 http://dx.doi.org/10.1109/IROS51168.2021.9636434

An Integrated Approach to Context-Sensitive Moral Cognition in Robot Cognitive Architectures

Ryan Blake Jackson*†, Sihui Li*†, Santosh Balajee Banisetty†, Sriram Siva†,
Hao Zhang†, Neil Dantam†, and Tom Williams†

***Abstract*— Acceptance of social robots in human-robot collaborative environments depends on the robots' sensitivity to human moral and social norms. Robot behavior that violates norms may decrease trust and lead human interactants to blame the robot and view it negatively. Hence, for long-term acceptance, social robots need to detect possible norm violations in their action plans and refuse to perform such plans. This paper integrates the Distributed, Integrated, Affect, Reflection, Cognition (DIARC) robot architecture (implemented in the Agent Development Environment (ADE)) with a novel place recognition module and a norm-aware task planner to achieve context-sensitive moral reasoning. This will allow the robot to reject inappropriate commands and comply with context-sensitive norms. In a validation scenario, our results show that the robot would not comply with a human command to violate a privacy norm in a private context.**

I. INTRODUCTION

The fields of social robotics [1] and socially assistive robotics [2] are enabling robot integration into human environments. Unlike other robots, autonomous socially assistive robots such as therapy robots [3] and companion robots [4] interact with human partners on a social level through humanlike verbal [5] and non-verbal [6], [7] communicative cues (e.g., speech, gaze, gestures, and posture).

For social robots to be effectively integrated into human societies, they must be able to take actions (and make sense of the actions of others) with sensitivity to the social and moral norms that govern society. Social and moral norms are well understood to be both dynamic and malleable [8]. That is, different norms apply in different situations, with bundles of norms activated based on different contextual factors and cues; and norms change over time, on the basis of whether and how they are communicated between and enforced by community members. Shouting, for example, is permissible on a beach, but not in a library, and this standard is only upheld so long as library-goers continue to communicate this norm to each other, and sanction those who violate it.

Malle and Scheutz propose three key requirements for robotic moral competence that leverage knowledge of systems of moral norms [9], [10]: (1) moral cognition (the ability to make moral judgments in light of norms); (2) moral decision making and action (the ability to choose actions that conform to norms); and (3) moral communication (the ability to use norm-sensitive language and explain norm-relevant actions). While there is some previous work in the human-robot interaction literature on representing moral norms and enabling these key competencies, they have neither captured the context-sensitive nature of realistic moral norms, the need to account for morally impermissible actions that may be necessitated in the future if a given course of action is immediately adopted, nor the way that primitive actions are dynamically provided by distributed components of current integrated robot architectures.

Specifically, there has recently been a significant body of research towards enabling *transparent* and *explainable* robot systems, including approaches for explaining plans [11], [12], [13], [14], [15], [16], [17], rationalizing actions [18], transparently representing intent [19], preemptive explanation [20], and intention projection [21]. However, these approaches have not explicitly sought to enable explainability on moral grounds, nor have they captured the realistic way that human commands are typically framed, especially with respect to humans' use of sociocultural linguistic norms.

In contrast to Raman et al. [22], for example, socially assistive robots embedded into human social environments must be able not only to appropriately handle direct commands, but also more common indirect language as well; not only parsing natural language but also performing functions such as pragmatic inference and reference resolution; not only identifying contradictions between current and previous commands, but also identifying when commands are impermissible based on various context-sensitive systems of moral norms. As described in subsequent sections, these are capabilities enabled by our approach.

Similarly, there have been a number of attempts to devise mechanisms to ensure (or at least support) moral decision making for robots [23], [24], [25], [26], [27], [28], and some approaches towards enabling moral communication in robots, including work on *command rejection* [29], and generation of language to explain the robot's ethical (or unethical) decisions [30], [31], [32], [33], [34], [35], [36]. Other work has also recognized the need for robust and flexible task planning for HRI, and has sought to integrate that capacity with the various other capabilities necessary for task-based HRI [37].

Of greatest relevance to this work is that of Briggs et al. [29], who parse natural language into predicate logic formulae, and, after performing pragmatic reasoning, check whether or not the robot is permitted to perform the requested

* The first two authors contributed equally to this work.

† Department of Computer Science, Colorado School of Mines, USA. Email: {rbjackso, li, sbanisetty, sivasriram, hzhang, ndantam, twilliams}@mines.edu. This work was funded in part by National Science Foundation grant #IIS-1849348.

action. Our approach is similar to this approach, as it uses the same robot architecture, and more specifically, the same components for dialogue and goal management. However, their approach does not provide the breadth of situated, context-sensitive capabilities (such as reference resolution) needed to engage in task-based communication in situated contexts, and only identified commands that violated norms through immediate action, involving no planning to consider the permissibility of future actions, and was unable to automatically detect and leverage changes in context that should activate different sets of moral norms.

In this work, we seek to enable these new capacities through an integrated systems approach. By integrating goal-directed reasoning, task-planning, and context recognition capabilities provided by distinct robot architectures, we enable robots to successfully reject courses of action that would ultimately require actions that would violate context-sensitive deontic norms, using a set of actions and knowledge dynamically provided by a flexible set of distributed architectural components.

