et al., 2008; Duff et al., 2003), material handling (Durrant-Whyte, 1996; Walter et al., 2015), underwater and ocean science (Singh et al., 2004; Johnson-Roberson et al., 2010; Williams et al., 2012; Yoerger et al., 2007; Bowen et al., 2008; Camilli et al., 2010; German et al., 2008), and space exploration (Maimone et al., 2007; Furgale and Barfoot, 2010; Arvidson et al., 2010). The 2004 and 2005 DARPA Grand Challenges (Thrun et al., 2006; Urmson et al., 2006) and the 2007 Urban Challenge (Urmson et al., 2008; Bacha et al., 2008; Miller et al., 2008; Montemerlo et al., 2008; Bohren et al., 2008; Leonard et al., 2008) accelerated capabilities in autonomous navigation on rough terrain and in urban environments, respectively. Collectively, these efforts have led to significant progress in planning, perception, state estimation, and control in the face of uncertainty inherent in these environments.

Despite these advances, and with few exceptions (Walter et al., 2015), most systems do not consider the challenges of operating field robots that interact and operate alongside humans. The way field robots accept input and provide feedback has typically been restricted to specifically engineered, human-interface devices for direct or supervised teleoperation, providing goals and returning sensor observations and synthesized representations of the environment. As field robots become more capable, the scope of information that must pass between operators and robots in multi-robot human-robot teams increases dramatically. Human teams use language in audial or visual forms to express diverse concepts compactly at dramatically different scales. For example, a human-robot team performing urban search-and-rescue might involve high-level goals like "search for survivors in the building on the corner of the park," specific instructions such as "clear the door of debris," or requests for information like "how many people were on the ground floor of that building?" To interpret these instructions, the robot must be able to understand spatial concepts such as "corner of the park" and "ground floor," in addition to executing such procedures as "search" and "clear," and responding to queries for information collected from sensor observations using natural language. Recent advances in natural language understanding, which interprets the meaning of operator utterances in the context of the perceived environment, has enabled mobile robots (Barber et al., 2016), manipulators (Broad et al., 2017), and mobile manipulators (Patki et al., 2020) to follow human-provided instructions using text-based interfaces or automatic speech-recognition tools. Each of these applications uses a symbolic representation of actions that the robot can execute and concepts that it can represent, both tailored for their specific domains. In contrast to our architecture, these systems only accept instructions and do not provide linguistic feedback to the operator. Recent work on bidirectional communication shows that language-understanding models capable of reasoning about their observed environment can be inverted to ask for clarifications (Knepper et al., 2015) and express discrepancies in facts provided by the human operator (Arkin et al., 2020).

In this article we present a novel robot intelligence architecture for bidirectional, grounded language communication with autonomous field robots. The proposed framework enables the robot to construct a symbolic representation of the activities an operator wants it to perform. This architecture converts those symbols both to and from language in the context of the robot's environment and constructs behavior trees of actions from those symbols with robot-specific implementations of said actions to enable scalable and portable intelligence architectures that are not bespoke for individual platforms. Descriptions are provided for modules that implement semantic

perception, environment modeling, language understanding and generation, mission planning and execution, navigation, manipulation, and scene description. Experimental results on three different field robots demonstrate how this intelligence architecture enables human-specified tasks that require understanding of spatial concepts, declared knowledge, disambiguation, and a diversity of different actions that the robot may be required to perform. These experiments illustrate how different language inputs and perceived environments can result in unique activation patterns through the architecture and also that these routes exploit many shared components and a centralized world model. This architecture represents the evolution of an intelligence architecture developed under the Army Research Laboratory's Robotics Collaborative Technology Alliance (RCTA) for an activity involving human-robot execution of complex missions. This research program, spanning more than a decade, supported the goal of creating manned/unmanned teaming by transitioning robots from tools to teammates through the development of technologies that enable bidirectional exchange of information between human and robot teammates in human-understandable terms.

2. Related Work

While there are a large number of field robots that allow operators to perform complex tasks in semi-structured and unstructured environments, the large majority operate either as entirely or partially autonomous agents (Scheding et al., 1999, 1997; Roberts et al., 2000; Marshall et al., 2008; Duff et al., 2003; Durrant-Whyte, 1996; Singh et al., 2004; Johnson-Roberson et al., 2010; Williams et al., 2012; Yoerger et al., 2007; Bowen et al., 2008; Camilli et al., 2010; German et al., 2008; Maimone et al., 2007; Arvidson et al., 2010; Buehler et al., 2009) or under full teleoperation (Kang et al., 2003; Ryu et al., 2004; Yamauchi, 2004; Fong et al., 2003; Keskinpala et al., 2003; Ballard, 1993; Fong and Thorpe, 2001). However, few field robots are capable of operating in the continuum between full autonomy and full teleoperation, which requires an intelligence architecture capable of reasoning over abstract directives provided by the operator. One exception is the Agile Robotics for Logistics project (Walter et al., 2015) that developed a voice-controllable forklift capable of loading and unloading trucks and other material handling tasks in dynamic, minimally prepared environments, including human-occupied outdoor warehouses typical of disaster relief and military forward operating bases. The system allows operators to issue task-level commands using a combination of speech commands (e.g., picking up a named object) and stylus-based gestures (e.g., placing cargo at a user-annotated location in a bird's-eye rendering of the robot's surroundings). The architecture uses a combination of monocular cameras and planar LIDARs to maintain a model of the robot's environment that expresses the name and pose of relevant objects (e.g., palletized cargo, trucks, and people) and locations (e.g., storage bays). Unlike our approach, which utilizes class-level object detection, their architecture performs instance-level recognition, which restricts the world model to a small set of objects. In turn, this restriction limits the diversity of the language commands that the system is able to reason over. Additionally, while our architecture is able to generate natural language that conveys a rich amount of knowledge to human teammates, their framework is limited to a small set of utterances that are hard-coded to convey the robot's next task and current autonomy state. Three additional works involving human-robot interaction for field robots include the systems described in Perzanowski et al. (2001), Ryu et al. (2004), and Heikkilä et al. (2012). Perzanowski et al. (2001) proposed a multimodal interface to ground robots that allows users to issue navigation-related commands through pen-based gestures and a limited set of spoken utterances. Ryu et al. (2004) described a multimodal teleoperation interface to a mobile manipulator designed for field operations. The interface supports a limited set of speech commands that trigger mode changes (e.g., switching from manipulation to navigation) and uses synthesized speech along with visualizations to convey a pre-defined set of information back to the operator. Heikkilä et al. (2012) described a system that allows people to command a mobile manipulator designed for space operations to manipulate objects using simple spoken utterances. The proposed framework described here differs from these three systems in that it is capable of bidirectional human-robot interaction of high-level goals and low-level actions, and additionally capable of resolving observed ambiguities through a dialogue

system capable of inferring unique descriptions of objects that cannot be uniquely resolved. The proposed architecture is a significant expansion of the one proposed in previous work (Boularias et al., 2015; Oh et al., 2015, 2017), which enables grounded language interaction with a field robot through a multimodal interface. In contrast to this work, our architecture is able to communicate bidirectionally through language, resolve ambiguous instructions, proactively ground symbols for more efficient probabilistic inference, deliberatively interact with objects, and perform a diverse set of behaviors across a variety of field robots, each with different capabilities.

3. Intelligence Architecture

Our approach to an intelligence architecture for collaborative human-robot teams is illustrated in Figure 1. The architecture is composed of a number of modules that generally fall into three categories: perception and environment modeling, grounded language communication, and mission planning and execution. These modules are connected using ROS message passing to exchange information in the form of text, objects, observations, actions, and/or commands to perform sequences of behaviors corresponding to human-specified tasks. At the center of this architecture is an intelligence world model that contains a store of objects, their semantic properties, and their metric poses informed by sensor observations and language interaction with a human operator. This model is used at multiple layers of the architecture to provide a common representation for the robot to reason over when interpreting instructions, generating descriptions, and executing missions. Communication between the human and the robot is done through a multimodal interface. The multimodal interface is a human-held device that accepts and displays text describing observed environments from the robot's perspective, which originate from the scene description module, poses questions and shows facts from the natural language generation module via the intelligence world model, and illustrates a top-down visual perspective of the world. The natural language pipeline can be visualized by the red modules surrounding the intelligence world model. The natural language understanding module takes phrases and symbols from the parsing and declarative knowledge module. This module then uses these phrases and symbols with the context of the robot's observed environment and any proactively grounded symbols to generate a distribution of symbols representing the grounded meaning of the statement. When symbols indicate that a referenced object is ambiguous, these symbols are sent to the natural language generation module that inverts the language model for unique descriptions of the observed instances of the referred to semantic object type to pose a disambiguation query to the human operator through the multimodal interface. The natural language model maintains the content of previously grounded statements so that when a statement resolving the ambiguous query arrives, it can be incorporated with the previous ambiguous instruction to generate an unambiguous action. These resolved actions, just as more direct statements that do not require disambiguation, are passed as actions, modes, and constraints to a mission planner that generates a behavior tree that sequences actions the robot must perform to complete the described behavior or sequence of behaviors. These actions are routed to one of four different modules in the intelligence architecture responsible for performing these actions, and their states are tracked to quantify progress in the inferred mission. The first of these modules, the navigation planner, resolves motion planning queries that require the robot to drive from one pose to another pose in a partially observed environment and additionally handles more nuanced modes of locomotion that consider traits like "stealthiness". The second module, the manipulation planner, handles base locomotion, grasp planning, and manipulation planning for mobile manipulators that use the presented intelligence architecture. The third module, deliberative interactive estimation, handles actions that require the robot to physically interact with an object to measure and report a semantic property such as "full" or "empty" that would be fused with visual observations to better inform the state of objects tracked by the intelligence world model. The last module, scene description, uses a learned image-captioning model to generate detailed descriptions of the current scene observed by the robot. In this section, we will expand the description of the role of each of these modules, describe an implementation of each module and situate it in the

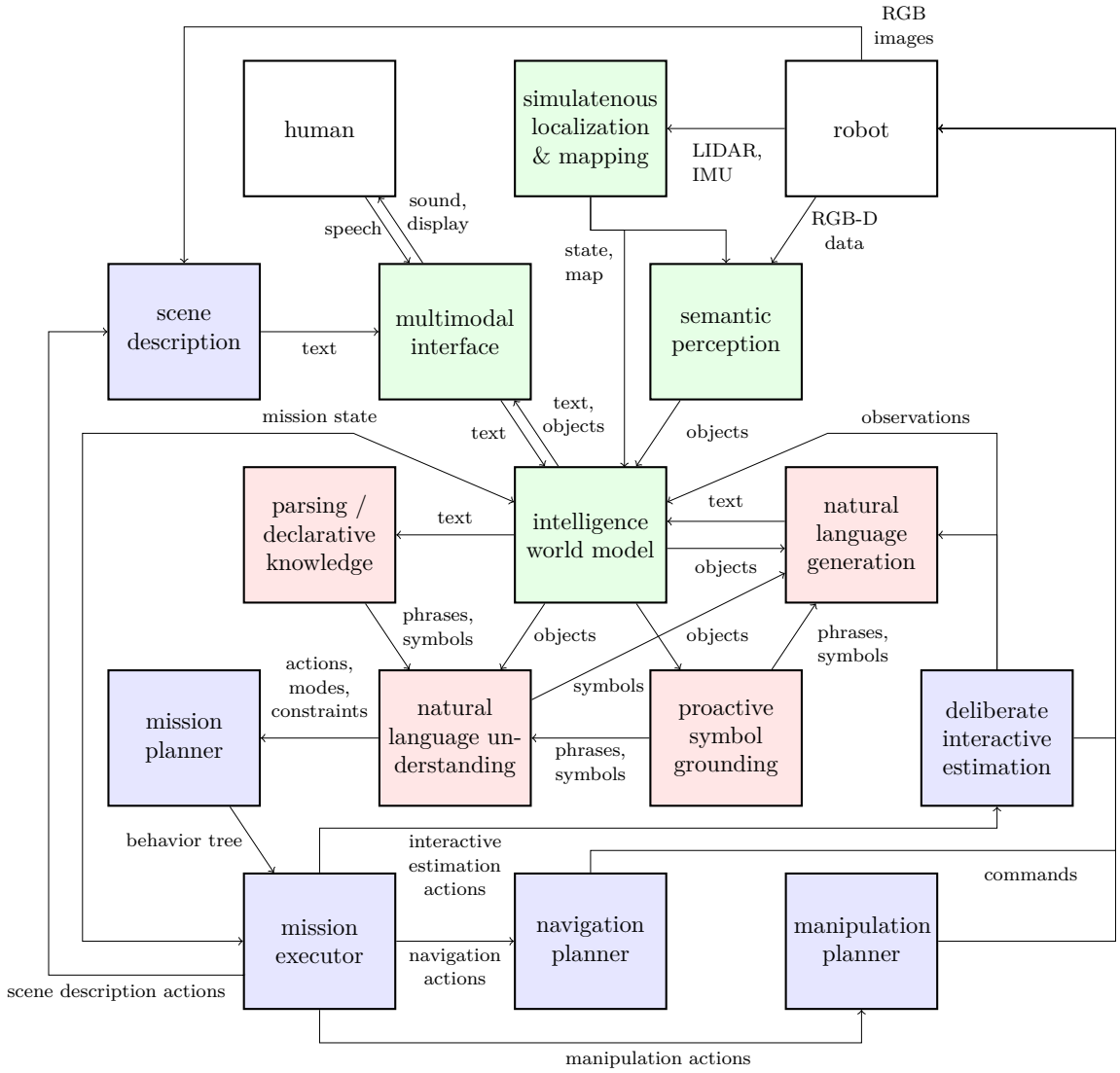


Figure 1. An intelligence architecture for collaborative human-robot teaming. The diagram illustrates how information travels between the human and the robot using a number of different modules representing component technologies in the architecture. Modules for simultaneous localization and mapping, semantic perception, the intelligence world model, and the human-robot interface are shown in green. Modules for natural language understanding and generation are illustrated in red. Modules for mission planning and execution are shown in blue. Some modules implemented in this system, such as the metric world model used by the navigation planner, are omitted for clarity.

context of the contemporary literature, and expand on unique features or capabilities brought forth by the presented implementation. We introduce each component of the architecture and how they interact to perform missions and tasks guided by the human in the context of the robot's observed environment. Note that *missions* and *tasks* in this paper describe high-level activities from human guidance while *actions* and *behaviors* describe components of a *mission* or *task* performed by the modules for navigation planning, manipulation planning, deliberative interactive estimation, and scene description. Since the human guidance may only require one *action* or *behavior* to perform a simple *mission* or *task*, human instructions such as "drive to the cone" could be described as a

human-directed *mission* or *task* involving navigation to an object or the single *action* or *behavior* needed for completion of the mission by the navigation planning module.

3.1. Modules for Perception and Environment Modeling

The first group of modules in the architecture – comprising the multimodal interface, semantic perception, simultaneous localization and mapping, and the intelligence world model – handle inputs and observations of humans and the environment to construct a semantic and metric model of the world that is appropriately but not overly detailed for grounded language communication, mission planning, and motion planning. The multimodal interface also provides a channel for the robot to directly interact with human operators by presenting text-based descriptions of the environment from the scene description module, disambiguation requests generated by the natural language understanding and generation modules, and text-based descriptions of observed facts that are inconsistent with facts informed through the deliberative interactive estimation module.

3.1.1. Multimodal Interface

To effectively interact with human teammates, robots must be able to receive guidance, necessary knowledge, and commands from humans and provide information back to the human. This bidirectional interaction requires methods that display information to the human teammate and that acquire inputs from them. Traditional approaches that use specifically engineered input devices or programmed touchscreen devices to allow human operators to provide guidance and receive information are subject to constraints imposed by the engineer or programmer before the system was fielded. Alternative approaches that use language are inherently more flexible but fraught with difficulties of automatic speech-recognition (ASR), parsing, language understanding, dialogue management, language generation, and text-to-speech (TTS). This architecture blends both of these modes of communication in a device described as the multimodal interface (MMI). The MMI enables the operator to communicate with a robot through speech and gestures, and receive auditory, visual, and tactile responses from the robot through a hand-held touchscreen device with a headset for ASR and sound, a gesture recognition glove for interpreting operator gestures, and a C-2 Tactor Belt for tactile messages (Barber et al., 2016, 2015b). According to Barber et al. (2015a), a user-centered design approach was applied to the development of the MMI. This approach takes into account the task variables (i.e., situational constraints, frequency of the task, and rigidness of options available to complete the task) with the goal of meeting the demands dismounted Soldiers are exposed to operationally. For example, input and output of the device must not only capture and deliver the information between a Soldier and robot, but also be robust to noise and visual clutter while operating together or separated. As a result, the MMI enables multimodal communication, which supports redundancy and levels of communication that are more robust than single mode interaction (Bischoff and Graefe, 2002; Partan and Marler, 1999), and is tailored to the needs of the Soldier, even leveraging vocabulary preferences of Soldiers (Barber et al., 2014). The final design of the MMI incorporates this feedback into the fielded interface that we experimentally evaluate in Section 5.3. Figure 2 presents a screenshot of the MMI that shows the overhead map, robot view, and language interface.

3.1.2. Semantic Perception and Simultaneous Localization and Mapping

In order to intelligently interact with the environment, the robot must be able to perceive the environment and build an internal world model. The primary means for inserting objects into the world model is our image and range sensor-based perception system, which relies on RGB, RGB-D, and LIDAR measurements to detect, localize, and track object instances. The simultaneous localization and mapping module processed LIDAR and IMU data to produce map and state estimates used to inform the frame of reference of local observations and the state of the robot in the intelligence world model. The semantic perception module is comprised of three main components: an object detector, an object tracker, and an object pose estimator. We use Faster-RCNN (Ren et al., 2015)

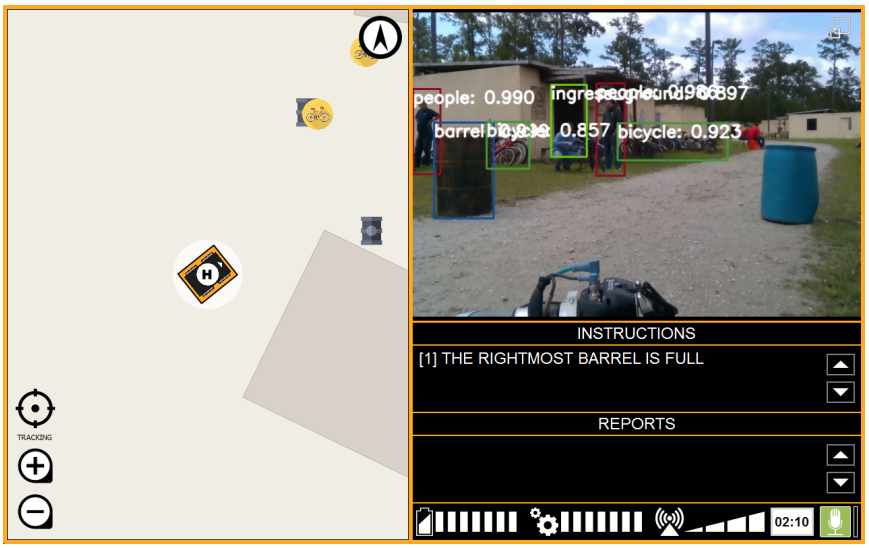


Figure 2. Visual display for the multimodal interface, which can be decomposed into three areas: map, video, and information. On the left, the situation awareness map depicts a top down, north-up representation of the environment with iconography for objects, buildings, and road networks. The top right portion shows video feeds from the robot, with the ability to subscribe to multiple channels (e.g., raw video, labeled video). The bottom right contains information and commands given to the robot, and its reports back (e.g., "I see two barrels and a bicycle"). Included below this information are system health indicators (e.g., battery, signal, run-time, microphone).