The remainder of this paper is structured as follows. In Section II, we present a brief overview of the DIARC architecture implemented in the agent development environment (ADE), including the key components used for goal-directed robotic cognition and sociolinguistic-norm-sensitive natural language understanding. In section III, we discuss a constraint-based robot task planner used to achieve high-level goals – and to generate an unsatisfiable core containing a minimal "explanation" for goal infeasibility when achievement of a user request is not possible or would require violation of norms to complete. In section IV, we discuss voxel-based representation learning for place recognition. Sections V and VI detail the integration and implementation of the proposed method and its validation, respectively. Finally, in section VII, we conclude with a discussion of limitations and directions for future work.

II. ADE & DIARC

The Distributed, Integrated, Affect, Reflection, Cognitive (DIARC) Robot Architecture [38] is a hybrid deliberative-reactive robot architecture that facilitates a wide variety of cognitive capabilities [39], with special attention to goal-driven cognition and natural language understanding and generation. DIARC is implemented in the Agent Development Environment (ADE) distributed multi-agent system middleware. ADE facilitates distributed computation, fault tolerance, recovery mechanisms, autonomic computing, and dynamic system configuration by treating architectural components as autonomous software agents [40], [41], [42]. Unlike other classic cognitive architectures, DIARC's polylithic nature is designed to enable autonomous, long-term robotic operation. Similar to robot middlewares such as ROS [43], Yarp [44], and JAUS [45], ADE facilitates parallel distributed communication and computation between architectural components. ADE was designed to be secure and fault-tolerant [40], [46].

The specific DIARC components leveraged in this work, and their interaction with the rest of our integrated system, are detailed in Section V. However, critical to note at this stage is that DIARC takes a goal-driven cognition approach to action selection, with different goals, derived from interlocutors or formulated by the robot itself, arbitrated between by the robot on the basis of their priority or affective appraisal, primarily taking whichever primitive actions can be immediately used to satisfy those goals. Previous work has demonstrated how this just-in-time goal-driven action selection can be made with sensitivity to deontic moral norms; however, in order to ensure that those actions do not necessitate *future* performance of norm-violating actions, forward-looking task planning is required. As such, in the next section we describe the task planning capabilities integrated with DIARC in this work.

III. TASK PLANNER

Robot task planning focuses on achieving high-level goals [47], [48]. In task planning, we describe the physical world through symbolic, typically discrete, states and actions that are abstracted from continuous motions. A task plan is a step by step sequence of actions that the robot takes to achieve a goal state. For example, to grasp an object inside a cabinet, the robot would need to perform the following actions: moving to the cabinet, opening the cabinet door, and grasping the target object. Given a proper description of the world, a task planner reasons about the robot's state and actions that change state over time to reach an intended goal.

The task planner requires a symbolic description of the world as input. We use Planning Domain Definition Language (PDDL) [49], a de facto standard in the planning community [50], to describe the world. PDDL describes the domain using first order logic and includes a set of predicates, objects in the world, actions with preconditions and effects, a start state, and a goal condition. States, i.e., truth of predicates applied to objects, change over time as a result of actions. PDDL separates a planning problem into two parts. First, a domain description specifies the discrete dynamics of the planning domain. The domain description includes a list of predicates that can be used to describe the state space of the robot, and a list of actions a robot can take in the world. Second, a fact description specifies a problem to be solved. The fact description includes a list of objects in the world, initial conditions, and goals. Since first developed, different versions of PDDL have been designed that enrich the types of problems PDDL can describe [51], [52], [53]. In our case, we use PDDL3 [53] because of its ability to specify facts that always hold during planning, which is essential when we encode moral norms.

A norm states how agents should behave in order to comport with community standards. In this work, we specifically focus on norms of obligation, permission, and forbiddeness, which indicate that certain actions or states must, can be, or must not be entered into or taken by "good" community members.

Definition 1: Formally, a moral norm is $C \implies \texttt{op}(x)$, where C is a context, op is "forbidden", "permitted", or "obligated", and x is set of states or actions.

We encode moral norms in the planning domain. We represent contexts as logical expressions on state variables. A norm is then a constraint to indicate some set of states or actions must not (forbidden), may (permitted), or must (obligated) occur.

$$\overbrace{C \implies \texttt{obligated}(x)}^{\text{moral norm}} \equiv \overbrace{\forall k,\ \neg(C^{\langle k \rangle} \wedge \neg x^{\langle k \rangle})}^{\text{planning constraint}} \quad (1)$$

$$\overbrace{C \implies \texttt{forbidden}(x)}^{\text{moral norm}} \equiv \overbrace{\forall k,\ \neg(C^{\langle k \rangle} \wedge x^{\langle k \rangle})}^{\text{planning constraint}} \quad (2)$$

With these definitions, we can encode moral norms into the PDDL descriptions using PDDL3's ability to describe facts that always hold during planning.

Given the PDDL descriptions, we run a constraint-based task planner to generate a plan [54]. The task planner encodes a planning problem into a set of constraints in the form of a Boolean formula, then adopts advanced Satisfiability Modulo Theories (SMT) solvers [55] to find a satisfying plan. To incorporate the moral norms, we add to the planer the ability to encode the moral norms in the PDDL descriptions into a set of Boolean formulas that must be satisfied at each step of the plan. In this way, the task planner always returns plans that follow the moral norms.

When no plan can be found, the task planner will produce an unsatisfiable core, which we use to analyze the cause of the planning failure. The unsatisfiable core is the minimal set of clauses (e.g., goals, norms, and action preconditions) that make the plan infeasible. If an unsatisfiable core result includes both a moral norm and actions, it means the actions are mutually exclusive with the moral norm, thus the actions are the cause of plan failure under the moral norm.

At this stage, norms, and the contexts in which they apply, are specified *a priori* to the robot by human operators. However, we have included endpoints in the task planner's API to allow DIARC components to dynamically add norms if they have norms that govern their actions or the capacity to learn norms over time. The context-sensitive norm-based moral reasoning performed using this planner relies on knowledge of the robot's context, so we will now describe the system we use to perceive and recognize context.

IV. Place Recognition For Context Generation

In this work, we specifically considered location-based contexts that can be recognized using place recognition techniques [56], which seek to identify a given location from a set of templates. Place recognition is a generally useful capability for robotic systems, as it can be used to reduce the uncertainty and ambiguity in estimated maps and robot poses, thereby significantly improving the accuracy of robot mapping and localization.

Long-term Place recognition [57] addresses the key challenge that many robot navigation environments are dynamic in nature and change over time. For example, in the case of indoor navigational environments the lighting conditions, arrangement of furniture, and human activities and movements can change on a daily basis.

In this paper, we achieve Long-term Place recognition using voxel-based representation learning approach [58] (VBRL)that uses 3D point clouds to recognize previously visited locations. Unlike methods that rely on RGB cameras [59], [60], [61], the VBRL approach uses a LiDAR sensor to obtain the 3D point cloud representation of the environment. This enables the robot to operate in environments with low lighting conditions and also helps to recognize contexts from the 360-degree field of view of the LiDAR sensor. This is especially helpful in dynamic indoor environments where humans may occlude the limited field of view of an RGB-based camera sensor.

The VBRL approach divides each 3D point cloud obtained from a LiDAR sensor into multiple voxels in the 3D space. Multiple types of features are then extracted from each voxel. The VBRL approach then automatically learns the importance of each feature modality extracted from these 3D voxels, as well as the importance of the voxels themselves. Voxel importance learning is inspired by the insight that specific set of voxels are more representative and better encode location-based contexts. For example, in a 3D point cloud based voxel representation, the voxels closer to the LiDAR sensor can be more informative in representing the place, since more details are captured by the 3D points of objects that are closer to the sensor. Mathematically, the learning of voxel importance is achieved in the VBRL approach using structured sparsity-inducing norms as regularizations into the optimization formulation. These learned representations are then integrated in a unified regularized optimization formulation to best represent location-based contexts.

V. Architecture & Integration

In this section, we briefly discuss the way in which the planning and context recognition capabilities described in the previous sections are integrated with the DIARC architecture. This integration is shown in Figure 1.

A. Natural Language Understanding

We begin by describing the Natural Language Understanding components used in our DIARC configuration because the goals that drive robot behavior typically come from natural language human utterances. The first component used for Natural Language Understanding is Automatic Speech Recognition (ASR), which converts natural speech signals (acoustic signals) into text representations, which are sent to the architecture's Parser. The Parser translates the text representations provided by ASR into unbound logical predicates representing the surface semantics of the speaker's utterance, by means of a Combinatory Categorial Grammar. Uniquely, this grammar encodes Givenness Hierarchy theoretic information in resultant parse representations, to facilitate anaphora resolution. These representations are then provided to the Pragmatics component, which uses a set of context-sensitive rules encoding sociocultural norms