Table 1. Object classes in the RCTA dataset. Table replicated with permission from Narayanan et al. (2020).

Generator	Crate	Bench
Dumpster	Pelican Case	School Bus
Backpack	Suitcase	Gas Can
Debris	Trash Bin	Gravestone
Wood Pallet	Tower	Barrier
People	Toilet	Police Truck
Light Pole	Barrel	Gas Pump
Control Tower	Tank	Motorcycle
Shop	Window	Electrical Box
Gate	Chair	Bicycle
Table	Traffic Sign	Truss
PVC Pipe	Weapons	Stairs
Door Ground	Ingress Ground	

for object detection in RGB images, specifically the maskrcnn-benchmark implementation (Massa and Girshick, 2018). The object detector is trained to detect the RCTA specific object classes, using data from (Narayanan et al., 2020). Figure 3 visualizes example detections of some of these object classes. Table 1 provides the full list of object classes.

People detection and tracking is implemented in its own module, separate from the other object classes handled by semantic perception. Detection is achieved from monocular imagery using the Mask-RCNN instance segmentation model (Mertz et al., 2013). The monocular masks are then applied to a calibrated stereo image to obtain a robust 3D estimate for the detection in the reference frame of the camera using the RANSAC algorithm. The detection is then transformed into the world frame using vehicle odometry, where a person object instance is maintained in the world model. The

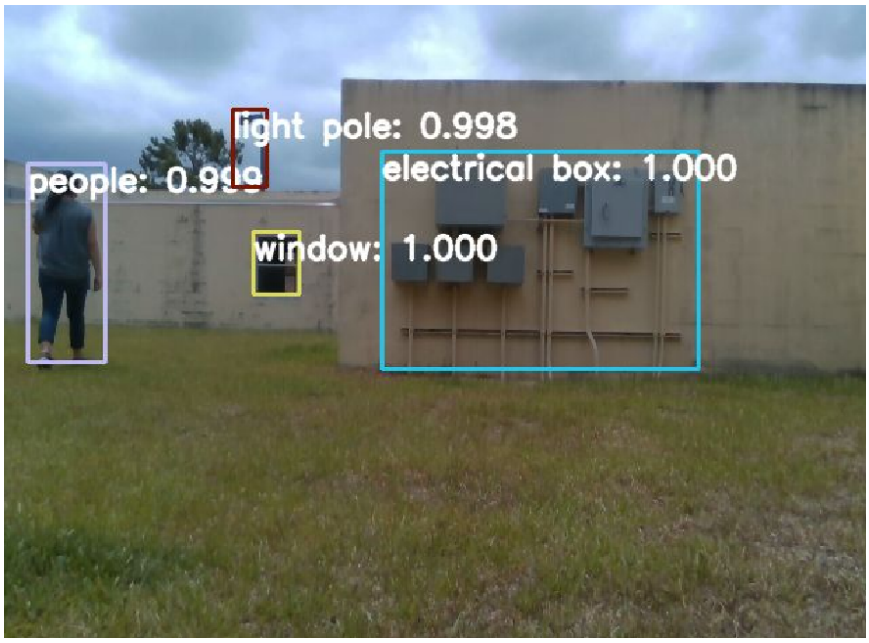


Figure 3. Example classifications of both objects and people at the testing environment.

effective range of this method is limited to about ten meters but could function both indoors and outdoors.

Person tracking is achieved using two integrated trackers. The first consists of a linear Kalman filter that tracks the position, height, and respective velocities of each person object in the world reference frame. A set of active tracks are maintained using a filter assigned to each track. When a new observation is made, optimal assignments to existing tracks are estimated using bipartite matching. Unmatched observations are initialized into new tracks. Similarly, tracks that have not been associated with an observation within a time threshold are removed from the set of active tracks. A second LIDAR-based method (He et al., 2017) tracks all entities above the ground plane. This tracker is designed to maintain the persistence of tracks outside the field-of-view of the stereo cameras, which was limited to sixty degrees. Due to the abrupt turning motions of the skid-steer robot, human tracks often translate in and out of stereo view and would be quickly lost if not maintained by a secondary tracker. The LIDAR-based tracker used a VLP-16 LIDAR with a 360-degree field-of-view, and considerably longer range than stereo. Furthermore, the update rate of the LIDAR tracker is much faster than stereo, making it more robust to rapid motion of the robot. Due to the limited resolution of LIDAR, however, this tracker could not differentiate between object classes of similar size, and as a result would track essentially any object that protruded from the ground plane. To integrate the trackers, two independent sets of active tracks are maintained for each of the trackers. Whenever a stereo track left the field-of-view of the camera, the track is associated with a corresponding LIDAR track if one exists. Similarly, when a new stereo track is created, all LIDAR tracks with stereo associations are queried and if a match is made, the stereo-track is reinstated using its previously associated state.

3.1.3. Intelligence World Model

As seen in Figure 1, many components of the intelligence architecture interact through object descriptions. To facilitate these interactions, the Intelligence World Model (IWM) is conceived as a simple object store that collects and tracks fused information about the presence and state of objects in the world. Earlier efforts on implementing a world model focused on combining metric and symbolic information into one unified structure (Dean et al., 2014), but this design was scaled

back in an attempt to discover what is the minimal design necessary to realize the language-guided mission concepts presented here.

The concept of an "object" is broadened beyond the object-level output of perception to include symbols such as regions and waypoints that are manually specified or created as part of the mission planning process. In particular, objects in the IWM consist of:

- A unique ID, autogenerated
- Object type and sub-type (e.g., type "cone" with sub-type "traffic")
- Object pose and velocity in global coordinates
- Object geometry, i.e., the bounding box from the original detection or a region description as a polygon
- Semantic property distributions (e.g., property "heavy" with distribution over true/false).

As a data store, the IWM allows for inserting, modifying, or deleting objects by ID; the IWM also has a simple query interface that allows for returning a list of all objects, objects by ID, objects by type and/or subtype, and objects with poses inside a given query region. The natural language understanding and generation modules use this query interface to pull the most recent set of relevant objects that are available for grounding.

3.2. Modules for Grounded Language Communication

After language is received by the architecture, it must be understood in the context of the perceived environment. The process of grounded language communication is performed by four modules. First, a string of text populated by the multimodal interface is converted into a parsed representation that includes both imperative and declarative components with symbols that could be extracted without the environment model. The meaning of this representation of the processed text is interpreted in the context of the observed environment model populated by the intelligence world model using variations of Distributed Correspondence Graphs (Howard et al., 2014) in the natural language understanding module. Inference is accelerated using symbols grounded before the instruction is observed by language and grounding pairs populated by a proactive symbol grounding module. This module opportunistically aligns phrases that the robot may encounter with symbols in the observed environment. This architecture is significantly different from purely reactive approaches to natural language understanding that wait until the instruction is observed to start aligning symbols. The natural language generation module is the last step in this architecture that is used when a natural language description of a symbol is needed for either disambiguation of a statement interpreted by the natural language understanding module or for generating text-based descriptions of facts observed by the deliberative interactive estimation module.

3.2.1. Parsing and Declarative Knowledge

The parsing and declarative knowledge module receives the natural language utterance from the MMI and performs syntactic analysis on the text. The module uses a grammar comprising a set of production rules that can generate the natural language utterances we are interested in. Some example production rules in our grammar that are applied in the examples explored in Figure 4 are shown in Table 2:

Table 2. Example production rules for the CKY parser used for converting text extracted to the parse trees shown in Figure 4.

VP → VB PP	PP → TO NP	PP → IN NP	NP → DT NN	NP → DT JJS NN
NP → NP PP	VB → *drive*	VB → *go*	IN → *behind*	IN → *on*
TO → *to*	JJS → *leftmost*	NN → *barrel*	NN → *left*	DT → *the*

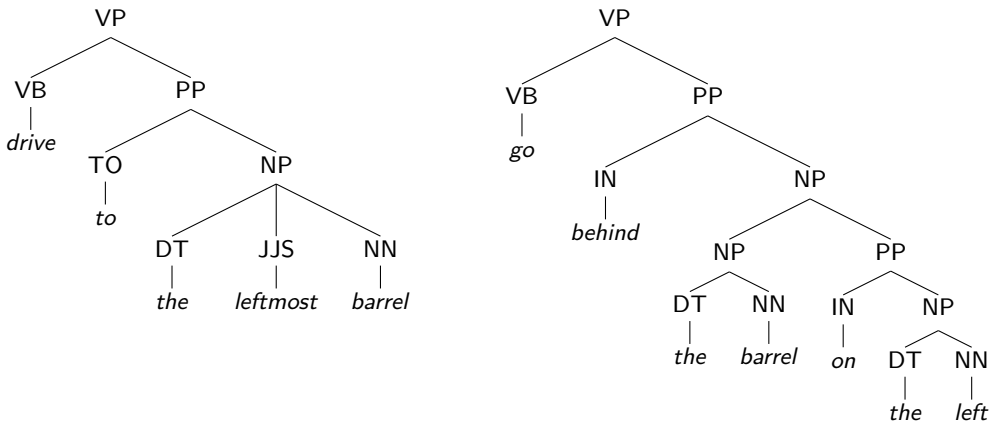


Figure 4. The parse trees for the imperative sentences (commands) "drive to the leftmost barrel" and "go behind the barrel on the left".

In these production rules, "drive" and "go" are terminal words whereas other rules like "NP → PP NP" are grammar non-terminals. We use this grammar to parse such input utterances into parse trees. Each parse tree provides a sequence of grammar production rules that can be used to generate the utterance. We use the CKY algorithm with beam search allowing us to recover multiple valid parses for a single utterance. Two example utterances and possible corresponding parse trees are shown below in Figure 4 for the phrases "drive to the leftmost barrel" and "go behind the barrel on the left".

The parse tree allows us to distinguish between imperative and declarative sentences. Imperative sentences are ones that convey instructions, and usually constitutes a single verb phrase (VP), as seen in the example above. We directly transmit the parse tree of such sentences to the grounding module. Declarative sentences on the other hand provide information or assign properties to the subject phrase of a sentence. For example, the declarative sentence "the barrel on the right is dangerous" assigns the property "dangerous" to the object referenced by "the barrel on the right". In these cases, we split the sentence into the reference of the object ("the barrel on the right") and the property mention ("is dangerous") using conditions on the parse tree. The property is mapped to a knowledge predicate (in this case, a predicate called IsDangerous(·)) and passed to subsequent modules, along with the natural language reference of the object.

The ability of a robot to understand, reason and follow commands is intimately tied to its knowledge about the world encapsulated in the world model. Previously, we discussed the representation of the world model that encompassed spatial knowledge about objects in Section 3.1.3 and described in detail a grounding model to relate language from a human commander with spatial knowledge about the world yielding a symbolic representation amenable for the planner to synthesize appropriate behaviors for the robot to execute. In essence, the system detailed so far enabled the robot to understand and respond to instructions such as "drive to the center cone, go to the barrel, pick up the gas can" etc.

Apart from the knowledge about the world arising from knowledge of entities and spatial relations, humans use and reason about factual or declarative knowledge about the world. We now consider the problem of interpreting and reasoning with declarative knowledge during language grounding. One class of declarative knowledge can relate to facts or properties associated with objects an object. For example, the utterance, "Robot, I am your commander, follow me" conveys the fact ("about being a commander") is associated with a particular object in the world model and is required to appropriately execute the following task. A second category of declarative knowledge relates to properties that may not be directly visible to the robot. For example, consider the utterance, "the barrel on the left is empty". Here, the human gives knowledge about the internal state of an object, not visible to the robot. This knowledge is crucial for deciding how to ground subsequent interactions

that rely on those expressed properties, such as if a human operator gives the instruction "go to the empty barrel". This instruction could not be resolved using purely visual information like object color or location, so the robot must either accept human guidance or physically interact with the object to resolve any ambiguities.

The ability to interpret language with acquired background knowledge requires bridging two technical challenges. First, the robot must determine the fact or property and to which object it relates to in the world model. Second, since the knowledge conveyed by the human must persist for future situated linguistic interactions, there is a need for a model to retain and update the acquired knowledge. Finally, the robot must be equipped with the ability to use the acquired knowledge in its language grounding model. Next, we describe the approach to bridge the above-mentioned technical challenges.

Given an estimate of the language input, the parser partitions the statement into imperative (commands) and declarative (factual) components. For example, "is empty" is the imperative part of the utterance "the barrel on the left is empty". We observe that factual knowledge can often be determined using linguistic cues. We use such cues embedded in the parse of the sentence to determine which phrases contain declarative knowledge. This is realized using a rule based parsing approach which additionally allows multiple candidates to be determined. Next, the imperative part of the sentence, "the barrel on the left" is interpreted using the natural language grounding model discussed in Section 3.2.2. We then associate the grounding of the imperative part and the declarative part to determine that the grounding for the utterance as an expression of the property "is empty" with the barrel-type object instance towards the left of the robot.

We now address the task of persisting declarative knowledge for future language grounding tasks. We adopt a probabilistic approach that represent the belief over the properties of objects in the world model as random variables. The estimated properties such as "is empty" serve as observations for updating the robot's belief in the world model. The linguistic observations serve as evidence and can be incorporated using a measurement update using a Bayes filter. For a detailed exposition, please refer to Paul et al. (2017) and Arkin et al. (2020). Finally, when a robot encounters an instruction such as "inspect the empty barrel" the language grounding model probabilistically associated the utterance "is empty" with the belief over the property (IsEmpty) in the world model. The grounding model incorporate features that represent the degree of uncertainty inherent in the robot's knowledge. Once the instruction is correctly interpreted, the grounded symbol is used for behavior synthesis in future stages of the pipeline.

3.2.2. Natural Language Understanding

Once the system is able to identify constituents of the statement that need to be grounded, the natural language understanding module uses this linguistic information in the context of an environment model distributed by the centralized Intelligence World Model. Grounding is performed using variations of the Distributed Correspondence Graph (DCG) (Howard et al., 2014). The DCG shares the assumption of the Generalized Grounding Graph (Tellex et al., 2011) by assuming conditional independence across linguistic constituents when constructing a factor graph. The DCG however further assumes conditional independence assumptions across constituents of the symbolic representation that defines the space of groundings. In these models the size of the symbolic representation is a function of the observed environment and the complexity of symbols understood by the intelligence architecture. An illustration of the DCG for the three-phrase statement "drive to the leftmost barrel" and the DCG for the six-phrase statement "go behind the barrel on the left" are shown in Figure 5 along with their alignments to the corresponding parse trees in Figure 4.

The white and gray circles in Figure 5 represent known and unknown random variables in the DCG for the example expressions. The factors f_{ij}, represented as black boxes in Figure 5, provide the connections between the unknown correspondence variables (ϕ_{ij}) the current phrase λ_i, a symbolic constituent γ_{ij}, and any child groundings expressed by a non-false correspondence of the child phrase, represented as potential connections between the factor and all child phrase symbolic constituents. All factors also have an implicit connection to the environment model Υ that serves as an important

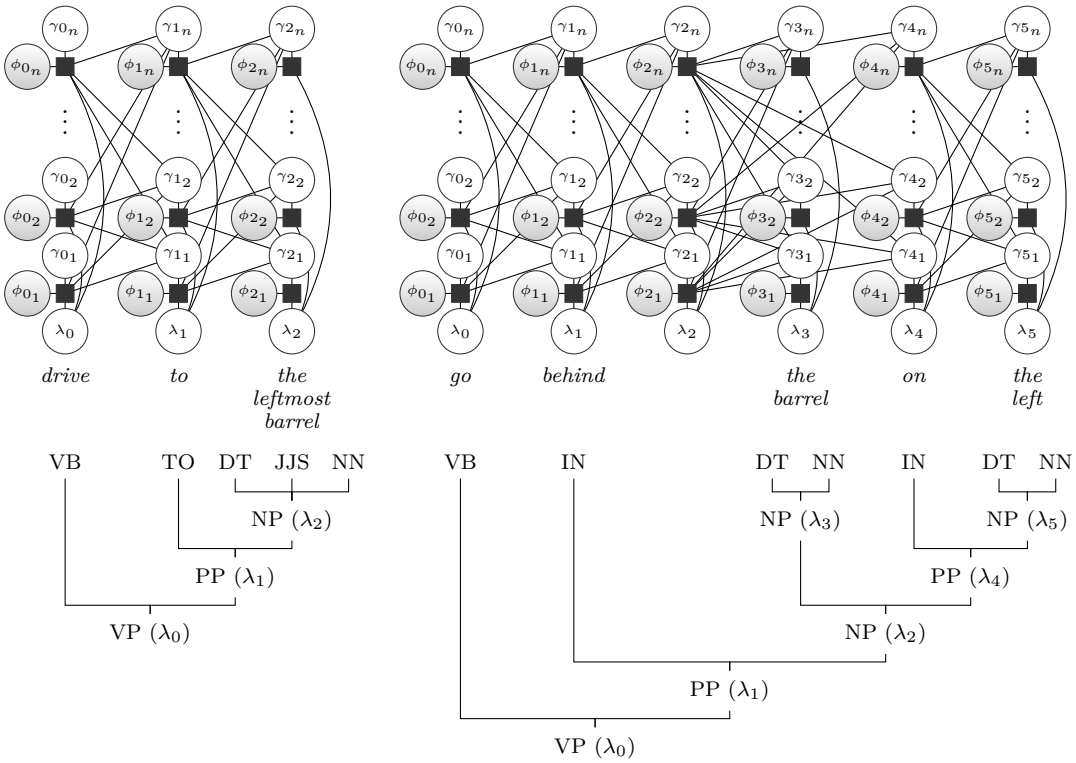


Figure 5. The DCGs for the expressions "drive to the leftmost barrel" and "go behind the barrel on the left" aligned with their corresponding parse trees from Figure 4. The structure of each DCG is a function of both the parse tree and the number of symbols expressed by the symbolic representation and the environment model. Each factor f_{ij} in the factor graph is illustrated as a black box. The observed and unknown variable nodes in the factor graph are shown in white and gray, respectively. Each factor also has an implicit connection to the environment model Υ.

signal for determining the meaning of expressions that involve spatial relations and the context of the symbols expressed from the prior utterance for monologue comprehension and disambiguation (Arkin et al., 2017). The conditional independence assumption across symbolic constituents here is valid for the space that we consider. Even if symbolic constituents represented at various parts of the inference are independent, the model has full context of the environment model to ground example expressions like "the leftmost barrel" or "the barrel" because that model includes all information about all observed objects with a semantic class that is associated with objects and the word "barrel" in the training data. Mathematically this inference appears as Equation 1. The conditional probabilities in this model are approximated using log-linear models trained from corpora of fully annotated data as described in Paul et al. (2018). Inference is performed using beam search to efficiently prune off low-likelihood combinations of symbols during search for the most likely set of correspondence variables. In this formulation, the child correspondence variables Φ_{c_i} determines the expression of the child symbols and the conditional probability is expressed as a function of those values.

$$\Phi^* = \arg\max_{\phi_{ij} \in \mathbf{\Phi}} \prod_{i=1}^{|\Lambda|} \prod_{j=1}^{|\mathcal{G}|} p(\phi_{ij} | \gamma_{ij}, \Phi_{c_i}, \lambda_i, \Upsilon) \tag{1}$$

In the DCG, the runtime of the inference process increases linearly with the number of symbols and phrases considered by the utterance since there is a constant number of correspondence variables ("true" and "false" for binary correspondence variables) for each factor. This linear

growth is problematic in cluttered environments composed of many objects and complex symbolic representations capable of representing many different types of objects, spatial relations, regions, actions, sets of objects, sequences of actions, etc. The Adaptive Distributed Correspondence Graph (ADCG) (Paul et al., 2016) proposed a solution to this problem by adapting the expression of the symbolic representation within the inference procedure so that only subsets of the larger symbol space would be expressed based on the expression of other symbols. For example, the model was able to learn that the statement "five blocks on the right" could assume only spaces of objects containing a number equal to the value interpreted by the meaning of the word "five" and include only those collections of objects with semantic labels learned to associate with the label "blocks". From that sampled space of object sets, the unique meaning of "five blocks on the right" would be resolved by searching in the context of the spatial relation represented by the phrase "on the right". This model was shown to significantly improve the computational efficiency of probabilistic inference in comparison to DCGs where the symbolic representation is fully explored at inference time.