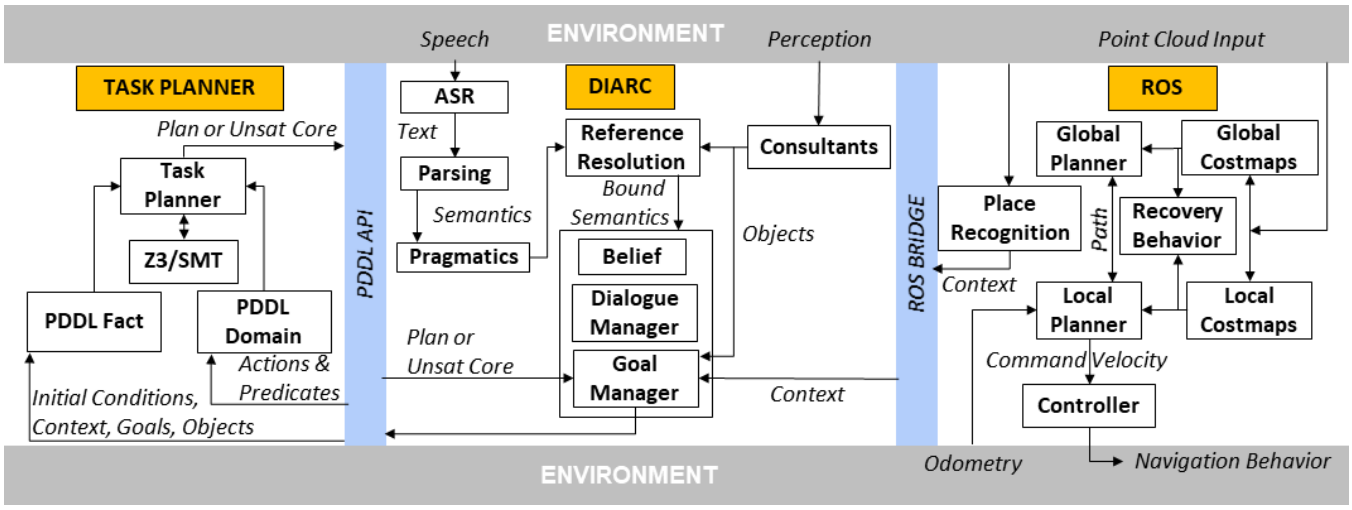

Fig. 1. Integrated Robot Architecture

(especially the sociocultural politeness norms needed to understand and generate indirect speech acts) to translate those surface semantics into the (unbound) intended meaning of the utterance [62], [63]. This Utterance Structure is then provided to Reference Resolution, which uses Givenness Hierarchy theoretic processes [64], [65] with a Probabilistic Open-World Reference Resolution [66], [67] subroutine to identify what objects, locations, people, etc., were involved in any noun phrases, in order to produce a bound utterance structure precisely encoding the meaning of the speaker's utterance as grounded in the robot's knowledge of its environment. These representations can then be provided to the Dialogue Manager component, which, *if* the robot decides to do so, uptakes any assertions, questions, and goals.

B. Goal Manager

Goals uptaken from human utterances or otherwise formulated by the robot are handled by DIARC's Goal Manager, which selects actions to take in response to those goals [68], [69]. These actions could be steps towards achieving the goal, or communicative acts relating to the goal, such as issuing a command refusal for a goal that does not comply with the robot's moral reasoning capabilities.

Because the goal manager functions as the central executive of high-level cognition in the DIARC architecture, it is where we decided to integrate DIARC with the task planning and context recognition systems described in the previous sections. Prior to this integration with the task planner and context recognizer, ADE's Goal Manager had some rudimentary moral reasoning capabilities[26]: Given a list of forbidden states or actions, the Goal Manager would never take a forbidden action or an action that was known to directly cause a forbidden state. However, any forbidden action was forbidden categorically, regardless of context, and, without the ability to determine context continuously from perceptual information, it was also not practical to specify context-sensitive forbidden states. The new context recognizer and norm specification method solve these issues.

However, an even greater advantage of integrating the Goal Manager and task planner is the ability this enables to communicate about infeasible or impermissible goals. Previous experimental work has demonstrated a need for robots to communicate clearly, thoroughly, and proactively about morally impermissible human commands. Failure to do so can both mislead human interlocutors about the robot's moral intentions and also, perhaps more worryingly, weaken human perception or application of moral norms within their current context [70]. Recently, we have developed mechanisms that avoid these issues in certain situations by communicating more proactively about infeasible and impermissible human commands [71]. Obtaining more detailed information from the planner's unsatisfiable core will allow us to construct more detailed and effective command refusals.

The Goal Manager communicates with the task planner via a REST API as shown in Figure 1. We now describe the five types of information that the Goal Manager aggregates and sends to the planner, where this information comes from, and exactly how it is communicated.

1) Actions: Every component in ADE advertises actions that correspond to the abilities of the robot. For example, the natural language generation components provide actions for saying words, while the component controlling the robot's body provides actions for moving in and manipulating the environment. These actions may be annotated with preconditions that must be met before they can be taken, and effects of having performed them. Every action is automatically assumed to have the effect of having done the action (e.g., the action "grasp(x)" is automatically given the effect "did_grasp(x)", and may be optionally annotated with further effects like "holding(x)"). The Goal Manager is notified by the central registry of ADE components whenever a component joins or leaves the system (since ADE is designed to allow distributed multi-robot systems, components can join and leave dynamically at unpredictable times). Whenever a component joins, the Goal Manager updates the task planner with all of that component's actions, as well as their

parameters, preconditions, and effects. The Goal Manager does the same thing for any components that are already running when it starts running. Likewise, when a component leaves, the Goal Manager removes the actions that are no longer available from the task planner's domain.

2) Predicates: For purposes of the task planner, predicates specify everything that can be true of the world and the objects in it. A variety of predicates are sent to the task planner for different reasons, as detailed below.

First, every action precondition and effect are automatically added to the task planner as predicates when the relevant action is added.

Second, due to our use of a context recognition algorithm, the Goal Manager adds a predicate "in(?context)" when it starts running that allows it to later specify the context that the robot is in (e.g., "in(corridor)").

Third, other predicates are provided by the robot's perceptual capabilities and built-in ontologies, through a general *Consultant* interface as described in previous work [72]. Specifically, ADE uses the Givenness Hierarchy theoretic version [65], [64] of the Probabilistic Open-World Entity Resolution (POWER) algorithm [67] and its associated consultant framework [72] for reference resolution, and the same consultants are relevant here. The robot can be provided with a variety of different consultants to handle the different kinds of information that it might need to know in any given role. We commonly use a vision consultant to perceive and store knowledge about visually perceptible objects and their properties. Predicates that would come from the vision consultant might include color like "red(x)" and "green(x)", and type of object like "ball(x)" or "box(x)", but could include any object property the robot can discern. Another example is the agent consultant that stores information about other agents (like humans) with which the robot interacts.

Unlike with actions, we do not simply update the task planner with predicates from consultants whenever a consultant joins or leaves. Some consultants can dynamically change the properties that they handle, for example, by learning new properties (e.g., the vision consultant being taught a new color "blue(x)" when previously the only two known colors were red and green). To allow for this kind of learning and flexibility in the consultants, the Goal Manager always queries the consultants for any new properties handled immediately before requesting a new plan for a new goal. It then sends these new properties as new predicates to the task planner. Likewise, any properties that used to be handled by some consultant but are not anymore (e.g., if a consultant stopped running) are removed from the task planner's list of predicates at this stage.

3) Objects: Objects, as far as the task planner is concerned, are things in the world to which predicates can apply or actions can be done. One important set of objects for our integration with the context recognizer is the set of all possible contexts that can be recognized. These context labels are necessarily known *a priori*, so the Goal Manager sends all of them as objects to the task planner when it starts running. Other objects come from the consultants described above. Since consultants can continuously learn of new objects or discard misperceived objects or objects that become irrelevant for whatever reason, the Goal Manager always queries the consultants for known objects immediately before requesting a new plan, and updates the task planer's list of objects accordingly.

4) Initial Conditions: Since the state of the world relative to the robot can change during the time between calls to the task planner, the Goal Manager updates the task planner with a new set of initial conditions each time it requests a new plan. One important initial condition is the context that the robot is in, which the Goal Manager gets from the ROS topic associated with the context recognizer and then sends to the task planner as an "in" predicate (e.g., "in(corridor)").

Other initial conditions could theoretically come from the consultants described above, but it is computationally wasteful to update the planner with all knowledge from every consultant about every known entity in the world, when the vast majority of this information is likely irrelevant to any given goal. Furthermore, many consultants deal with uncertainty and ambiguity, both perceptual and linguistic, and therefore cannot always assert all properties of an entity with a useful degree of certainty. Therefore, we have created a way for the Goal Manager to query consultants about specific objects and send the results to the task planner so that, in the future, we can either specify important domain-specific objects *a priori* or alter the task planner such that there is a bidirectional interchange between it and the Goal Manager throughout the planning process such that the task planner can request specific information that the Goal Manager can then provide via consultants (e.g., where can we find a cutting board?).

5) Goals: Of course, to obtain a task plan for a goal, the Goal Manager must send that goal to the task planner. Goals are specified as predicates describing some desired state of the world (e.g., "did-grasp(object1)"). After specifying a goal to the task planner, the Goal Manager activates an API endpoint telling the planner to make a plan, and waits for it to finish. When planning is done, the Goal Manager receives either the completed plan if possible or the unsatisfiable core if a plan could not be made for whatever reason. This result remains available until a new plan is requested so that it can eventually be accessed multiple times by upstream dialogue components if necessary without re-planning.

C. The task planner

The inputs to the task planer (left of Figure 1) are the PDDL domain description and fact description. The task planner exposes a Web Service API, which ADE uses to communicate changes to the domain and fact descriptions. The task planer outputs a plan if the goals are satisfiable under the norms, or an unsatisfiable core containing actions, goals, and norms that cause planning failure.

The domain description encodes all the actions the robot can take. These actions come from the various ADE components that advertise the actions that they enable the robot to do. The domain description is automatically generated

from these components as described above. The task planner API automatically adds new predicates in the actions' pre-conditions and effects fields to the domain description.

Moral norms are encoded in the PDDL as described in section III. We update the objects, initial conditions and goals in the fact description every time a plan is required for a new goal. Most notably, place recognition results update the initial condition in the fact description with a predicate like "in(corridor)", which changes the context of the current plan. Other initial conditions and objects come from ADE consultants such as the vision consultant, as described above.

D. Navigation and Place Recognition

Robot navigation is achieved through the navigation stack of ROS. ROS *nav_stack* is configured to use *Search Based Planning Library* (SBPL) global planner and *Model Predictive Path Intergral* (MPPI) local planner for global and local planning respectively. The *map_server* input is used by *global_costmap* package to represent global environmental obstacles with the help of sensory input such as laser scanners. On the other hand, *local_costmap* package represents dynamic and nearby obstacles as costmaps using the same laser scan input. The *global_planner* takes a goal pose as input and computes the shortest path from the robot's current position to the goal. This computed path is fed as input to the *local_planner* which follows the path closely by avoiding obstacles as detected in the local costmaps. The local planner computes the *cmd_vel* (desired robot velocity) to reach the desired goal based on odometry data from the environment. ROS navigation stack also incorporates recovery behaviors to help the robot if it gets stuck (right of Figure 1).

Our voxel-based place recognition module uses 360-degree 3D point cloud data as input to determine the robot's current location (place label). A 3D point cloud of the environment is constructed using LiDAR input which is fed to our VBRL place recognition node, which in turn outputs the label of the recognized place as a ROS topic accessible to DIARC's goal manager; for example, corridor, classroom, etc. This is the source of the context information for the goal manager and task planner. Adding further perceptual capabilities could allow for more detailed context information or other types of context information. The integration of ROS components with DIARC is through the *rosbridge_suite* package [73], which uses a WebSocket interface via Java API to communicate with non-ROS parts of the robot, in our case, DIARC's goal manager.