Other variations of DCGs, such as the Hierarchical Distributed Correspondence Graph (HDCG) (Chung et al., 2015) and Hierarchical Adaptive Distributed Correspondence Graph (HADCG) (Paul et al., 2018), also were shown to improve the computational efficiency of natural language symbol grounding. The HDCG treats inference as a two-step procedure where first the symbolic representation is constrained by inferring bounds using only the phrases in the instruction. Grounding is performed in a distribution of these simplified models and aggregated to interpret the meaning of the utterance. Using one of the statements from Figure 5 as an example, the sentence "drive to the leftmost barrel" does not necessarily need information about the physical location of barrel, vehicle, building, and other semantically classified objects not learned to associate with the phrase "barrel" or spatial relations learned to correspond with the word "leftmost" to accurately interpret the meaning of this statement. Pruning these symbols from the space of symbols considered based purely on language was shown to significantly improve the computational efficiency of heuristic search. There are however many examples where this model does not provide sufficient pruning. For example, in an environment with many instances of the expressed object and the symbolic representation that must consider ordering of objects within identified sets, HDCG-based pruning may not substantially reduce the size of the symbolic representation. The symbolic representation for the instruction "pick up the middle block in the row of five blocks on the right" in an environment containing more than a dozen blocks would not exclude all variations of blocks and sets of blocks from consideration like the ADCG model would. Observing that aspects of the ADCG and HDCG could be combined to provide even more efficient approximations of the symbolic representation, the HADCG (Paul et al., 2018) combined these ideas into an approach that demonstrated an improvement in the efficiency of language grounding without a loss of accuracy over the DCG, HDCG, and ADCG in a series of experiments involving manipulation and navigation commands. Variations of these models with domain specific symbolic representations have been applied for applications involving language grounding for synthesis of verifiable controllers (Boteanu et al., 2016, 2017), adaptive grasp control (Esponda and Howard, 2018), planning corrections for assistive robotic manipulators (Broad et al., 2017), and homotopy-aware motion planning (Yi et al., 2016).

The symbolic representation for the experiments described in Section 4 and reported on in Section 5 was an extension of the symbolic representation discussed in Paul et al. (2018). Novel symbols used to represent temporal relationships, temporal relationships between actions, and temporal constraints between actions like "before" and "after" and features that considered the relationship of such symbols to those previously inferred by prior utterances permitted symbol grounding of instructions like "navigate to the pelican case on the right and then report," "instead navigate to waypoint bravo," "go to the case but first go to the gascan" and "after that navigate to the barrel". Novel symbols for action properties and object properties were also implemented to permit the inference of symbols that describe the semantic properties of an action or object, such as an action that is to be performed "quickly" or an object described as a "dangerous barrel," without needing to resolve the connection to a physical object at that stage of DCG inference.

3.2.3. Proactive Symbol Grounding

Grounded language communication systems often treat the problem of language understanding as a reactive process in which a human first provides an utterance in the context of the current world to then trigger the process of grounding the language into physical concepts and entities. A reactive approach may seem intuitive as a result of the inherent dependence of the meaning of the utterance on the current state of the world. However, this reasoning does not apply to predominantly static environments. In practice, there is often system idle time while a robot teammate awaits an instruction from their human collaborator; if this idle time occurs in the context of a static world, it is possible to instead take a proactive approach to language understanding. By anticipating likely utterances that a human might say, or phrases they might use, the robot can precompute the meaning and store the result for potential reuse once an utterance is provided.

The proactive symbol grounding module of the intelligence architecture is designed to use idle system time to perform this kind of pre-computation. It is provided with an identical grammar model as used by the parsing module (i.e., the grammar production rules and set of possible linguistic tokens) that drives the generation of a subset of possible utterances and phrases that the human teammate might express. In principle, this ungrounded language generation process need only be done once and is performed in a bottom-up approach.

Given the generated subset of possible utterances and phrases, this module can sample examples and compute their associated grounded symbolic representation in exactly the same fashion as would be done reactively. This sampling is done in a bottom-up approach to conveniently align with its DCG-specific utility of providing insertable partial solutions due to the phrase-level conditional independence assumptions of the model formulation (i.e., leaf phrases are most likely to be reused, with that likelihood decreasing as parse tree depth increases). Once computed, the phrase and symbol pairs are stored in a lookup table and maintained for the duration of time that the static world assumption holds. Since the grounded symbolic meaning of each phrase will become stale when the world configuration or content changes sufficiently, this module regularly queries the intelligence world model module and will discard its stored solutions if it can no longer guarantee their validity.

Adopting the notation of DCGs from Equation 1, let $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_N\}$ be a novel utterance provided by a human teammate and $\Lambda^{psg} \subset \Lambda$ indicate that there exists a subset of phrases in Λ that have been precomputed via the proactive symbol grounding process. These Λ^{psg} have an associated set of the most likely *true* correspondences Φ^{psg}. If we let $\Lambda^{new} = \Lambda \setminus \Lambda^{psg}$, then it directly follows that $|\Lambda^{new}| \leq |\Lambda|$. The reactive inference process triggered by the novel utterance Λ can be reformulated to only search over Λ^{new} as shown in Equation 2 below. This modified reactive inference process can be thought of as being bootstrapped with precomputed solutions for parts of the novel parse tree.

$$\Phi^* = \underset{\phi_{ij} \in \mathbf{\Phi}}{\arg\max} \prod_{i=1}^{|\Lambda^{new}|} \prod_{j=1}^{|\mathcal{G}|} p(\phi_{ij} | \gamma_{ij}, \Phi_{c_i}, \lambda_i, \Upsilon) \tag{2}$$

The proactive symbol grounding module provides a service to the natural language understanding module to populate the solutions for Λ^{psg} given the utterance Λ. It accepts a query message containing the parse tree matching the full novel utterance provided by the human (as constructed by the parsing module earlier in the pipeline). Upon receiving a query, it performs top-down search for matching phrase structures in its stored solutions, inserts the corresponding symbols for the found matches and their associated child phrases, and sends a reply message containing the populated parse tree back to the natural language understanding module. In the best case, the returned tree is fully annotated at the cost of a lookup; in the worst case, the returned tree is empty at the cost of lookups for each phrase in the tree.

The module also provides a service to the natural language generation module. It accepts a query message containing the symbols for which the language generation module is looking for a corresponding utterance. Upon receiving a query, it iterates through all of the stored solutions to find a matching root-level set of symbols. If found, it sends a reply message containing the most likely

corresponding phrase or utterance. If not found, it processes any remaining ungrounded phrases for the current world and sends a reply message containing the most likely utterance. In some cases, no phrase corresponding to the input set of symbols is found and a reply message is sent containing a failure signal.

3.2.4. Natural Language Generation

In collaborative grounded language communication systems, it is often useful for a robot to say things to a human teammate, whether to clarify ambiguous statements or commands, communicate knowledge updates gained through physical interaction with the world, ask for help to accomplish a task, among others. In the context of a symbol-based meaning representation, speech generation can typically be considered as a problem of generating the most likely corresponding utterance for a symbol set that represents meaning of the clarification, knowledge update, or query for assistance. The natural language generation module's role within the intelligence architecture is to solve precisely this problem for a provided set of symbols associated with the first two situations: (1) clarifying an ambiguous statement or command and (2) conveying knowledge updates about the world as acquired via inference during physical interactions.

The problem of finding the best utterance given a set of symbols can be solved by inverting the language understanding problem in which the most likely symbol set is inferred for a given utterance and world model. This is sometimes referred to as *inverse semantics* (Tellex et al., 2014). In practice, inverse semantics can be implemented as a sequence of language understanding problems for a set of linguistic candidates and choosing the most likely candidate with a corresponding set of symbols that matches the desired symbol set to be communicated.

As also described for the proactive symbol grounding module in Section 3.2.3, the natural language generation module produces a set of linguistic candidates according to an identical grammar model as used by the parsing module. Given the set of linguistic tokens and production rules, this module produces a subset of the possible utterances and phrases. This ungrounded language generation process need only be done once and is performed in a bottom-up approach. In order to prevent recursive construction of an infinite set of candidates, the generation process is constrained by a user-specified parse tree depth limit.

The problem of inverse semantics using DCGs can be formalized as estimating the most likely utterance Λ^* from the generated subset of possible utterances $\{\Lambda_1, \Lambda_2, ..., \Lambda_N\} \in \mathbf{\Lambda}$ given the known corresponding set of symbols of the intended meaning Γ^Λ, the associated *true* correspondences Φ^Λ, and the state of the world Υ:

$$\Lambda^* = \underset{\Lambda \in \mathbf{\Lambda}}{\arg\max} p(\Phi^\Lambda = \text{true} | \Lambda, \Gamma^\Lambda, \Upsilon) \tag{3}$$

As mentioned, this is implemented as a sequence of language understanding evaluations in which the input language parameter is assigned to the next element in the set $\mathbf{\Lambda}$. This imposes a computational cost that directly impacts runtime performance. While using DCGs as the language understanding model does provide runtime performance improvements, the cumulative runtime cost for all $\Lambda \in \mathbf{\Lambda}$ is non-trivial and plausibly prohibitive with respect to the desired or required mission tempo. It is important to provide additional mechanisms with which this grounded language generation process can produce faster solutions.

Fortunately, the proactive symbol grounding process and associated stored solutions can fill this role. By construction, the set of linguistic candidates used by both the proactive symbol grounding module and the natural language generation module are equal. Any solutions stored by the proactive symbol grounding process can be treated as precomputed solutions for the inverse semantics problem and can be used to effectively bootstrap the process at the cost of lookup. The resulting cardinality of the set of utterances that need to be reactively computed by the inverse semantics process is necessarily less than or equal to the initially constructed problem: $\mathbf{\Lambda}^{new} = \mathbf{\Lambda}^{psg} \setminus \mathbf{\Lambda} \leq \mathbf{\Lambda}$. Equation 3

can thus be reformulated as:

$$\Lambda^* = \arg\max_{\Lambda \in \mathbf{\Lambda}^{new}} p(\Phi^\Lambda = \text{true} | \Lambda, \Gamma^\Lambda, \Upsilon) \tag{4}$$

In the best case, the proactive symbol grounding process will have exhausted the full set of generated possible utterances and the cost is that of searching over a stored list of solutions to find the most likely utterance with a corresponding set of symbols that matches Γ^Λ. In the worst case, the proactive symbol grounding process will have computed no solutions and the cost is equivalent to the cost of computing Equation 3 with a trivial message-passing overhead.

3.3. Modules for Mission Planning and Execution

Once the robot has a symbolic representation of the task that the human wants the robot to perform, it must sequence and initiate the necessary behaviors to complete the action. The mission planner receives the symbolic representation of actions, modes, and constraints and then constructs behaviors trees for the mission planner to operate on. The navigation planner, manipulation planner, deliberative interactive estimator, and scene description modules generate and send commands to the robot or provides feedback to the operator through the multimodal interface.

3.3.1. Mission Planner

The role of the mission planner is to take the set of symbols produced by language grounding and produce an executable behavior tree that will accomplish the intended action (Colledanchise and Ögren, 2018). Our mission planner is an implementation of the PA-BT algorithm (Colledanchise and Ögren, 2018): planning is carried out as an ongoing behavior tree expansion that only triggers when the system encounters unsatisfiable conditions, in the spirit of "Planning in the Now" (Kaelbling and Lozano-Pérez, 2011). Missions are initialized with behavior trees that consist of only the high-level conditions satisfying the grounded language commands. These conditions are a direct translation of the action symbols produced by the natural language understanding system (Sec. 3.2.2). The translations used for the experiments described here were:

- `navigate`(*object*) becomes the condition for the robot to be within a pre-configured radius of given object
- `follow`(*object*) becomes the condition to be following the given object
- `report` becomes the condition to have "reported" (a dummy condition)
- `push`(*object*) becomes the condition to have "pushed" the given object (a dummy condition)

When multiple action symbols are grounded, their translated goal conditions are concatenated into a sequence node with memory, to ensure that each is done only once. If action ordering constraints are provided in the grounding, then these are used to define a partial ordering over the action symbols and the actions are sorted by this ordering before translating and concatenating. Manipulation planning (Sec. 3.3.4) was not integrated into the mission planning system for these experiments.

As described in Colledanchise and Ögren (2018), the main job of the planner is to attempt to expand failed conditions into sub-trees that attempt to make them true and then re-execute the behavior tree. These expansions are encoded as action templates, or small behavior trees that follow a common pattern (i.e., the *postcondition-precondition-action*, or PPA, template) for describing how actions can be used to satisfy a condition and parameterized by the arguments to the condition. In our system, complexity of missions was largely driven by complexity of language, so our conditions were actually trivially satisfied by a particular action. An example of the expansion of the *at_object* condition is seen in Figure 6.

"Dummy" conditions will always evaluate to false but have defined action templates that will expand out into a more complex set of conditions and actions. As an example, the "*reported*" condition has an action template that expands into the `report` action that triggers the scene

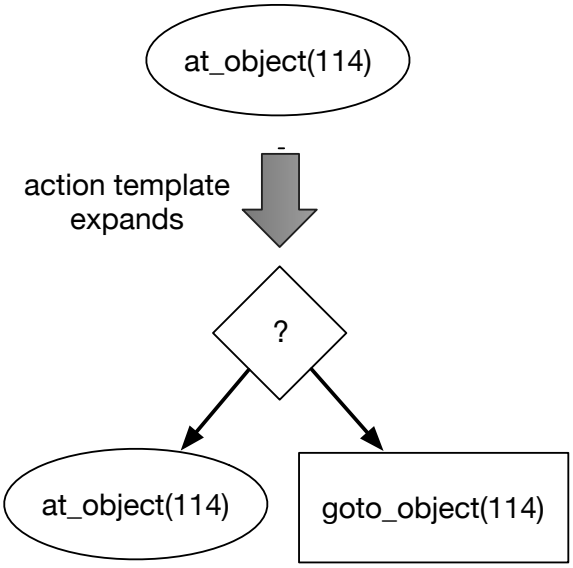


Figure 6. Simple action template expanding a navigation condition. The behavior tree uses a selector node to perform a sequence of actions that determines if the robot satisfies the condition described by "at_object(114)" and performs navigation action described by the "goto_object(114)" command.

description system (Sec. 3.3.6). In this way, we can further decouple requests for actions from the behaviors that produce those actions and leave a path to having multiple approaches that could satisfy those requests.

The planner actually proceeds by a string of failed behavior tree executions: after the initial tree consisting of only goal conditions is executed and likely fails, the failed conditions are expanded one-by-one and the new trees are executed until reaching the next failed condition. The mission is declared a failure only after trying to expand every failed condition and still failing.

Though the command language here is structured enough that it would have been possible to build a system that directly translated symbol groundings into full behavior trees, implementing it as a real mission planner capable of composing action templates on-the-fly to expand the plan affords a degree of modularity that behavior trees are well-suited to exploit. This modularity also lets us package up actions together with the action-templates that make use of them and dynamically inject these into the planner when that action is initialized during system setup. The result is a planning system with a dynamic planning domain, built by the actions that are actually available.

3.3.2. Mission Executor

The mission executor is responsible for linking platform state, mission state, and available control behaviors in order to continuously execute a behavior tree (Colledanchise and Ögren, 2018) until it returns SUCCESS or FAILURE. The key details of the behavior tree implementation are in the handling of conditions and actions. Conditions are implemented as statements in the `lua` programming language (Ierusalimschy, 2006) for flexibility. The embedded `lua` interpreter is augmented with primitive functions that allow for interacting with the intelligence world model and getting state information from ROS such as the robot position. Actions are implemented as modules, specific to each action type, that can take parameterized behavior tree action labels and translate them into calls to underlying behaviors through ROS ActionLib[1] interfaces.

[1] https://github.com/ros/actionlib

During operation, the mission executor node receives new behavior trees from the mission planner, sets up all condition-checking and action-dispatching functions before initiating execution, and then executes the behavior trees in a continuous loop as long as the tree returns a status of RUNNING. As discussed in Sec. 3.3.1, the mission planner can send behavior trees that are not fully expanded with the expectation that the mission executor will run until it reaches the next step that needs to be expanded. Normal operation can see many rounds of failed executions and subsequent plan expansions until the behavior tree has been expanded enough to successfully execute until the original goal conditions are satisfied.

3.3.3. Metric Planning and Execution

With a symbolic goal resolved to a metric goal, the behavior-tree mission executor relies on a metric planning and execution system to attempt that goal and report success or failure accordingly. We follow a classic model, splitting this effort into one of finding a kinematically-feasible path from the robot's current pose to its goal, i.e., global planning, and computing a receding-horizon optimization that corrects for errors and accounts for local, possibly dynamic, obstacles at a higher resolution, i.e., local planning.

Kinematically-feasible motion planning is computed with the Search-Based Planning Library (SBPL)[2]. We generate a custom set of motion primitives based on a maximum curvature of 0.4 and 0.2 m occupancy-grids. We use the ARA* planning algorithm and compute plans from the goal to the start so that computations can be reused as the robot drives for fast re-planning actions. Re-planning allows the system to quickly correct its path in the event of errors in platform control or updates of the occupancy-grid map. Feasible solutions to most initial planning queries are found in less than a second with optimal solutions being found in a few seconds for most scenarios, providing collision-free paths that incorporate turn-in-place and k-turn maneuvers to move through tight spaces.