VI. VALIDATION

To demonstrate the functionality of our integration and to more concretely illustrate the concepts described above, we evaluate our system in a simple example scenario. This scenario is designed to showcase our multi-step planning capability that takes into account context-sensitive norms as the robot moves through various contexts (see Figure 2). Because this work is not concerned with having the robot actually manipulate its environment, but rather with the cognitive capacities required to make a plan to do so, and to avoid gatherings of students during the COVID-19 pandemic, point cloud information was pre-collected and played back during testing (Figure 2 inset), with courses of action planned but not executed.

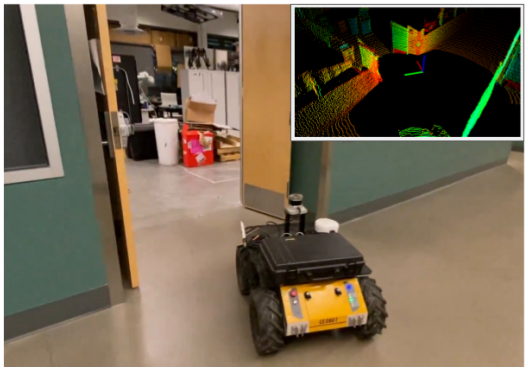

Fig. 2. The Clearpath Husky used in the validation of our system. **Inset:** Sensory input to the robot.

A. Setup

This scenario takes place in a typical academic building on a university campus. The four contexts involved in our scenario, which are recognized from point cloud data, labeled, and supplied to the Goal Manager as described above, are: Corridor, Lab, Washroom, and Studyroom. The robot moves between these contexts, and receives a human command to "report occupants" in each.

The robot knows three actions relevant to reporting the occupants in a room. The `report-occupants` action achieves the goal of reporting the occupants, but requires that the robot take a picture of the room as a prerequisite. The `take-picture` action fulfills this prerequisite, but requires as a prerequisite of its own that the robot make a noise to get the attention of the people in the room. The `attention-noise` action achieves this prerequisite and has no prerequisites. Thus, the instruction to report occupants requires three steps: (1) `attention-noise`, (2) `take-picture`, (3) `report-occupants`. We chose these actions to present a multi-step process that would be feasible for our robot, which has perceptual and movement capabilities but no arms or graspers for manipulation.

There is also a norm in our scenario that the robot is not allowed to perform the "take picture" action in the washroom context. We believe that typical privacy norms make this rule very realistic. This norm is represented in the PDDL fact file as follows: `(and (always (or (not (did-takepicture)) (not (in washroom)))))`

B. Results

As expected, in any room except the washroom, the planner returns the sequence of three actions required to achieve the goal of reporting the occupants such that the Goal Manager could then parse this plan and execute this sequence of actions. In the washroom, the planner returns the unsatisfiable core specifying that taking a picture is incompatible with being in the washroom. This information

could then be used by the natural language generation pipeline to communicate this reasoning to the human in a command refusal.

VII. DISCUSSION & FUTURE WORK

To summarize, our integrated approach to context-sensitive moral cognition uses automatically generated context-specific domain descriptions to encode the actions a robot can take, as provided by a dynamic and flexible set of architectural components. By doing so, a robot can perform context-aware rejection of morally impermissible or infeasible plans. Our work differs from existing methods in its ability to (1) activate different moral norms based on its (automatically sensed) context, (2) assess the permissibility of future behaviors that would be required when committing to an immediate course of action, and (3) perform moral reasoning regarding natural language containing realistic references and indirect speech acts that must be resolved based on the robot's situated context. Finally, the integration presented in this paper and the novel capabilities enabled by this integration lay the groundwork for a variety of directions for future work.

The first step for building on this architecture will be to parse plans from the task planner into action scripts usable by DIARC. This will allow each action in the plan to be sent to the component responsible for performing that action, and for plans to be executed in a distributed fashion.

Second, in future work the unsatisfiable core may be used to generate natural language command rejections for morally impermissible human commands. Prior work has shown that properly calibrating the politeness of robotic command rejections to conversational and social context is critical to HRI [74], [75], so it will be important not only to convey the information in the unsatisfiable core to humans, but also to do so in contextually appropriate polite language.

Third, there may be advantages to more closely integrating the task planner with DIARC. As mentioned above, planning for complex tasks in uncertain and open worlds may require the task planner to query the Goal Manager during the planning process. For example, if a food preparation task requires a cutting board, the planner may need to ask the consultant framework which objects are cutting boards and where the nearest one is, before it can plan to obtain a cutting board. Likewise, it may be useful for the planner to request human clarification between alternative plans, which would involve DIARC's natural language generation pipeline. Likewise, context information may be relevant to more DIARC components than just the Goal Manager. Different contexts, for example, might entail different speech norms that would be relevant to pragmatic generation.

Fourth, prior work in socially-aware navigation and human-robot proxemics [76], [77] identified the need for unified socially-aware navigation (USAN) methods for context-sensitive long-term human-robot interaction in public places. In future work, the social and moral norms activated in a given context may be fed to a low-level social navigation planner [78] to achieve context-sensitive social navigation.