We implement a receding-horizon model predictive controller to compute optimal trajectory generation over the space of time-varying control inputs in order to provide local planning to the system. Based on prior work in trajectory generation (Howard and Kelly, 2007), we formulate a parameterization of the control input for a differential-drive platform such that a relatively small number of variables (8) to provide an expressive description of the possible trajectories available to the robot over a short time horizon of 3 seconds. An objective function is devised that performs a weighted minimization of the error between the robot's path and the desired global path coupled with some curvature minimization terms to prevent overly aggressive trajectories. The final optimization problem, including bounds on the parameterization of the control input, can be solved with a variety of algorithms implemented in the NLOPT library[3]. We are typically able to solve the trajectory generation optimization for a time horizon of $T = 3$ s in 5 to 10 ms, allowing for a control frequency of 10 Hz. Finally, the system directly commands the optimized time-varying control inputs to the robot's underlying motor-control system.

There may be scenarios during mission execution where the robot should navigate in a more deliberate manner rather than simply planning an obstacle free shortest path. For example, the robot could evaluate environment terrain to identify more easily traversable areas, or it may consider using objects in the environment to operate more covertly when a threat is perceived during mission execution. To achieve this, the mission executor passes attributes along with metric goals to the navigation system that trigger the use of an alternative global planner that produces motion plans learned from human demonstration. Specifically, the planner uses a reward function learned via inverse optimal control (IOC), or commonly referred to as inverse reinforcement learning (IRL).

In our IOC problem formulation, we learn a linear reward function, $R(s) = \theta^T \phi(s)$, where $\phi(s)$ represents the set of semantic features for state s, and θ are the feature weights we optimize to

[2] https://github.com/sbpl/sbpl
[3] http://ab-initio.mit.edu/nlopt

Table 3. Manipulation stack operation for various actions: Prep is base repositioning for reachability, Grasp attempts grabbing an object, Lift and Place are object-in-hand manipulation actions.

Action	Operation
Prep	WBP consolidates ROI candidates and Object to Base goals are passed to IC.
Grasp	Grasp planning examines the selected ROI and resulting grasp poses are tried by WBP.
Lift	EE goal with wrench compliance parameters are given to IC.
Place	A "dropoff" location, as a Base to EE goal, is passed to IC.

encode the demonstrated behavior. Semantic features, ϕ, are represented as binary occupancy grids (and blurred versions of these) that encode, for example, the presence/absence of grass and road terrain derived from the semantic segmentation algorithm running in the perception system, and obstacles coming from LiDAR. We employ the maximum entropy IRL approach (Ziebart et al., 2008) for reward function learning given human-teleoperated trajectories as demonstrations. Our learning approach for navigation behaviors has been extensively tested in previous field experiments for edge of road following and covert behaviors (Wigness et al., 2018).

During deployment, the semantic-aware planner generates a cost map using the learned reward weights and feature extraction function. Given the robot's current location and goal pose, the planner searches for the lowest-cost trajectory that reaches the goal. This trajectory is then executed by the navigation system with a kinematically-feasible approximation to this solution by the local planner described above. In this work, we integrate a single learned IOC behavior into the mission planning and execution. This behavior encodes traversal patterns that maintain relatively close proximity to the edge of a road. During experimentation we use the natural language command "Go covertly" to trigger the execution of this navigation planner[4].

3.3.4. Manipulation Planner

We developed the manipulation stack around several layers of abstractions that allow for the separation of concerns to facilitate interoperability and platform independence. Communication among the layers utilizes the ROS Actionlib[5] interface to reliably pass along goals, feedback and results. A mobile manipulator may receive various manipulation actions such as Grasp, Lift, or Place from the Mission Executor. We will briefly explain the integrated planners and controllers below and summarize stack operations for each action in Table 3.

Operation in the first layer, which we define as the "Extrinsic Layer", is divided between global and local planners. The global planner references the map frame which includes object instances, traversability and 2D cost maps to solve the point A to point B problem. Typically, these are our navigation planners as described in Section 3.3.3. Our manipulation planners locally plan with respect to the robot frame and uses search-based algorithms on 3D occupancy grids to generate collision-free trajectories for object to end-effector (EE) goals. This choice of reference frame significantly reduces planning times. At this layer, we introduce a planning arbiter known as the Whole Body Planner (WBP) (Kessens et al., 2020) in which coordinates arm movements through a diversity of planners and movement profiles. Successive planning algorithms in the stack enable a mobile manipulator's prehensile manipulation of objects within its task space. Initially, grasp planning using the Grasp Pose Synthesis (Detry et al., 2017) algorithm returns kinematically feasible EE poses that are geometrically suited for our particular gripper within a selected region

[4] Although we use the adverb "covertly," we note that this specific behavior does not accurately represent the definition of covert used in other fields. We simply use this as a stand-in to demonstrate the differences between simple obstacle avoidance and semantic-aware planners.
[5] https://github.com/ros/actionlib

known as the region-of-interest (ROI) (Kessens et al., 2020). WBP then invokes two motion planning algorithms in parallel: Anytime Repairing A* (ARA*) (Likhachev et al., 2004) and Generalized Lazy Search (GLS) (Mandalika et al., 2019) to search for a solution on a graph in the 15-dimensional configuration space of both arms and torso. The implementation is a Search-based Motion Planning Library (SMPL)[6] plugin for the MoveIt! (Chitta et al., 2012) framework, and a GLS[7] plugin for the AIKIDO[8] framework, respectively. For detailed exposition of each algorithm, the reader is directed to Mandalika et al. (2019) and Likhachev et al. (2004).

The second layer, which we call the "Intrinsic Layer", runs underneath the "Extrinsic Layer" and is responsible for reachability mapping based on inverse kinematics, and trajectory discretization/following through a whole body controller termed Intrinsic Controller (IC) (Kessens et al., 2020) and the open-source ROS Joint Trajectory Controller[9]. The IC is capable of performing short Cartesian arm and base movements. A wrench reactive component of the IC counters reactive forces and torques during motion, essentially creating a "software compliance." This is achieved by deflecting the goal pose of the Cartesian motion as a linear function from measured wrist torques and a pre-specified compliance parameter. Such a strategy is particular useful during motion with extended arms or with an object-in-hand. For general trajectory following, we use the Joint Trajectory Controller.

The final layer, which we call the "Hardware Layer", is where real-time motor control and state estimation occur. Modular mechanisms and proprioceptive sensors use an EtherCAT server-client interface.

3.3.5. Deliberate Interactive Estimation

The mission planner is required to robustly carry out high-level language commands, but the lack of world knowledge, which is visually imperceptible or hard to estimate from noisy measurements, leads to failures in synthesizing and executing plans robustly. For example, a robot asked to "Lift the heavy suitcase" may not know which of several objects semantically identified as a "suitcase" is heavy based purely on visual observations. For robust decision making in the absence of knowledge provided by a human operator, we introduce a deliberate interactive estimator that incrementally estimates semantic knowledge from noisy language descriptions, physical interactions, and background knowledge using our previous work. This section describes the deliberative interactive estimation module, which is an extension of our previous work on Bayesian multimodal semantic-knowledge estimation that appears in Arkin et al. (2020).

The estimation of semantic knowledge is challenging as it involves distilling high-level semantic knowledge from low-level sensory observations or language utterances. Conventional grounding approaches use the declarative knowledge in utterances as correct semantic knowledge for task execution (Matuszek et al., 2012; Thomason et al., 2016; Paul et al., 2017; Kollar et al., 2013; Perera and Allen, 2013). Similarly, low-level noisy sensory observations (e.g., vision or force/torque) have been directly used for estimating world knowledge such as traversability (Bhattacharjee et al., 2014) or abnormality (Park, 2018). In contrast, our method incrementally updates the semantic knowledge from language and force/torque interactions to make the mission planner more robust to inaccurate or incorrect instructions.

The mission executor triggers an interactive estimation action. Then, our estimator computes the probabilistic semantic knowledge state K_t at time t given past linguistic inputs $\mathbf{\Lambda_{0:t}}$, physical interaction measurements $\mathbf{Z_{0:t}}$, and the space of grounding symbols Γ: $p(K_t|\mathbf{\Lambda_{0:t}}, \mathbf{Z_{0:t}}, \Gamma)$. The knowledge state K_t represents latent object attributes such as "emptiness." We model the likelihood of the attribute using a *beta*-distribution parameter α_t. By introducing the correspondence variable

[6] https://github.com/aurone/smpl
[7] https://github.com/personalrobotics/gls
[8] https://github.com/personalrobotics/aikido
[9] https://github.com/ros-controls/ros_controllers

Figure 7. Experiment demonstrating the deliberate interactive estimation with physical interaction. The Husky robot with a UR5 arm estimates the latent semantic attribute "emptiness" by pushing the barrel.

Φ_t used in grounding processes, we can factor the distribution over K_t as

$$p(K_t|\Lambda_t, Z_t, \alpha_{t-1}, \Gamma) = \sum_{\Phi_t} \underbrace{p(K_t|\Phi_t, \alpha_{t-1}, \Gamma)}_{\substack{\text{Knowledge Belief} \\ \text{Update}}} \underbrace{p(\Phi_t|\Lambda_t, Z_t, \Gamma)}_{\substack{\text{Language \&} \\ \text{Interaction} \\ \text{Groundings}}}. \tag{5}$$

Assuming the conditional independence between observations, we factor the language and interaction groundings to $p(\Phi_t^\Lambda|\Lambda_t, \Gamma)$ and $p(\Phi_t^Z|Z_t, \Gamma)$, respectively. The former factor models the factual knowledge in declarative language utterances. The later factor models the correspondence between interactive measurements and object attributes in groundings. We particularly train two hidden Markov models that output the observation likelihoods $p(Z_t|m_{\text{True}})$ and $p(Z_t|m_{\text{False}})$ conditioned on the presence and absence of the attribute, respectively. The likelihood ratio $p(Z_t|m_{\text{True}})/p(Z_t|m_{\text{False}})$ gives the likelihood of correspondence Φ_t. To update the knowledge belief, we incrementally update the *beta*-distribution parameter α_{t-1} with respect to the binarized correspondence variable.

Our experimental studies in Figure 7 shows that, when a user provides incorrect declarative knowledge, such as describing a barrel as full when it is, a robot can quickly correct and report the true latent property to the user after manipulating the barrel. This work marks a qualitative change of robot capabilities by having a shared mental model in human-robot team scenarios.

3.3.6. Scene Description

The scene description module is where the last of several actions that the robot can perform is implemented. The ability to summarize the salient information in a scene is important for several reasons. First, it can be prohibitively expensive to stream high-resolution images over a low-bandwidth network. Second, streaming images still requires that the human visually inspect and process these images. A robot that is able to translate the information contained in an image and provide that information in text or audial form to a human operator may be enabling in a number of collaborative robot scenarios. Most current research in image or scene captioning focuses on the fluency of the generated language but not on the utility of the generated description.

The approach explored here applies the technique described in Tian and Oh (2019) that combines sequential and compositional models by following a word-by-word generation process and combining *grounded* attributes from *specialized* modules. This approach follows the idea of the Neural Module Networks (NMNs) (Andreas et al., 2016) used for the visual question answering problem, where each module is responsible for a specialized functionality and the final result is a dynamic composition

of different modules. In this approach, Compositional NMNs are designed for image captioning to generate a final caption by dynamically attending to different modules. The attributes of this model, therefore, have a hierarchical dependency on and are grounded to the proposed regions. For example, this model should generate descriptions like "a red apple" instead of "a piece of fruit" and "three people" instead of "a group of people."

The model used in the scene captioning module consists of three components: Recurrent Neural Network (RNN) Trio, Stacked Noisy-Or Object Detection, and Modular Attribute Detection. The object and attribute detection of the stacked noisy-or object detection mechanism and the modular attribute-detection mechanism makes up the compositional component while the RNN trio incorporates the detection results to generate a sentence in a sequential manner. Mathematically, this can be seen as computing a distribution, P_t^m over labels for module m at time t using a softmax-activated function denoted by f_m:

$$P_t^m = f_m(\tilde{v}_t, h_{t-1}^s, w_t^{obj}). \tag{6}$$

The outputs of the modules are word vectors $w_t^m = E_m P_t^m$, where E_m is the word embedding matrix for module m. The compositional module attention mechanism selects which model to use depending on the context. Inspired by Lu et al. (2017), the implementation used an adaptive attention mechanism and a softmax operation to get an attention distribution of the modules:

$$z_t = W_z^\top \tanh(W_m w_t^m + W_g h_{t-1}^s) \tag{7a}$$

$$\alpha_t = \text{softmax}(z_t) \tag{7b}$$

$$c_t = \sum_{i=1}^{k} \alpha_{t,i} w_t^i \tag{7c}$$

where $w_t^m \in \mathbb{R}^{D_{\text{voc}} \times k}$ is the module network outputs at time t. k denotes the number of modules in consideration. We add a new element $w_t^{\text{init}} = E y_t^{\text{init}}$ to the attention formulation. This element is the word vector of the initial estimation of the next word from the V-LSTM.

$$\hat{\alpha} = \text{softmax}\left([z_t; W_z^\top \tanh(W_i w_t^{\text{init}} + W_g h_{t-1}^s)]\right) \tag{8a}$$

$$\beta_t = \hat{\alpha}[k+1] \tag{8b}$$

$$\hat{c}_t = \beta_t w_t^{\text{init}} + (1 - \beta_t) c_t \tag{8c}$$

Depending on the context, the network composes a different set of modules to obtain word-vector $\hat{c}_t \in \mathbb{R}^{D_{\text{voc}}}$ for the S-LSTM.

The model implemented for the experiments in Section 5.6 uses a Faster-RCNN in conjunction with a Resnet-101 backbone (He et al., 2016) to segment an image into a set of regions that likely contain objects of interest and encode each region r as a fixed-length feature vector $\{v_1, \ldots, v_{D_r}\} \in \mathbb{R}^{D_v}$ where D_r is the number of regions, and D_v, the size of the feature vector. Similar to Anderson et al. (2016), the vocabulary is divided into meaningful subcategories made from an object set and five attribute sets which are color, size, count, spatial relationship, and semantic relationship. The implementation used six-word lists based on word occurrence frequency in the training data while the object set consists of visual nouns and the other attribute sets consist of adjectives (e.g, "red", "green", and "blue" for the color set). The model used five modules corresponding to color, count, size, spatial relationship and semantic relationship object attributes. All modules are implemented as two-layer fully connected networks. An example of a scene description output from an image input can be found later in Section 5.6.

4. Experimental Design

The experimental evaluation of the proposed intelligence architecture focused on different forms of studies that quantify aspects of the grounded language communication pipeline and study

how information traces through the architecture under different experimental conditions. Several component-level studies on the utility of the MMI and the performance of the language model were designed to provide insight into the runtime performance and capabilities on the resulting system. The language study evaluated the performance of the DCG for understanding and generating language. The MMI study was designed as a human subjects experiment at a urban terrain facility to evaluate the usability of the MMI for completing tasks with a robotic teammate. The study investigated the performance of the speech processing capabilities designed to support exchange of spoken commands and outcomes of a small-scale field evaluation of the MMI. This study involved two human subjects and explored twenty-four spoken commands tested in naturalistic settings featuring audial artifacts such as background speech and ambient outdoor noise. Several system-level experiments on the entirety of the architecture were designed to investigate whether the system was capable of handling scenarios that required more complex interactions in the robot intelligence architecture. First, a series of statements and instructions are provided to a mobile manipulator to study the information routing in activities related to deliberative interactive estimation. In this experiment information is routed back to the operator when an inconsistency is observed with a statement concerning declared knowledge. Second, high-level guidance is provided to a separate mobile manipulator that is sequenced by mission and manipulation planning for the purpose of clearing a piece of debris from a route. Lastly, a series of commands involving spatial navigation and scene description is performed to evaluate the robot's ability to navigate to regions in reference to objects grounded by their spatial arrangement and generate reasonable descriptions of observed imagery. In each of these the intelligence architecture from Figure 1 is annotated with how information was routed through the modules in each of these experiments.

The language model for the experiments used a synthetic corpus consisting of 964 annotated instructions that mimic the speech patterns observed in earlier published studies on natural language understanding (Paul et al., 2016, 2018). The corpus consisted of aligned instructions, parse trees, environment models, and symbols corresponding to each phrase in the sentence. These symbols represented concepts required for navigation, manipulation, and mobile manipulation that the robots could perform in environments and scenarios similar to the ones explored in these experiments. For example, the instruction "drive to the leftmost barrel" described earlier in Sections 3.2.1 and 3.2.2 would result in a root-level symbol for an action of type "navigate" with a goal defined by a uniquely labeled object with a semantic class of "barrel" that is resolved by the object's spatial relationship to other barrels and the robot.

Field experiments were performed at two different locations with urban environmental features (buildings, doors, roads, etc.) with three different mobile robot platforms. The three platforms utilized in these experiments are shown in Figure 8. The first platform, a Clearpath Robotics Husky A200 Unmanned Ground Vehicle, was used in the multimodal interface evaluation and spatial navigation and scene description experiments. The second platform, a Clearpath Robotics Husky A200 mobile manipulator outfitted with a Universal Robotics UR5 manipulator, Robotiq FT300 force/torque sensor, and a three-fingered Robotiq Adaptive Robot Gripper, was used for the experiments involving deliberative interactive estimation. The third platform, the RoMan mobile manipulator (Kessens et al., 2020), is used in the language guided mobile manipulation experiment. All three robots shared a common intelligence architecture and language model, differences appear only at the level of implementing navigation and manipulation behaviors based on the physical and hardware interface differences between platforms.

5. Experimental Results

To evaluate the performance of the architecture, we performed a series of experiments that quantify the functionality of specific components and exhibit specific behaviors enabled by the proposed architecture. A quantitative evaluation of the natural language understanding model considered by this framework measures the accuracy and runtime of two different variations of the DCGs applied in this framework using an approach similar to the one explored in Paul et al. (2018). A small

(a) Clearpath Robotics Husky A200 Unmanned Ground Vehicle

(b) Clearpath Robotics Husky A200 mobile manipulator

(c) RoMan mobile manipulator

Figure 8. The three field robots used in the experimental evaluation of the proposed robot intelligence architecture. All three of these platforms used the same instance of the developed intelligence architecture, differing only at the layer of implementation of navigation and manipulation behaviors because of physical and hardware interface differences between platforms.

user study of the MMI explores user sentiment of two study participants while also demonstrating a sequence of examples that the system was capable of performing. This is followed with four component-level demonstrations involving spatial navigation and scene description, deliberative interactive estimation, language-guided mobile manipulation, and disambiguation of ambiguous instructions that show the different information paths through the architecture.

5.1. Language Model Quantitative Evalution

The nature of the symbolic representation utilized throughout this program is different than that of corpora explored in previous model-focused quantitative evaluations (Paul et al., 2016, 2018). Those previous corpora included spatial semantic concepts representing abstract spatial concepts for collections of objects, such as rows of chairs of columns of blocks; the resulting computational complexity of reasoning over the powerset of combinations of objects was a core motivation for the development of the Adaptive Distributed Correspondence Graph formulation and model described in Section 3.2.2. While the intelligence architecture is capable of handling such abstract spatial concepts, the corpus used in these experiments did not include examples that require these symbols.

However, the Adaptive Distributed Correspondence Graph formulation of a discrete set of abstract symbols is usefully applicable to other kinds of semantic concepts that are similarly conditioned on the expression of "concrete" grounded symbols. As described in Section 3.2.2, the space of symbols representing the meaning of ambiguous utterances for object or action references can be dynamically constructed during inference according to the expression of such concrete symbols. For emphasis, the key similarity between these abstract symbols for ambiguity and the abstract symbols for collections of objects is that the set of possible symbols can be approximated to a reduced set resulting in improved runtime performance. A key difference, however, is the relationship between the size of full abstract space and the number and type of objects of in the world; the collections of objects grow as the powerset of objects whereas the symbols for ambiguity grow linearly with object type. Due to this difference in the nature of symbols, it is interesting to perform a quantitative evaluation of the difference in learned model performance (DCG vs ADCG) from the corpora used in these experiments for both accuracy and runtime in the same way as that performed by Paul et al. (2016) and Paul et al. (2018).

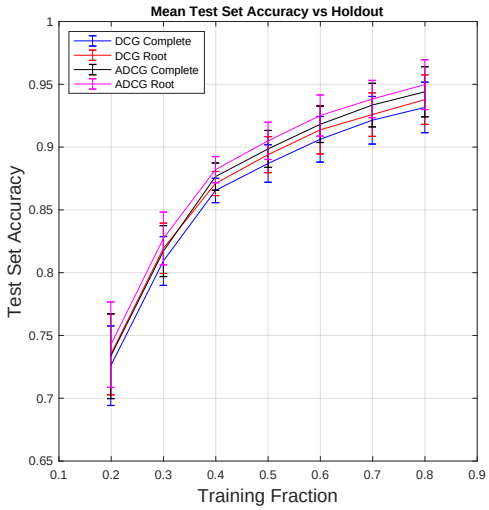
Table 4. Inference runtime (seconds) comparison

Objects	Instructions	Worlds	DCG	ADCG
0–9	872	242	4.4595 ± 2.0069	0.4649 ± 0.2662
10–19	49	28	4.7059 ± 2.0050	1.0131 ± 0.4223
20–29	20	3	4.5012 ± 2.4354	1.3574 ± 0.6943
30–39	8	7	5.3966 ± 1.9183	2.3253 ± 0.8317
40–49	2	2	6.7120 ± 2.2977	3.5078 ± 1.3276
50–59	9	7	7.8075 ± 2.5559	4.3373 ± 1.4423
60–69	2	1	9.1473 ± 0.5611	5.3368 ± 0.4257
70–79	1	1	4.7351 ± 0.0000	3.1193 ± 0.0000
80–89	—	—	—	—
90–99	1	1	5.7587 ± 0.0000	4.1047 ± 0.0000

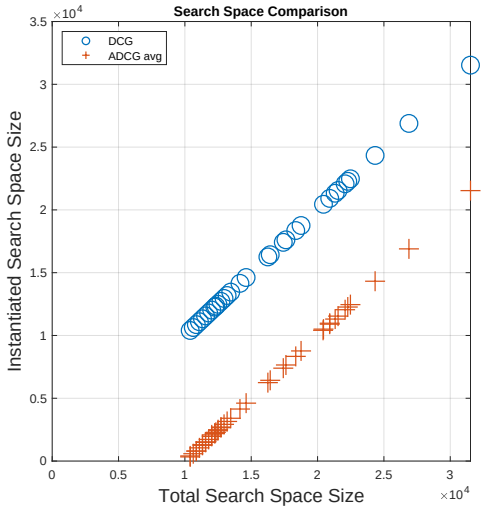
The quantitative evaluation consisted of first training log-linear models using incrementally increasing partitions of the corpus as a training dataset and then recording the accuracy and runtime performance of the models on the remaining corpus examples as a validation set. The training dataset partitions ranged from 20% of the corpus to 80% of the corpus, increasing in increments of 10%. For each partition size, 10 randomly sampled sets of examples were created and used as a novel training set for a trial; as such, the evaluation consisted of 70 different log-linear models each trained using a different training set and evaluated on the specific corresponding held-out validation set.

Figure 9 illustrates results from the quantitative experiments of the language model studied in this paper. The average accuracy of both the DCG and ADCG for incrementally increasing training set partitions is shown in Figure 9(a). This plot compares the average performance of the DCG and the ADCG with respect to two accuracy measures (root phrase and complete parse tree) on validation datasets for incrementally increasing training set partition sizes as described. We use two measures of accuracy, "root phrase accuracy" and "complete parse accuracy", to better facilitate the constituency parse tree representation of language used in these models. "Root phrase accuracy" compares only the inferred symbol set of the root phrase in the tree to the ground truth; any mislabeled phrases lower in the tree are ignored. The "root phrase accuracy" is pragmatically interesting because the inferred solution for the root is typically the information used by the intelligence architecture, making the system robust to errors that may be due to the conditional independence assumptions of the language models. "Complete parse accuracy" compares the inferred symbol set of each phrase in the tree to the ground truth; if any phrases are mislabeled, it is considered incorrect. The "complete parse accuracy" is a more demanding measure of accuracy and provides further insight into the models' performance than the "root phrase accuracy". Overall, the DCG and ADCG performance on both measures of accuracy are very similar and well within the error bars of each other; this is consistent with our expectations and the results of previous model evaluations performed by Paul et al. (2016) and Paul et al. (2018). The runtime performance of these language models is predominantly a function of the size of the space of symbols being searched over. In the case of DCG, the search space consists exhaustively of every possible symbol; in the case of ADCG, a reduced search space is dynamically constructed according to expressed concrete symbols. Therefore, it is interesting to plot the size of the complete search space against the instantiated search space, or the actual space searched during inference. Figure 9(b) plots the size of the instantiated search space for both DCG and ADCG as a function of the total search space size.

While the search space size is useful for understanding a difference in the models' performance, it is important to also report the actual corresponding runtimes. The expectation from the results in Figure 9(b) is that ADCG will have consistently lower runtimes than DCG. The difference in runtime performance between the two models should exhibit a similar kind of offset as seen in the instantiated search space sizes, and this difference should approximately reflect the runtime cost of searching over those additional symbols. Table 4 shows the runtime performance for 964 instructions across 293 unique worlds.



(a) The root phrase accuracy and complete parse tree accuracy of DCG and ADCG language models on validations for varying training set partition sizes.



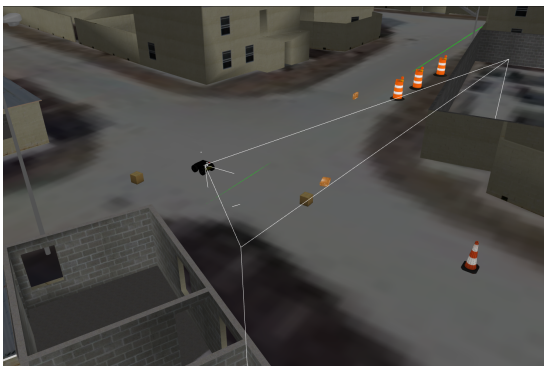
(b) Plot comparing the total size of the search space to the instantiated search space, or space used during inference, for both DCG and ADCG.

Figure 9. A comparison of accuracy and search space sizes for the quantitative evaluation of DCG and ADCG models. The plot on the left shows the root and complete parse tree accuracy of the DCG and ADCG models were shown to be comparable for varying sizes of training fractions, showing that the approximations of the symbolic representations for ADCG implemented to improve the efficiency of probabilistic inference do not have a significant negative impact the accuracy of probabilistic inference. The plot on the right shows the size of the search spaces considered by DCG and ADCG during these experiments. The search space used by DCG during inference is the complete space. The search space used by ADCG during inference is reduced according to expressed concrete symbols. Due to the nature of the abstract symbols used for ambiguity, the ADCG instantiated search space trends with effectively the same slope as the complete search space but with a constant offset.

5.2. Language Generation for Disambiguation

As initially described in Section 3.2.4, the natural language generation module in the intelligence architecture is capable of addressing situations in which an utterance is underspecified in such a way as to be ambiguous. For example, consider the scenario shown in Figure 12(a) in which there are a variety of objects in the world including three cones in front of the robot. The command "drive to the cone" is underspecified in that it is ambiguous which of the three cones is being referenced; note, however, that other semantic aspects of the command are not ambiguous, namely it is a navigation command, as opposed to a report or follow command, and the reference object is a cone, as opposed to a barrel or crate object that are also in the scene. Because the reference object is underspecified, the robot cannot complete the intended action. In situations like this, the natural language understanding module will ground to a set of symbols representing the unambiguous semantic information before prompting the natural language generation module to pose a clarifying question to the operator.

A simple solution could consist of identifying the ambiguity and asking a generic clarifying question, such as "what object did you say?". Instead, the natural language generation module uses the proactive symbol grounding module to help invert the language model and generate the most likely phrase associated with each candidate object. The candidate objects are found by selecting each object in the environment that matches the available unambiguous semantic information, e.g., that the object is a cone. Once the best phrase for N candidate objects is found, a question template "Did you mean {phrase for candidate 1}, {phrase for candidate 2}, ..., {phrase for candidate N}" is



(a) A Husky and environment using Gazebo. The world contains one construction cone, two cardboard boxes, two wooden crates, three construction barrels, several street-lamps, and a variety of buildings with doors and windows. No command has been given.

(b) The world state after the robot completed its goal. The initial command "drive to the cone" was unambiguous because there is a single cone in the world. The natural language understanding module produced the illustrated and achieved goal state.

Figure 10. The baseline for grounded language communication experiment where there is only one candidate object of the referenced type.

populated and published to the multimodal interface to provide the operator with multiple options to choose from. The operator can merely select or speak the phrase matching the intended object rather than repeat the full instruction; the natural language understanding module will associate the new input with the context of the previous utterance.

This capability of dialogue interaction for disambiguation was demonstrated both in simulation and in the field. For a baseline, a scenario where no disambiguation is required was considered. This environment, which consisted of one cone and multiple other objects, is illustrated in Figure 10(a). The system grounded the instruction "drive to the cone" in 0.14 seconds with the ADCG model to an unambiguous action symbol with the single cone object illustrated in this figure as the goal object. After the natural language understanding module finished grounding this instruction, it passed the inferred symbol to a mission planner to plan a sequence of actions to navigate to this object. The mission executor and navigation planner then planned and executed this action, resulting in the final state of the simulated system shown in Figure 10(b).

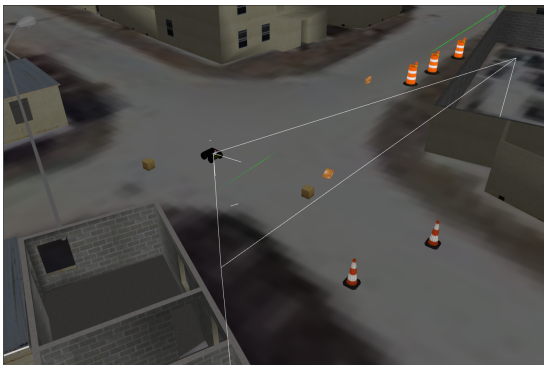
To test out the system's capability to resolve an ambiguous instruction, an additional cone was inserted into the simulation environment as illustrated in Figures 11(a) and 11(c). The same command used in the baseline, "drive to the cone", was again provided as an input into the system architecture and parsed accordingly. Two disambiguating experimental trials were performed in which the operator's clarifying response refers to a different cone. Using the ADCG model, the natural language understanding module grounded to an ambiguous symbol indicating that a "navigate" action was received for an object with "cone" object type, taking 0.16 seconds for the first trial and 0.17 seconds for the second trial. The presence of the ambiguous symbol directed the architecture to use natural language generation to resolve the ambiguous instruction. In both cases, the natural language generation module determined the candidate objects, found the most likely phrase associated with each, and published a query to the human operator.

The initial state of the first experimental trial is shown in Figure 11(a). The process for finding the candidate objects took 0.003 seconds and correctly identified the two cones in front of the robot as the candidate cone objects. The natural language generation module searched 551 phrases in 174.855 seconds and generated unique descriptions for each of the two cones in the scene. The most likely phrase for the cone that is furthest to the robot's left was "the far cone" and the most likely phrase for the cone on the robot's right was "the right cone"; these results are sufficiently disambiguating from the robot's perspective of the world. The system asked of the operator, "Did you mean the far

cone or the right cone?" The operator replied, "the far cone", resulting in a grounded navigation action with the referenced object as the navigation goal as illustrated in Figure 11(b). The initial state of the second trial is shown in Figure 11(c). The natural language generation module found the candidate objects in 0.003 seconds before again searching 551 phrases in 186.026 seconds. The most likely expressions "the left cone" and "the rightmost cone" were found for the objects that were previously referred to as "the far cone" and "the right cone" respectively from the first experimental trial. These expressions are again consistent with the environment model from the robot's perspective and help illustrate the diversity of expressions that the natural language generation could generate with the grammar used. The system asked of the operator, "Did you mean the left cone or the rightmost cone?" The operator provided the response "the rightmost cone", resulting in an action symbol with that uniquely defined object as the navigation goal as shown in Figure 11(d). These two experimental trials interestingly demonstrated ways in which the natural language generation module can produce different but effective disambiguating natural language descriptions of the same objects. While the system emits the most likely phrases for each candidate object, we observed a number of uniquely identifying phrases for objects that had high likelihoods; the variation in best phrases can be attributed to slight differences in the initial position and orientation of the robot in each of these simulated experiments.

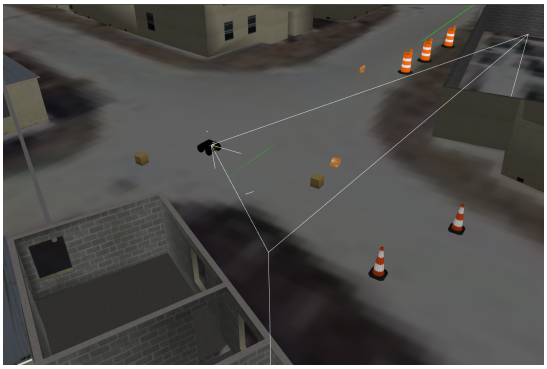
One last example involving three cones in the gazebo simulation environment was performed, as seen in Figures 12(a) and 12(b). In this scenario, the robot's environment model includes three construction cones in front of the robot, two cardboard boxes, two wooden crates, three construction barrels to its far left, several streetlamp posts, and a variety of buildings with doors and windows. Like the previous scenarios, the initial provided utterance was "drive to the cone" and was correctly associated with a symbol representing ambiguity while preserving the unambiguous semantic content of a "navigate" action and a "cone" object type; this process took 0.18 seconds with the ADCG model. Upon grounding to an ambiguous symbol, the natural language understanding module triggered the natural language generation module to determine the candidate objects, find the most likely phrase associated with each, and publish a query to the human operator. The process for finding the candidate objects took 0.005 seconds and correctly identified the three cones in front of the robot as the candidates. The inverse semantics process iterated through 551 generated phrases and took 205.42 seconds with no idle system time for the proactive symbol grounding module to provide bootstrapping. The slightly longer runtime of natural language generation with respect to the two cone examples explored in Figure 11 can be attributed to the larger symbolic representation that resulted from the addition of a cone object. The resulting best phrases generated for the candidates were "the leftmost cone", "the middle cone", and "the rightmost cone"; these phrases were used to ask the user "Did you mean the leftmost cone, the middle cone, or the rightmost cone?" The user responded, "the rightmost cone." The natural language understanding module correctly grounded the phrase "the rightmost cone" in the context of the prior utterance to produce a "navigation" action with the rightmost cone as the navigation goal.

Another disambiguation example scenario from a series of field demonstrations can be seen in Figure 13; this scenario was the first vignette in a continuous sequence of language-driven behaviors. This environment was set up with bicycles, motor bikes, fruit stands, gas tanks, tables, barrels, and other types of objects. For this vignette, there are three barrels in front of the robot's initial starting position. A human teammate gave the command "go to the barrel" which was correctly associated with a symbol representing ambiguity while preserving the unambiguous semantic content of a "navigate" action and a "barrel" object type. The natural language generation module correctly identified the three barrels as candidate objects before triggering the proactive symbol grounding module's inverse semantics process to find the most likely phrases "the leftmost barrel," "the barrel in the middle" and "the right barrel." Given these phrases, the language generation module queried the human "Did you mean the leftmost barrel, the barrel in the middle, or the right barrel?" The human responded "the leftmost barrel" which was correctly grounded in the context of the initial ambiguous command to produce an action symbol indicating a navigation action with the leftmost barrel as a goal. Once completed, the human-robot team continued with rest of the demonstration.



(a) A Husky and environment using Gazebo. The world contains two construction cones, two cardboard boxes, two wooden crates, three construction barrels, several street-lamps, and a variety of buildings with doors and windows. No command has been given.

(b) The world state after the robot completed its goal. The initial command "Drive to the cone" was ambiguous. The inverse semantics process generated the disambiguating question "Did you mean the far cone or the right cone?". The reply "the far cone" produced the illustrated and achieved goal state.



(c) A Husky and environment using Gazebo. The world contains two construction cones, two cardboard boxes, two wooden crates, three construction barrels, several street-lamps, and a variety of buildings with doors and windows. No command has been given.

(d) The world state after the robot completed its goal. The initial command "Drive to the cone" was ambiguous. The inverse semantics process generated the disambiguating question "Did you mean the left cone or the rightmost cone?". The reply "the rightmost cone" produced the illustrated and achieved goal state.

Figure 11. A grounded language communication experiment involving disambiguation with two candidate objects of the referenced type.

5.3. Multimodal Interface Evaluation

To study the user experience with the MMI a scenario was designed that would involve language interaction with a field robot and a questionnaire to be completed after performing the tasks. After agreeing to participate in the study, participants were given a brief training session on the MMI that was intended to familiarize them with the operation of the device and what they should expect from the commands issued to their robotic teammate. Participants would interact with the robot and MMI in a manner similar to that illustrated in Figure 2 where the operator can observe both the robot and the MMI, give voice commands to the MMI, and receive feedback from the MMI. An illustration of the sequence of tasks that the robot would need to perform is shown in Figure 14.

The protocol involved the human providing seven instructions through spoken language prompted by actions that the robot would perform in the environment. The first command involved telling the robot to "drive to the nearest barrel" which would cause the robot to interpret the scene from current

(a) A Husky and environment using Gazebo. The world contains three construction cones, two cardboard boxes, two wooden crates, three construction barrels, several street-lamps, and a variety of buildings with doors and windows. No command has been given.

(b) The world state after the robot completed its goal. The initial command "Drive to the cone" was ambiguous. The inverse semantics process generated the disambiguating question "Did you mean the leftmost cone, the middle cone, or the rightmost cone?". The reply "the rightmost cone" produced the illustrated and achieved goal state.

Figure 12. A grounded language communication experiment involving disambiguation with three candidate objects of the referenced type.