REFERENCES

[1] C. L. Breazeal, *Designing sociable robots*. MIT press, 2002.
[2] D. Feil-Seifer and M. J. Mataric, "Defining socially assistive robotics," in *9th International Conference on Rehabilitation Robotics*, 2005.
[3] K. Wada and T. Shibata, "Robot therapy in a care house-change of relationship among the residents and seal robot during a 2-month long study," in *Proc. IEEE RO-MAN*, 2007.
[4] A. Liang, I. Piroth, H. Robinson, B. MacDonald, M. Fisher, U. M. Nater, N. Skoluda, and E. Broadbent, "A pilot randomized trial of a companion robot for people with dementia living in the community," *Journal of the American Medical Directors Association*, 2017.
[5] S. Nikolaidis, M. Kwon, J. Forlizzi, and S. Srinivasa, "Planning with verbal communication for human-robot collaboration," *ACM Transactions on Human-Robot Interaction (THRI)*, vol. 7, no. 3, 2018.
[6] Y. Che, A. M. Okamura, and D. Sadigh, "Efficient and trustworthy social navigation via explicit and implicit robot–human communication," *IEEE Transactions on Robotics*, vol. 36, no. 3, 2020.
[7] A. D. Dragan, S. Bauman, J. Forlizzi, and S. S. Srinivasa, "Effects of robot motion on human-robot collaboration," in *Proc. HRI*, 2015.
[8] F. Gino, "Understanding ordinary unethical behavior: Why people who value morality act immorally," *Curr. Opinion in Behavioral Sci.*, 2015.
[9] B. F. Malle and M. Scheutz, "Moral competence in social robots," in *Symposium on Ethics in Science, Technology and Engineering*, 2014.
[10] B. F. Malle, "Integrating robot ethics and machine morality: The study and design of moral competence in robots," *Ethics & Info. Tech.*, 2016.
[11] N. Tintarev and R. Kutlak, "Sassy-making decisions transparent with argumentation and natural language generation," in *IUI 2014 Workshop: Interacting with Smart Objects*, 2014.
[12] T. Chakraborti, S. Sreedharan, Y. Zhang, and S. Kambhampati, "Plan explanations as model reconciliation: Moving beyond explanation as soliloquy," in *Proceedings of IJCAI*, 2017.
[13] R. Korpan and S. L. Epstein, "Toward natural explanations for a robot's navigation plans," in *HRI WS on Explainable Robotic Systems*, 2018.
[14] B. Seegebarth, F. Müller, B. Schattenberg, and S. Biundo, "Making hybrid plans more clear to human users-a formal approach for generating sound explanations," in *Proc. ICAPS*, 2012.
[15] M. Fox, D. Long, and D. Magazzeni, "Explainable planning," *IJCAI Workshop on Explainable AI*, 2017.
[16] S. Sreedharan, T. Chakraborti, and S. Kambhampati, "Balancing explicability and explanation in human-aware planning," in *AAAI Fall Symposium on AI-for-HRI*, 2017.
[17] S. Sohrabi, J. A. Baier, and S. A. McIlraith, "Preferred explanations: Theory and generation via planning.," in *AAAI*, 2011.
[18] S. Gregor and I. Benbasat, "Explanations from intelligent systems: Theoretical foundations and implications for practice," *MIS Q'ly*, 1999.
[19] E. Holder, "Defining soldier intent in a human-robot natural language interaction context," tech. rep., US Army Research Laboratory, 2017.
[20] Z. Gong and Y. Zhang, "Robot signaling its intentions in human-robot teaming," in *HRI Workshop on Explainable Robotic Systems*, 2018.
[21] R. S. Andersen, O. Madsen, T. B. Moeslund, and H. B. Amor, "Projecting robot intentions into human environments," in *Proc. Int'l Symp. on Robot and Human Interactive Communication*, 2016.
[22] V. Raman, C. Lignos, C. Finucane, K. C. Lee, M. P. Marcus, and H. Kress-Gazit, "Sorry dave, i'm afraid i can't do that: Explaining unachievable robot tasks using natural language.," in *Robotics: Science and Systems*, 2013.
[23] R. C. Arkin, "Governing lethal behavior: Embedding ethics in a hybrid deliberative/reactive robot architecture," in *Proc. 3rd ACM/IEEE International Conference on Human-Robot Interaction*, 2008.
[24] S. Bringsjord, K. Arkoudas, and P. Bello, "Toward a general logicist methodology for engineering ethically correct robots," *Intelligent Systems*, vol. 21, no. 4, 2006.
[25] R. Sun, "Moral judgment, human motivation, and neural networks," *Cognitive Computation*, vol. 5, no. 4, 2013.
[26] M. Scheutz, B. Malle, and G. Briggs, "Towards morally sensitive action selection for autonomous social robots," in *Proc. ROMAN*, 2015.
[27] D. Vanderelst and A. Winfield, "An architecture for ethical robots inspired by the simulation theory of cognition," *Cognitive Systems Research*, 2017.
[28] W. Wallach, S. Franklin, and C. Allen, "A conceptual and computational model of moral decision making in human and artificial agents," *Topics in Cognitive Science*, vol. 2, no. 3, 2010.
[29] G. Briggs and M. Scheutz, ""Sorry, I can't do that": Developing mechanisms to appropriately reject directives in human-robot interactions," in *Proceedings of the AAAI Fall Symposium Series*, 2015.