Figure 13. A robot field experiment scenario of a street market with various realistic entities, such as bicycles, shops, motor bikes, fruit stands, among others. The perspective is captured from a camera mounted on the front of a Clearpath Husky A200 Unmanned Ground Vehicle. Prominently displayed are three barrels directly in front of the robot.

sensor observations and then drive to the specified barrel and halt. The operator was then guided to "drive behind the left barrel" which again required that the robot interpret the meaning of the instruction in the context of the environment and stop after navigating behind the object referenced by the instruction. The third task involved asking the robot to "report", which is interpreted by the architecture as a behavior that calls the scene description module for a natural language description of the current scene. After the description is received and displayed on the MMI, the robot was to be told "your location is waypoint Alpha", which creates a semantic description of this metric location in the environment model and provides feedback to the operator on this behavior through the MMI. The fifth command involves telling the robot to "drive to the nearest doorway" which causes the robot to again drive to that location that it interprets as "the nearest doorway" in the map and stop after it reaches that location. The human teammate was then told to walk over to the robot and position the robot in front of the main camera and issue the command "follow me". The human teammate was then told to walk down the main avenue of the urban test site approximately thirty feet and stop. Given the commands, the robot followed the human operator at a safe distance

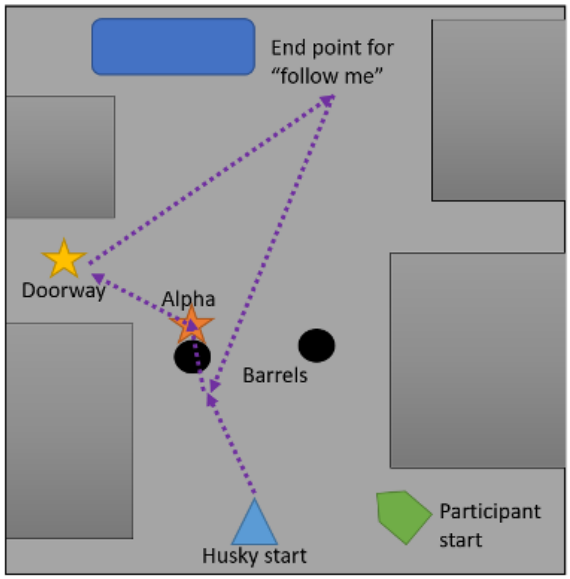


Figure 14. An illustration of the scenario used in the component level study of the MMI.

and stopped when the human operator had stopped. The last command issued to the robot was "drive to waypoint Alpha" which should cause the robot to navigate back to the human specified landmark in the environment model.

Since the study only enrolled two participants, only qualitative observations are reported. The enrolled participants were between the ages of 20 and 50, had no prior military experience, very highly familiar with robotics, and had experience with the robot platform. Study participants included RCTA personnel who were blind to study hypotheses. Generally, the participants in this experiment indicated that they had positive experiences using the MMI to interact with their robotic teammate. In particular, participants found the system to be straightforward to learn, and appreciated the modes of feedback that were provided (visual images, natural language communications, world map representation). Though the participants did not suggest that the system was very innovative or especially pleasing to use, those outcomes may be the result of prior experience with robotics and robotics interfaces. Another interpretation of this finding could be a result of the underlying design meeting user needs in such a way that it was "obvious" the interface behaves in the way it does for the given scenario. Combined with successful task completions, the positive experiences of the participants indicate that the MMI was straightforward to learn and usable for supporting human-robot teaming.

5.4. Deliberative Interactive Estimation

To evaluate the behaviors of the deliberative interactive estimation model, we devised an experiment with the Husky A200 mobile manipulator that involved declared facts and physical interaction with these objects. The flow of information through the architecture is shown in Figure 15, which begins with the blue path of text coming from the multimodal interface and objects from the semantic perception modules that are passed through the intelligence world model, parsing and declarative knowledge, proactive symbol grounding, and natural language understanding modules to interpret facts that update the robot's state of the world and missions for the robot to perform. These activities involved both navigation and deliberative interactive estimation actions. The robot tracks inconsistencies in the human's mental model of the world by observing when facts expressed by the human operator are inconsistent with the robot's own observations. When this occurs the

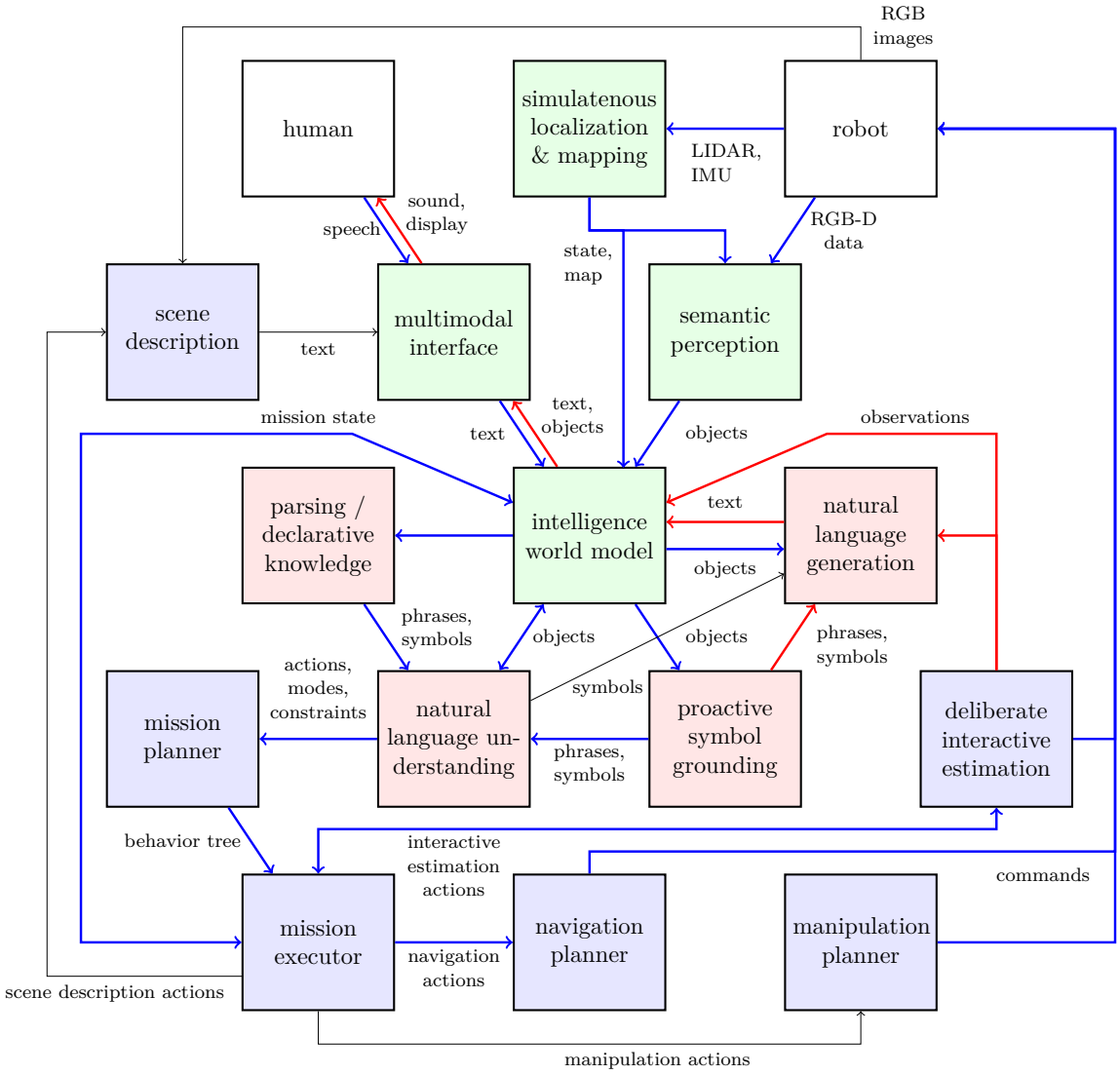


Figure 15. The dataflow for the deliberative interactive estimation experiments.

natural language generation module, which tracks these facts in the context of observations made by the deliberative interactive estimation module, to generate a unique description of the object and the observed fact by inverting the natural language understanding module. This return path of information to the human operator, shown in red, is sent through the intelligence world model to the multimodal interface to correct the human's mental model of the environment.

A series of images from this experiment are shown in Figure 16. The experiment began with a human expressing a series of facts declaring that "the leftmost barrel is full" and "the rightmost barrel is full" where "the leftmost barrel" and "the rightmost barrel" were uniquely resolved by the natural language understanding module's ability to resolve spatial relationships of objects of the same semantic class. The robot was then instructed to "push the leftmost barrel" which caused the robot to physically interact with the barrel to the robot's left. The robot deploys its manipulator and makes an observation of the "fullness" of the object by the learned model of that semantic property using information from the force/torque sensor mounted between the gripper and the arm. The robot does not believe this object to be full based on the reaction force observed at

(a) The robot is initially provided with the statements "the leftmost barrel is full" and "the rightmost barrel is full".

(b) The robot deliberatively interacts with the barrel to the robot's left given the robot instruction "push the leftmost barrel".

(c) The robot states "the fact the leftmost barrel is full is inconsistent with world model" after estimating the semantic state as being inconsistent with the world model. The robot also internally updates it's representation of that object.

(d) The robot is provided with the instruction "push the full barrel" which is now ambiguous because after interactive estimation it no longer considers "the leftmost barrel" as being full.

(e) The robot deliberatively interacts with the object originally identified as "the rightmost barrel".

(f) The robot finds that the human-provided description is consistent with the interactive measurement of the object and provides no report back to the operator at the conclusion of the action.

Figure 16. Excerpts from experiments on deliberative interactive estimation with a mobile manipulator.

the wrist from the pushing action with the arm and therefore generates a correction back to the human. In this experiment that correction came in the form of a template-based statement that prepends the tracked fact with "the fact" and appends the tracked fact with "is inconsistent with world model". The robot is then instructed "push the rightmost barrel" which caused the robot to navigate to the other barrel and perform the same deliberative interaction behavior. The robot finds that this observation is consistent with the human's declared fact and therefore does not generate a corrected fact through the natural language generation pipeline shown in Figure 15. Variations of this experiment are explored in more detail in Arkin et al. (2020) where a more natural description of the corrected fact (e.g., "heavy" vs "light" instead of "heavy is inconsistent with world model") is able to be generated by a more complete inverse semantics model used by the natural language generation module.

5.5. Language Guided Mobile Manipulation

To assess the ability of the architecture to interpret and execute a mobile manipulation command, a scenario was constructed that required a robot to remove an obstruction from a narrow passage between two buildings. To demonstrate how pithy commands could be used to sequence complex behaviors, we provided the robot the instruction "clear the debris", a two-phrase utterance that grounded "the debris" to the object placed in the middle of the two barriers as illustrated in Figure 18. This action shows the simplest flow of information through the architecture, annotated as the blue path in Figure 17, starting with the instruction parsed and sent through the intelligence world model to the parsing and declarative knowledge module to generate the parse tree that is sent to the natural language understanding module. The natural language understanding module grounds the instruction to a symbol describing a "clear" action with a grounded instance of an

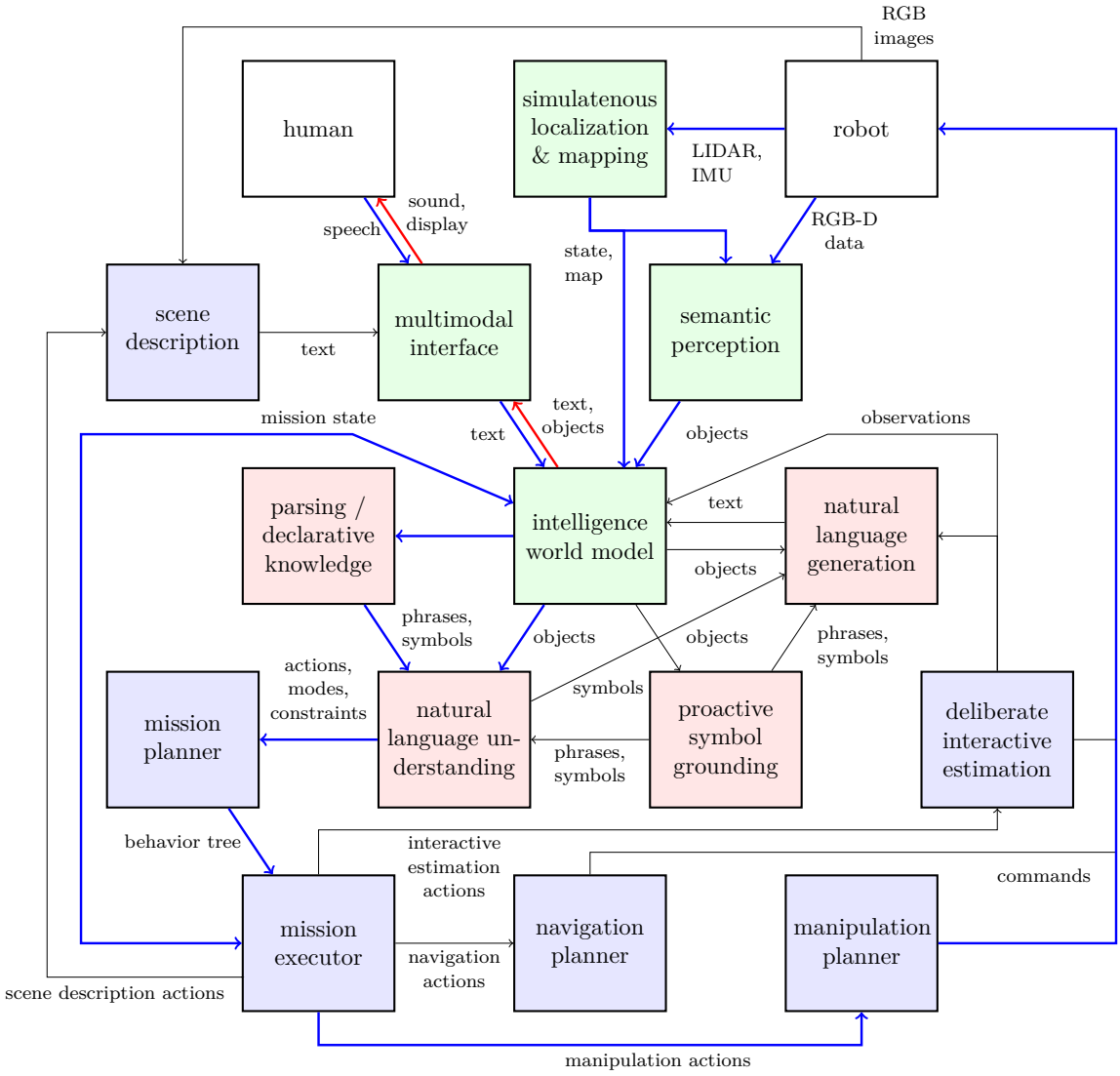


Figure 17. The dataflow for the language guided mobile manipulation experiment.

object with the semantic label "debris". The manipulation planner receives this information, plans a sequence of actions to satisfy the terminal boundary state described by this activity, and sends control inputs to the robot whose state updates are processed by the mission executor to monitor for mission completion.

A sequence of three images showing initial, intermediate, and terminal states is shown in Figure 18. In the image on the left we see a human operator giving the RoMan robot the instruction "clear the debris" in an environment model that recognizes the piece of metal truss in the lower right corner of the image as being "the debris" and other objects as barriers. The "clear" activity involves navigation to, grasping, lifting, and placing of the object in a different location. The middle image shows the approach to the debris object where a grasp is planned with the observed geometry of the object. The image on the right shows the location of the place activity in a location different than the object's initial state, ending the "clear" behavior.

(a) The robot and human operator at their initial locations with an obstacle present in the desired path of the robot

(b) The robot grasps the object to remove after given the language command "clear the debris" to the MMI

(c) The robot places the object away from the initial location to permit passage through the corridor

Figure 18. A sequence of robot motions in response to the human-provided command "clear the debris" through the MMI. The symbols inferred from this utterance in the context of the perceived environment invoke the mission planner to construct a behavior tree that directs the manipulation stack to grasp, lift, move, and place an object inferred by semantic perception to be a piece of debris far from the initial location.

5.6. Spatial Navigation and Scene Description

Experimental results of the scene description module show that our model achieves both the fluency of sequential models and the specificity of compositional models. A detailed exploration of such scene descriptions can be found in Tian and Oh (2019). Specifically, our approach excels at including fine-grained details such as counting that are generally avoided or overlooked. The framework is easily expandable to include additional functional modules of more sophisticated designs. Improved interpretability via visualized attention is another bonus because the model enables a quantitative analysis of both visual and semantic information.

To evaluate the performance of spatial navigation and scene description on a field robot, a sequence of language commands was provided to a Husky A200 unmanned ground vehicle in a visually interesting scene with objects that are not uniquely identified by a semantic class in the instance of the world model constructed from perception and the time of human-robot interaction. Excerpts from the MMI during this interaction are shown in Figure 20, which show both the MMI's representation of the world model during the course of the interaction, the robot's perspective taken from a forward-facing RGB-D camera mounted on the top of the robot, and the received instructions and reports back from the robot.

Additional spatial navigation experiments were carried out on the Husky A200 unmanned ground vehicle to demonstrate the use of natural language to switch between the system's two metric planners discussed in Section 3.3.3. Using the same field environment previously discussed for the scene description experiments, a language command was provided to the unmanned ground vehicle and the SBPL-based metric planner would begin navigating to the defined location. During the execution of the planned trajectory the language command "Go covertly" was provided to the robot, causing it to transition to the learned IOC planner that considers terrain and object perception information to execute a context-aware navigation behavior. In these experiments, the learned IOC behavior deployed on the robot planned paths near the edge of the road. When the robot was given the command "Go covertly," it transitioned to the IOC planner but maintained the originally specified waypoint destination.

Figure 21 shows examples of the Rviz display when running this navigation experiment. In the top row, the Rviz displays for the two commands can be seen. In each image the most recent natural language command provided to the system can be seen in the top left and the associated behavior tree generated from this command is shown on the right. In the bottom row, the display shows a third person view of the robot during experimentation, the robot's map shown with objects and terrain (grass is green and road is purple) from the world model, and the trajectories that were output by each of the planners given the natural language commands. The blue trajectory illustrates the path

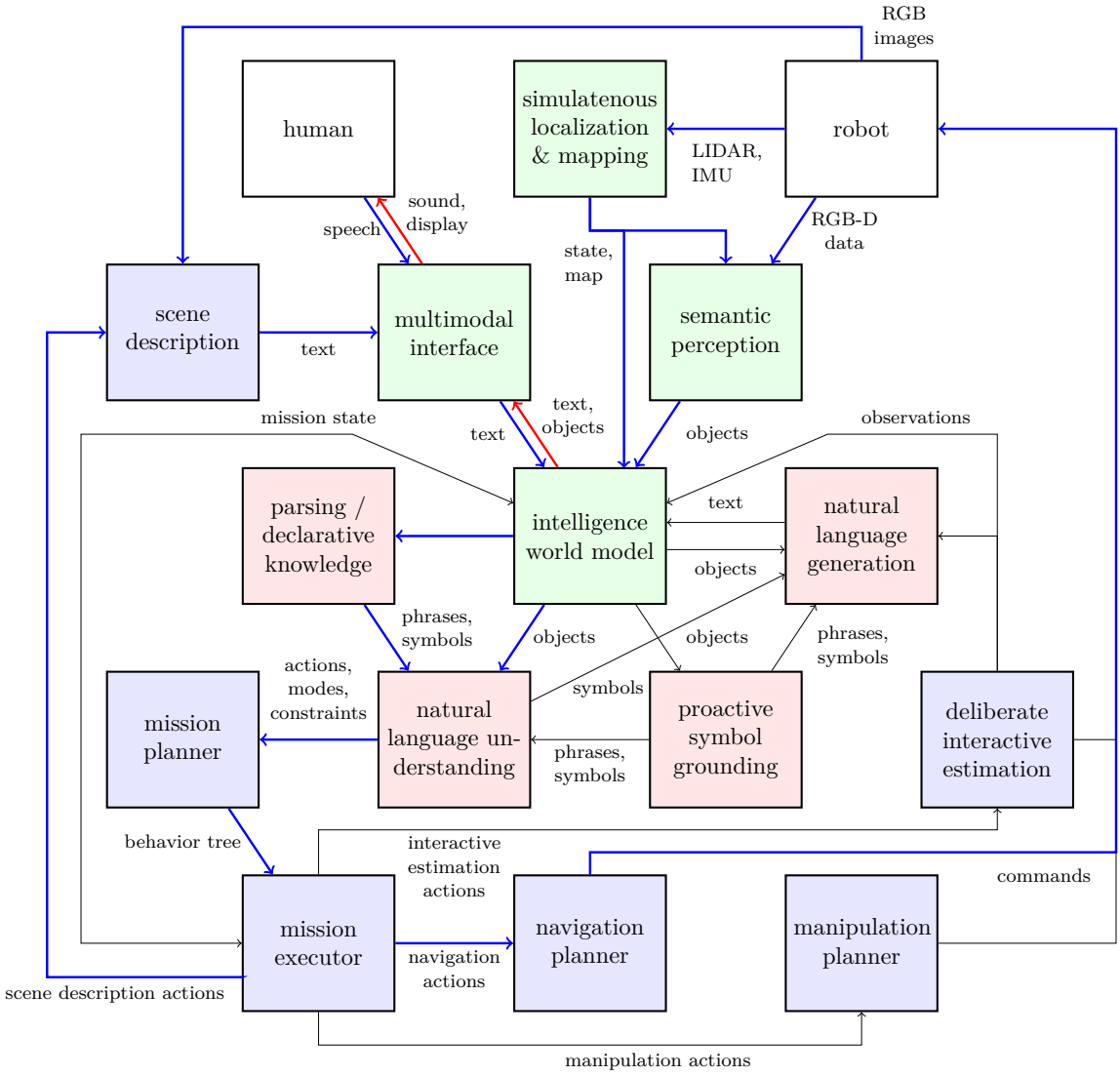
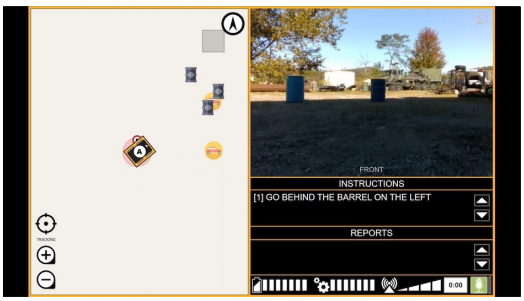


Figure 19. The dataflow for the spatial navigation and scene description experiment.

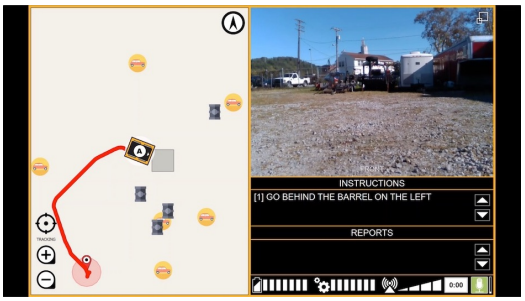
originally planned to waypoint alpha, and the green trajectory represents the terrain-aware path that was planned after given the "Go covertly" command. This experiment serves to illustrate that a human teammate can command behavior preferences for navigation to better suit mission needs. In this case, the robot may take a slightly longer path to the waypoint when operating "covertly" but does so without traversing through a highly visible open area of the environment.

6. Discussion

The presented intelligence architecture represents a novel approach to grounded language communication with a field robot. Experiments with this architecture demonstrate that it is able to declare information, ground statements to the physical environment, and resolve ambiguities or observed discrepancies between mental models held by the operator and the robot. This architecture captures visual and audial information from on-board sensors and human guidance through a multimodal interface that can also report back information about the synthesized environment model that

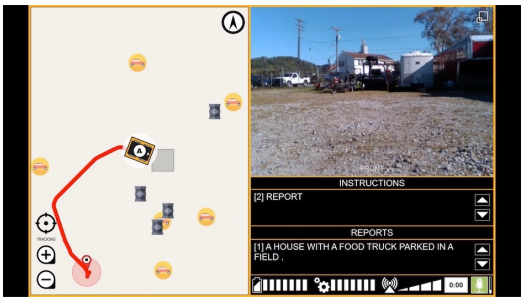


(a) The robot is initially provided with the instruction "go behind the barrel on the left"



(b) The state of the robot after planning a route to navigate to a location with a spatial relation behind the leftmost barrel front the perspective of the robot

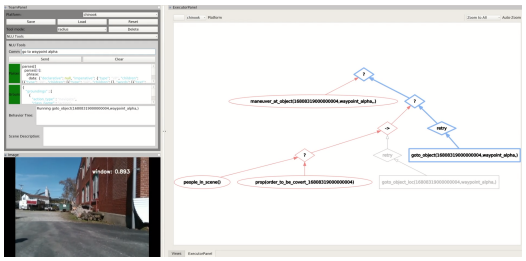
(c) The robot is provided with a new instruction "report"



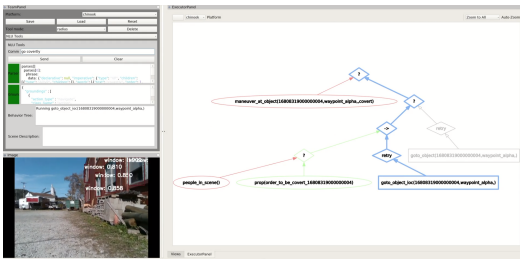
(d) The scene description action provides a response "a house with a food truck parked in a field"

Figure 20. Screen captures from the MMI for a sequence of language instructions that guide the robot through a pair of spatial navigation and scene description activities. First, the robot interprets the meaning of "go behind the barrel on the left" by inferring a navigation goal as a position behind the objected identified by "the barrel on the left" in relation to the robot's current pose. The robot then is provided the command "report", which the natural language understanding module infers as an action symbol of type scene description. The mission planner constructs a behavior tree that calls the scene description to provide publish a descriptive phrase based on the current viewpoint, which is captured by the MMI and shown to the human operator.

the robot uses to make decisions. The diversity of instructions understood by the robot using a corpus of annotated instructions, combined with a mission planning and execution layer enables the robot to perform activities beyond waypoint navigation. This enables the robot to sequence multiple, complex actions at scales that executing interdependent behaviors that are conditioned on the closed loop execution of these actions. The experiments explored in this paper highlighted or reinforced limitations that have motivated parallel lines of research in intelligent field robotics. The grounded language communication models explored in (Duvallet et al., 2014; Hemachandra et al., 2015; Patki et al., 2020) show how language can be used as a sensor to inform distributions of environment models that may be only partially observable with visual perception. Similar in some ways to the idea of declarative knowledge, spatial relationships between objects seen and unseen can be gleaned from instructions provided by the human operator to infer distributions of objects not directly represented in the environment model. If the robot in this framework were presented the instruction "navigate to the barrel behind the building" without a notion of a "barrel" in the environment model, it would not be able to ground the utterance to any object of that type and may attempt to resolve this problem by posing a question dialogue between the robot and the operator through the MMI. The alternative idea explored in this sequence of papers enables the robot to hypothesize the presence of an object in a distribution of environment models using the spatial relation between the unseen object ("barrel") in relation to the observed object ("building").



(a) The robot is given the command "go to waypoint alpha."



(b) During execution of the path to waypoint alpha the robot is given the command "go covertly".

(c) Illustration of the two distinct paths generated by the different planners when navigating to the waypoint alpha. The blue trajectory comes from the shortest path, obstacle avoidance planner, and the green trajectory is generated by the learned behavior for terrain-aware planning.

Figure 21. Screen captures from the Rviz console during experiments that tested the transition between different system planners based on natural language commands. The bottom left of each image shows the RGB camera sensor input from the robot with object detections. The top row displays the natural language command and associated behavior tree, and the bottom row illustrates the differing trajectories from each planner.

Another important limitation that we encountered during physical experiments is that tracking errors in the perception system could result in very large environment models. This is problematic because large environment models can grow the size of the symbolic representation used for symbol grounding and degrade the efficiency of language grounding. Although application of the HDCG and HADCG models would have made the system more robust to these disturbances, a re-examination of how perception and language interact in this architecture is worthy of attention. Another recent line of work (Patki and Howard, 2018; Patki et al., 2019) uses language to infer what classifiers and observations are necessary to ground a particular instruction. This approach, which selectively classifies objects based on their inferred need for interpreting the meaning of specific instructions, removes the need to completely label all semantic objects the robot may encounter at all times to generate a rich and complete world model. Similar to the way humans may selectively observe objects of interest when entering a room, the robot uses a variation of the DCG with a symbolic representation tied to the configuration of the perception system to determine the minimal but sufficient environment representation necessary for interpreting the meaning of the sentence. The

architecture described in this article uses a flat perception pipeline that consistently classifies a set number of predefined classes at a fixed rate, irrespective of whether or not the intelligence architecture needs such information to perform a sequence of activities or tasks.

We also observed that while the proactive symbol grounding approach described in this architecture does demonstrate the ability to facilitate natural language understanding and generation, it opens up new and interesting questions about the space of environments for which an inferred symbol will remain valid. Currently the proactive symbol grounding module is constantly grounding expressions, regardless of whether or not their meaning has changed based on differences in the robot's pose or observed environments in the world model. An approach that could infer not only what the most likely meaning of a symbol is, but the space of worlds for which that symbol would remain the most likely symbol, would allow us to search, store, and replace more efficiently in the space of proactively-grounded symbols. For example, if a robot takes only fifty seconds to populate a space of proactively grounded symbols but the robot and environment are stationary for twice that amount, it should not be necessary to reevaluate these symbols because the answer produced by the model is deterministic when presented with the same environment and language inputs.

Lastly, we recognize that the architecture does not use an explicit dialogue manager that is able to track the conversational ground. Other architectures, frameworks, and algorithms that focus on the dialogue aspects of spoken or written communication (Scheutz et al., 2011, 2013; Williams et al., 2016; Thomason et al., 2015, 2020) are more capable in tracking conversational dialog than our design, which only tracks the context of recent utterances for disambiguation, corrections, or sequencing of tasks. This architecture is also not currently capable of handling inputs from multiple robot operators that may be needed in a human-robot team composed of multiple robots and operators. Integration of the presented ideas involving efficient and effective symbol grounding and generalization across multiple platforms in unstructured environments with a dialogue manager that exploits richer linguistic representations (Bonial et al., 2020) may further advance this architecture to one that is capable of such interactions.

7. Conclusion

This article has explored a unique robot-intelligence architecture capable of bidirectional, grounded language communication with field robots. The framework, developed and evolved during the course of the Army Research Laboratory's Robotics Collaborative Technology Alliance program, fuses information from both on-board sensors and human guidance to interpret instructions that can define complex missions and individual actions. The architecture uses unique capabilities for proactive symbol grounding, natural language generation, and deliberative interactive estimation. These components provide the system with the ability to ask questions to resolve ambiguities in human-provided statements and make corrections to imperfections in the human's mental model of the world. The architecture maintains an environment model that describes the metric and semantic properties of objects and which is used pervasively by modules for language grounding, mission planning, and motion planning. Component and system-level experiments of the same architecture on three different field robots demonstrates the utility of these abstractions that permit robot-specific implementations of general actions while exploiting shared models for mission planning and grounded language communication. The integrated experiments described in this paper demonstrate the progress made in grounded language communication over the past decade that enables manned-unmanned teaming and the transition of robots from tools to teammates. Ongoing and future work centers on the expansion of these capabilities in partially-observed and dynamic environments and more intricate tasks that require coordination between multiple humans and robots. Additional research directions also include models capable of acquiring and reasoning with common sense, exploiting physical knowledge (e.g., tool use and affordances), richer modeling of human intent (as plans or prior beliefs), and addressing the nuances and challenges of implicit and situated human-robot communication in dialogue.

Acknowledgment

The research reported in this document was performed in connection with Contract Number W911NF-10-2-0016 with the U.S. Army Research Laboratory. The views and conclusions contained in this document are those of the authors and should not be interpreted as presenting the official policies or position, either expressed or implied, of the U.S. Army Research Laboratory, or the U.S. Government unless so designated by other authorized documents. Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

ORCID

Thomas M. Howard https://orcid.org/0000-0002-8270-0806
Ethan Stump https://orcid.org/0000-0002-0119-2169
Jonathan Fink https://orcid.org/0000-0002-1834-2442
Jacob Arkin https://orcid.org/0000-0002-1074-9248
Rohan Paul https://orcid.org/0000-0001-9082-0376
Daehyung Park https://orcid.org/0000-0002-1287-9433
Daniel Barber https://orcid.org/0000-0002-9263-3109
Karl Schmeckpeper https://orcid.org/0000-0003-4989-2022
Jean Oh https://orcid.org/0000-0001-9709-2658
Maggie Wigness https://orcid.org/0000-0003-1707-8106
Brandon Rothrock https://orcid.org/0000-0003-2237-6589
Jeremy Nash https://orcid.org/0000-0002-1854-8504
Matthew R. Walter https://orcid.org/0000-0003-1425-6050
Florian Jentsch https://orcid.org/0000-0001-9879-7764
Nicholas Roy https://orcid.org/0000-0002-8293-0492

References

Anderson, P., Fernando, B., Johnson, M., and Gould, S. (2016). SPICE: Semantic propositional image caption evaluation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 382–398.

Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. (2016). Learning to compose neural networks for question answering. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies (NAACL HLT)*, pages 1545–1554. Association for Computational Linguistics.

Arkin, J., Park, D., Roy, S., Walter, M. R., Roy, N., Howard, T. M., and Paul, R. (2020). Multimodal estimation and communication of latent semantic knowledge for robust execution of robot instructions. *International Journal of Robotics Research*.

Arkin, J., Walter, M. R., Boteanu, A., Napoli, M., Biggie, H., Kress-Gazit, H., and Howard, T. M. (2017). Contextual awareness: Understanding monologic natural language instructions for autonomous robots. In *IEEE International Symposium on Robot and Human Interactive Communication*.

Arvidson, R. E., Bell, J. F., Bellutta, P., Cabrol, N. A., Catalano, J. G., Cohen, J., Crumpler, L. S., Des Marais, D. J., Estlin, T. A., Farrand, W. H., Gellert, R., Grant, J. A., Greenberger, R. N., Guinness, E. A., Herkenhoff, K. E., Herman, J. A., Iagnemma, K. D., Johnson, J. R., Klingelhofer, G., Li, R., Lichtenberg, K. A., Maxwell, S. A., Ming, D. W., Morris, R. V., Rice, M. S., Ruff, S. W., Shaw, A., Siebach, K. L., de Souza, P. A., Stroupe, A. W., Squyres, S. W., Sullivan, R. J., Talley, K. P., Townsend, J. A., Wang, A., Wright, J. R., and Yen, A. S. (2010). Spirit Mars Rover Mission: Overview and selected results from the northern Home Plate Winter Haven to the side of Scamander crater. *Journal of Geophysical Research: Planets*, 115(E7).

Bacha, A., Bauman, C., Faruque, R., Fleming, M., Terwelp, C., Reinholtz, C., Hong, D., Wicks, A., Alberi, T., Anderson, D., Cacciola, S., Currier, P., Dalton, A., Farmer, J., Hurdus, J., Kimmel, S., King, P.,

Taylor, A., Covern, D. V., and Webster, M. (2008). Odin: Team VictorTango's entry in the DARPA Urban Challenge. *Journal of Field Robotics*, 25(8):467–492.

Ballard, R. D. (1993). The MEDEA/JASON remotely operated vehicle system. *Deep Sea Research Part I: Oceanographic Research Papers*, 40(8):1673–1687.

Barber, D., Abich, J. A., Phillips, E., Talone, A., Jentsch, F., and Hill, S. (2015a). Field assessment of multimodal communication for dismounted human-robot teams. In *Proceedings of the Annual Meeting of the Human Factors and Ergonomics Society Annual Meeting*, pages 921–925.

Barber, D., Howard, T. M., and Walter, M. R. (2016). A multimodal interface for real-time soldier-robot teaming. In *Proceedings of SPIE Defense and Security, Unmanned Systems Technology XVIII*, volume 9837.

Barber, D., Wohleber, R. W., Parchment, A., Jentsch, F., and Elliott, L. (2014). Development of a squad level vocabulary for human-robot interaction. In Shumaker, R. and Lackey, S., editors, *Virtual, Augmented and Mixed Reality. Designing and Developing Virtual and Augmented Environments*, pages 139–148.

Barber, D. J., Reinerman-Jones, L. E., and Matthews, G. (2015b). Toward a tactile language for human–robot interaction: Two studies of tacton learning and performance. *Human Factors*, 57(3):471–490.

Bhattacharjee, T., Grice, P. M., Kapusta, A., Killpack, M. D., Park, D., and Kemp, C. C. (2014). A robotic system for reaching in dense clutter that integrates model predictive control, learning, haptic mapping, and planning. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) Workshop on Robots in Clutter: Perception and Interaction in Clutter*.

Bischoff, R. and Graefe, V. (2002). Dependable multimodal communication and interaction with robotic assistants. In *Proceedings. 11th IEEE International Workshop on Robot and Human Interactive Communication*, pages 300–305.

Bohren, J., Foote, T., Keller, J., Kushleyev, A., Lee, D., Stewart, A., Vernaza, P., Derenick, J., Spletzer, J., and Satterfield, B. (2008). Little Ben: The Ben Franklin Racing Team's entry in the 2007 DARPA Urban Challenge. *Journal of Field Robotics*, 25(9):598–614.

Bonial, C., Donatelli, L., Abrams, M., Lukin, S. M., Tratz, S., Marge, S., Artstein, R., Traum, D., and Voss, C. (2020). Dialogue-AMR: Abstract Meaning Representation for dialogue. In *Proceedings of the Language Resources and Evaluation Conference*, pages 684–695, Marseille, France.

Boteanu, A., Arkin, J., Howard, T. M., and Kress-Gazit, H. (2016). A model for verifiable grounding and execution of complex language instructions. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2649–2654. IEEE.

Boteanu, A., Arkin, J., Patki, S., Howard, T. M., and Kress-Gazit, H. (2017). Robot-initiated specification repair through grounded language interaction. In *AAAI Fall Symposium on Natural Communication for Human-Robot Collaboration*.

Boularias, A., Duvallet, F., Oh, J., and Stentz, A. (2015). Grounding spatial relations for outdoor robot navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Bowen, A. D., Yoerger, D. R., Taylor, C., McCabe, R., Howland, J., Gomez-Ibanez, D., Kinsey, J. C., Heintz, M., McDonald, G., Peters, D. B., Fletcher, B., C., Y., Buescher, J., Whitcomb, L. L., Martin, S. C., Webster, S. E., and Jakuba, M. V. (2008). The Nereus hybrid underwater robotic vehicle for global ocean science operations to 11,000 m depth. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*.

Broad, A., Arkin, J., Ratliff, N., Howard, T. M., and Argall, B. (2017). Real-time natural language corrections for assistive robotic manipulators. *International Journal of Robotics Research*, 36(5-7):684–698.

Buehler, M., Iagnemma, K., and Singh, S. (2009). *The DARPA urban challenge: autonomous vehicles in city traffic*, volume 56. springer.

Camilli, R., Reddy, C. M., Yoerger, D. R., Van Mooy, B., Albert, S., Jakuba, M. V., Kinsey, J. C., McIntyre, C. P., Sylva, S. P., and Maloney, J. V. (2010). Tracking hydrocarbon plume transport and biodegradation at Deepwater Horizon. *Science*, 330(6001):201–204.

Chitta, S., Sucan, I., and Cousins, S. (2012). Moveit! *IEEE Robotics & Automation Magazine*, 19(1):18–19.

Chung, I., Propp, O., Walter, M. R., and Howard, T. M. (2015). On the performance of hierarchical distributed correspondence graphs for efficient symbol grounding of robot instructions. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Colledanchise, M. and Ögren, P. (2018). *Behavior trees in robotics and AI: An introduction*. CRC Press.

Dean, R., Oh, J., and Vinokurov, J. (2014). Common world model for unmanned systems: Phase 2. In Karlsen, R. E., Gage, D. W., Shoemaker, C. M., and Gerhart, G. R., editors, *Unmanned Systems Technology XVI*, volume 9084, pages 169–176. International Society for Optics and Photonics.

Detry, R., Papon, J., and Matthies, L. (2017). Task-oriented grasping with semantic and geometric scene understanding. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Duff, E. S., Roberts, J. M., and Corke, P. I. (2003). Automation of an underground mining vehicle using reactive navigation and opportunistic localization. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3775–3780.

Durrant-Whyte, H. (1996). An autonomous guided vehicle for cargo handling applications. *International Journal of Robotics Research*, 15(5):407–440.

Duvallet, F., Walter, M. R., Howard, T. M., Hemachandra, S., Oh, J., Teller, S., Roy, N., and Stentz, A. (2014). Inferring maps and behaviors from natural language instructions. In *Proceedings of the International Symposium on Experimental Robotics (ISER)*.

Esponda, M. and Howard, T. M. (2018). Adaptive grasp control through multi-modal interactions for assistive prosthetic devices. In *Proceedings of the National Conference on Artificial Intelligence (AAAI) Fall Symposium Series on Artificial Intelligence for Human-Robot Interaction*.

Fong, T. and Thorpe, C. (2001). Vehicle teleoperation interfaces. *Autonomous Robots*, 11(1):9–18.

Fong, T., Thorpe, C., and Glass, B. (2003). PdaDriver: A handheld system for remote driving. In *Proceedings of the IEEE International Conference on Advanced Robotics*, Coimbra, Portugal.

Furgale, P. and Barfoot, T. D. (2010). Visual teach and repeat for long-range rover autonomy. *Journal of Field Robotics*, 27(5):534–560.

German, C. R., Yoerger, D. R., Jakuba, M., Shank, T. M., Langmuir, C. H., and Nakamura, K. (2008). Hydrothermal exploration with the Autonomous Benthic Explorer. *Deep Sea Research Part I: Oceanographic Research Papers*, 55(2):203–219.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 2961–2969.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Heikkilä, S. S., Halme, A., and Schiele, A. (2012). Affordance-based indirect task communication for astronaut-robot cooperation. *Journal of Field Robotics*, 29(4):576–600.

Hemachandra, S., Duvallet, F., Howard, T. M., Roy, N., Stentz, A., and Walter, M. R. (2015). Learning models for following natural language directions in unknown environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Howard, T. M. and Kelly, A. (2007). Optimal rough terrain trajectory generation for wheeled mobile robots. *The International Journal of Robotics Research*, 26(2):141–166.

Howard, T. M., Tellex, S., and Roy, N. (2014). A natural language planner interface for mobile manipulators. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 6926–6933.

Ierusalimschy, R. (2006). *Programming in Lua*. Roberto Ierusalimschy.

Johnson-Roberson, M., Pizarro, O., Williams, S. B., and Mahon, I. (2010). Generation and visualization of large-scale three-dimensional reconstructions from underwater robotic surveys. *Journal of Field Robotics*, 27(1):21–51.

Kaelbling, L. and Lozano-Pérez, T. (2011). Hierarchical task and motion planning in the now. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1470–1477.

Kang, S., Cho, C., Lee, J., Ryu, D., Park, C., Shin, K., and Kim, M. (2003). ROBHAZ-DT2: Design and integration of passive double tracked mobile manipulator system for explosive ordnance disposal. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2624–2629.

Keskinpala, H. K., Adams, J. A., and Kawamura, K. (2003). PDA-based human-robotic interface. In *Proceedings of the IEEE Conference on Systems, Man, and Cybernetics (SMC)*, pages 3931–3936, Washington, DC.

Kessens, C. C., Fink, J., Hurwitz, A., Kaplan, M., Osteen, P. R., Rocks, T., Rogers, J., Stump, E., Quang, L., DiBlasi, M., Gonzalez, M., Patel, D., Patel, J., Patel, S., Weiker, M., Bowkett, J., Detry, R., Karumanchi, S., Burdick, J., Matthies, L., Oza, Y., Agarwal, A., Dornbush, A., Likhachev, M., Schmeckpeper, K., Daniilidis, K., Kamat, A., Choudhury, S., Mandalika, A., and Srinivasa, S. (2020). Toward fieldable human-scale mobile manipulation using RoMan. In Pham, T., Solomon, L., and Rainey, K., editors, *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II*, volume 11413, pages 418 – 437.

Knepper, R. A., Tellex, S., Li, A., Roy, N., and Rus, D. (2015). Recovering from failure by asking for help. *Autonomous Robots*, 39:347–362.

Kollar, T., Perera, V., Nardi, D., and Veloso, M. (2013). Learning environmental knowledge from task-based human-robot dialog. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4304–4309.

Krotkov, E. and Blitch, J. (1999). The Defense Advanced Research Projects Agency (DARPA) Tactical Mobile Robotics Program. *International Journal of Robotics Research*, 18(7):769–776.

Leonard, J., How, J., Teller, S., Berger, M., Campbell, S., Fiore, G., Fletcher, L., Frazzoli, E., Huang, A., Karaman, S., Koch, O., Kuwata, Y., Moore, D., Olson, E., Peters, S., Teo, J., Truax, R., Walter, M. R., Barrett, D., Epstein, A., Maheloni, K., Moyer, K., Jones, T., Buckley, R., Antone, M., Galejs, R., Krishnamurthy, S., and Williams, J. (2008). A perception-driven autonomous urban vehicle. *Journal of Field Robotics*, 25(10):727–774.

Likhachev, M., Gordon, G. J., and Thrun, S. (2004). ARA*: Anytime A* with provable bounds on sub-optimality. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 767–774.

Lu, J., Xiong, C., Parikh, D., and Socher, R. (2017). Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Maimone, M., Cheng, Y., and Matthies, L. (2007). Two years of visual odometry on the mars exploration rovers. *Journal of Field Robotics*, 24(3):169–186.

Mandalika, A., Choudhury, S., Salzman, O., and Srinivasa, S. (2019). Generalized lazy search for robot motion planning: Interleaving search and edge evaluation via event-based toggles. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, pages 745–753.

Marshall, J., Barfoot, T., and Larsson, J. (2008). Autonomous underground tramming for center-articulated vehicles. *Journal of Field Robotics*, 25(6-7):400–421.

Massa, F. and Girshick, R. (2018). maskrcnn-benchmark: Fast, modular reference implementation of instance segmentation and object detection algorithms in PyTorch.

Matuszek, C., FitzGerald, N., Zettlemoyer, L., Bo, L., and Fox, D. (2012). A joint model of language and perception for grounded attribute learning. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Mertz, C., Navarro-Serment, L. E., MacLachlan, R., Rybski, P., Steinfeld, A., Suppe, A., Urmson, C., Vandapel, N., Hebert, M., Thorpe, C., Duggins, D., and Gowdy, J. (2013). Moving object detection with laser scanners. *Journal of Field Robotics*, 30(1):17–43.

Miller, I., Campbell, M., Huttenlocher, D., Kline, F., Nathan, A., Lupashin, S., Catlin, J., Schimpf, B., Moran, P., Zych, N., Garcia, E., Kurdziel, M., and Fujishima, H. (2008). Team Cornell's Skynet: Robust perception and planning in an urban environment. *Journal of Field Robotics*, 25(8):493–527.

Montemerlo, M., Becker, J., Bhat, S., Dahlkamp, H., Dolgov, D., Ettinger, S., Haehnel, D., Hilden, T., Hoffmann, G., Huhnke, B., Johnston, D., Klumpp, S., Langer, D., Levandowski, A., Levinson, J., Marcil, J., Orenstein, D., Paefgen, J., Penny, I., Petrovskaya, A., Pflueger, M., Stanek, G., Stavens, D., Vogt, A., and Thrun, S. (2008). Junior: The Stanford entry in the Urban Challenge. *Journal of Field Robotics*, 25(9):569–597.

Narayanan, P., Yeh, B., Holmes, E., Martucci, S., Schmeckpeper, K., Mertz, C., Osteen, P., and Wigness, M. (2020). An integrated perception pipeline for robot mission execution in unstructured environments. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II*, volume 11413, pages 447 – 465. International Society for Optics and Photonics, SPIE.

Oh, J., Howard, T. M., Walter, M. R., Barber, D., Zhu, M., Park, S., Suppe, A., Navarro-Serment, L., Duvallet, F., Boularias, A., Romero, O., Vinokurov, J., Keegan, T., Dean, R., Lennon, C., Bodt, B., Childers, M., Shi, J., Daniilidis, K., Roy, N., Lebiere, C., Hebert, M., and Stentz, A. (2017). Integrated intelligence for human-robot teams. In *Proceedings of the International Symposium on Experimental Robotics (ISER)*, pages 309–322.

Oh, J., Suppé, A., Duvallet, F., Boularias, A., Navarro-Serment, L., Hebert, M., Stentz, A., Vinokurov, J., Romero, O., Lebiere, C., and Dean, R. (2015). Toward mobile robots reasoning like humans. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, volume 29.

Park, D. (2018). *A multimodal execution monitor for assistive robots*. PhD thesis, Georgia Institute of Technology.

Partan, S. and Marler, P. (1999). Communication goes multimodal. *Science*, 283(5406):1272–1273.

Patki, S., Daniele, A., Walter, M. R., and Howard, T. M. (2019). Inferring compact representations for efficient natural language understanding of robot instructions. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Patki, S., Fahnestock, E., Howard, T. M., and Walter, M. R. (2020). Language-guided semantic mapping and mobile manipulation in partially observable environments. In *Conference on Robot Learning*, pages 1201–1210.

Patki, S. and Howard, T. M. (2018). Language-guided adaptive perception for efficient grounded communication with robotic manipulators in cluttered environments. In *Proc. Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*.

Paul, R., Arkin, J., Aksaray, D., Roy, N., and Howard, T. M. (2018). Efficient grounding of abstract spatial concepts for natural language interaction with robot platforms. *The International Journal of Robotics Research*, 37(10):1269–1299.

Paul, R., Arkin, J., Roy, N., and Howard, T. M. (2016). Efficient grounding of abstract spatial concepts for natural language interaction with robot manipulators. In *Proceedings of Robotics: Science and Systems (RSS)*.

Paul, R., Barbu, A., Felshin, S., Katz, B., and Roy, N. (2017). Temporal grounding graphs for language understanding with accrued visual-linguistic context. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4506–4514.

Perera, I. E. and Allen, J. F. (2013). Sall-e: Situated agent for language learning. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, page 1241–1247.

Perzanowski, D., Schultz, A. C., Adams, W., Marsh, E., and Bugajska, M. (2001). Building a multimodal human-robot interface. *IEEE Intelligent Systems*, 16(1):16–21.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Roberts, J. M., Duff, E. S., Corke, P. I., Sikka, P., Winstanley, G. J., and Cunningham, J. (2000). Autonomous control of underground mining vehicles using reactive navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 3790–3795.

Ryu, D., Kang, S., Kim, M., and Song, J. (2004). Multi-modal user interface for teleoperation of ROBHAZ-DT2 field robot system. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 168–173.

Scheding, S., Dissanayake, G., Nebot, E. M., and Durrant-Whyte, H. (1999). An experiment in autonomous navigation of an underground mining vehicle. *IEEE Transactions on Robotics and Automation*, 15(1):85–95.

Scheding, S., Nebot, E. M., Stevens, M., Durrant-Whyte, H., Roberts, J., Corke, P., Cunningham, J., and Cook, B. (1997). Experiments in autonomous underground guidance. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 1898–1903.

Scheutz, M., Briggs, G., Cantrell, R., Krause, E., Williams, T., and Veale, R. (2013). Novel mechanisms for natural human-robot interactions in the diarc architecture. In *AAAI Workshop on Intelligent Robot Systems*.

Scheutz, M., Cantrell, R., and Schermerhorn, P. (2011). Toward humanlike task-based dialogue processing for human robot interaction. *AI Magazine*, 32(4):77–84.

Singh, H., Can, A., Eustice, R., Lerner, S., McPhee, N., and Roman, C. (2004). Seabed AUV offers new platform for high-resolution imaging. *Eos, Transactions American Geophysical Union*, 85(31):289–296.

Tellex, S., Knepper, R., Li, A., Rus, D., and Roy, N. (2014). Asking for help using inverse semantics. In *Proceedings of Robotics: Science and Systems (RSS)*, Berkeley, CA.

Tellex, S., Kollar, T., Dickerson, S., Walter, M. R., Banerjee, A. G., Teller, S., and Roy, N. (2011). Approaching the symbol grounding problem with probabilistic graphical models. *AI Magazine*, 32(4):64–76.

Thomason, J., Murray, M., Cakmak, M., and Zettlemoyer, L. (2020). Vision-and-dialog navigation. In Kaelbling, L., Kragic, D., and Sugiura, K., editors, *Proceedings of the Conference on Robot Learning (CoRL)*, volume 100 of *Proceedings of Machine Learning Research*, pages 394–406, Osaka, Japan.

Thomason, J., Sinapov, J., Svetlik, M., Stone, P., and Mooney, R. J. (2016). Learning multi-modal grounded linguistic semantics by playing "I Spy". In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3477–3483. AAAI Press.

Thomason, J., Zhang, S., Mooney, R., and Stone, P. (2015). Learning to interpret natural language commands through human-robot dialog. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. AAAI Press.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A., and Mahoney, P. (2006). Stanley: The robot that won the DARPA Grand Challenge. *Journal of Field Robotics*, 23(9):661–692.

Tian, J. and Oh, J. (2019). Image captioning with compositional neural module networks. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3576–3584.

Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M. N., Dolan, J., Duggins, D., Galatali, T., Geyer, C., Gittleman, M., Harbaugh, S., Hebert, M., Howard, T. M., Kolski, S., Kelly, A., Likhachev, M., McNaughton, M., Miller, N., Peterson, K., Pilnick, B., Rajkumar, R., Rybski, P., Salesky, B., Seo, Y., Singh, S., Snider, J., Stentz, A., Whittaker, W., Wolkowicki, Z., Ziglar, J., Bae, H., Brown, T., Demitrish, D., Litkouhi, B., Nickolaou, J., Sadekar, V., Zhang, W., Struble, J., Taylor, M., Darms, M., and Ferguson, D. (2008). Autonomous driving in urban environments: Boss and the Urban Challenge. *Journal of Field Robotics*, 25(8):425–466.

Urmson, C., Ragusa, C., Ray, D., Anhalt, J., Bartz, D., Galatali, T., Gutierrez, A., Johnston, J., Harbaugh, S., Kato, H., Messner, W., Miller, N., Peterson, K., Smith, B., Snider, J., Spiker, S., Ziglar, J., Whittaker, W., Clark, M., Koon, P., Mosher, A., and Struble, J. (2006). A robust approach to high-speed navigation for unrehearsed desert terrain. *Journal of Field Robotics*, 23(8):467–508.

Walter, M. R., Antone, M., Chuangsuwanich, E., Correa, A., Davis, R., Fletcher, L., Frazzoli, E., Friedman, Y., Glass, J., How, J. P., Jeon, J. H., Karaman, S., Luders, B., Roy, N., Tellex, S., and Teller, S. (2015). A situationally-aware voice-commandable robotic forklift working alongside people in unstructured outdoor environments. *Journal of Field Robotics*, 32(4):590–628.

Wigness, M., Rogers, J. G., and Navarro-Serment, L. E. (2018). Robot navigation from human demonstration: Learning control behaviors. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1150–1157. IEEE.

Williams, S. B., Pizarro, O. R., Jakuba, M. V., Johnson, C. R., Barrett, N. S., Babcock, R. C., Kendrick, G. A., Steinberg, P. D., Heyward, A. J., Doherty, P. J., et al. (2012). Monitoring of benthic reference sites: using an autonomous underwater vehicle. *IEEE Robotics & Automation Magazine*, 19(1):73–84.

Williams, T., Acharya, S., Schreitter, S., and Scheutz, M. (2016). Situated open world reference resolution for human-robot dialogue. In *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 311–318.

Yamauchi, B. M. (2004). PackBot: A versatile platform for military robotics. In *Proceedings of the International Society for Optics and Photonics (SPIE), Unmanned Ground Vehicle Technology VI*, volume 5422, pages 228–237.

Yi, D., Howard, T. M., Seppi, K., and Goodrich, M. (2016). Expressing homotopic requirements for mobile robot navigation through natural language instructions. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1462–1468. IEEE.

Yoerger, D. R., Jakuba, M., Bradley, A. M., and Bingham, B. (2007). Techniques for deep sea near bottom survey using an autonomous underwater vehicle. *The International Journal of Robotics Research*, 26(1):41–54.

Ziebart, B. D., Maas, A. L., Bagnell, J. A., and Dey, A. K. (2008). Maximum entropy inverse reinforcement learning. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 1433–1438.

How to cite this article: Howard, T., M. Stump, E., Fink, J., Arkin, J., Paul, R., Park, D., Roy, S., Barber, D., Bendell, R., Schmeckpeper, K., Tian, J., Oh, J., Wigness, M., Quang, L., Rothrock, B., Nash, J., Walter, M. R., Jentsch, F., & Roy, N. (2022). An Intelligence Architecture for Grounded Language Communication with Field Robots. *Field Robotics, 2*, 468–512.