[30] F. Alaieri and A. Vellino, "Ethical decision making in robots: Autonomy, trust and responsibility," in *Proceedings of the International Conference on Social Robotics*, 2016.

[31] V. Charisi, L. Dennis, M. F. R. Lieck, A. Matthias, M. S. J. Sombetzki, A. F. Winfield, and R. Yampolskiy, "Towards moral autonomous systems," *arXiv preprint arXiv:1703.04741*, 2017.

[32] B. Kuipers, "Human-like morality and ethics for robots.," in *AAAI Workshop: AI, Ethics, and Society*, 2016.

[33] F. Lindner and M. M. Bentzen, "The hybrid ethical reasoning agent immanuel," in *Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, 2017.

[34] M. Lomas, R. Chevalier, E. V. Cross II, R. C. Garrett, J. Hoare, and M. Kopack, "Explaining robot actions," in *Proc. Int'l Conf. Human-Robot Interaction*, 2012.

[35] P. Schermerhorn and M. Scheutz, "Dynamic robot autonomy: Investigating the effects of robot decision-making in a human-robot team task," in *Proc. Int'l Conf. Multimodal Interfaces*, 2009.

[36] P. Schermerhorn and M. Scheutz, "Disentangling the effects of robot affect, embodiment, and autonomy on human team members in a mixed-initiative task," in *Proc. Adv. in Computer-Human Int'n.*, 2011.

[37] S. Lemaignan, M. Warnier, E. A. Sisbot, A. Clodic, and R. Alami, "Artificial cognition for social human–robot interaction: An implementation," *Artificial Intelligence*, vol. 247, pp. 45–69, 2017.

[38] M. Scheutz, T. Williams, E. Krause, B. Oosterveld, V. Sarathy, and T. Frasca, "An overview of the distributed integrated cognition affect and reflection diarc architecture," *Cognitive architectures*, 2019.

[39] M. Scheutz, G. Briggs, R. Cantrell, E. Krause, T. Williams, and R. Veale, "Novel mechanisms for natural human-robot interactions in the DIARC architecture," in *Proc. AAAI WS on Int. Rob. Sys.*, 2013.

[40] V. Andronache and M. Scheutz, "Ade—an architecture development environment for virtual and robotic agents," *International Journal on Artificial Intelligence Tools*, vol. 15, no. 02, 2006.

[41] J. Kramer and M. Scheutz, "ADE: A framework for robust complex robotic architectures," in *Proc. Int'l Conf. Intel. Robots and Sys.*, 2006.

[42] M. Scheutz, "Ade: Steps toward a distributed development and runtime environment for complex robotic agent architectures," *Applied Artificial Intelligence*, vol. 20, no. 2-4, 2006.

[43] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, 2009.

[44] G. Metta, P. Fitzpatrick, and L. Natale, "YARP: yet another robot platform," *Int'l Journal of Advanced Robotic Systems*, 2006.

[45] S. Rowe and C. R. Wagner, "An introduction to the joint architecture for unmanned systems (JAUS)," *Ann Arbor*, 2008.

[46] P. Schermerhorn and M. Scheutz, "Natural language interactions in distributed networks of smart devices," *International Journal of Semantic Computing*, vol. 2, no. 04, 2008.

[47] R. E. Fikes and N. J. Nilsson, "Strips: A new approach to the application of theorem proving to problem solving," *Artificial intelligence*, vol. 2, no. 3-4, 1971.

[48] J. Hoffmann and B. Nebel, "The ff planning system: Fast plan generation through heuristic search," *Journal of Artificial Intelligence Research*, vol. 14, 2001.

[49] M. Ghallab, A. Howe, C. Knoblock, D. McDermott, A. Ram, M. Veloso, D. Weld, and D. Wilkins, "PDDL— the planning domain definition language," tech. rep., 1998.

[50] M. Vallati, L. Chrpa, M. Grzes, T. L. McCluskey, M. Roberts, and S. Sanner, "The 2014 international planning competition: Progress and trends," *AI Magazine*, vol. 36, no. 3, 2015.

[51] M. Fox and D. Long, "PDDL2. 1: An extension to PDDL for expressing temporal planning domains," *Jour. AI Research*, 2003.

[52] S. Edelkamp and J. Hoffmann, "PDDL2. 2: The language for the classical part of the 4th international planning competition," tech. rep., University of Freiburg, 2004.

[53] A. Gerevini and D. Long, "Preferences and soft constraints in PDDL3," in *ICAPS WS on planning with preferences and soft constraints*, 2006.

[56] F. Han, X. Yang, Y. Deng, M. Rentschler, D. Yang, and H. Zhang, "SRAL: Shared representative appearance learning for long-term visual place recognition," *IEEE Robotics and Automation Letters*, 2017.

[54] N. T. Dantam, Z. K. Kingston, S. Chaudhuri, and L. E. Kavraki, "An incremental constraint-based framework for task and motion planning," *The International Journal of Robotics Research*, vol. 37, no. 10, 2018.

[55] L. De Moura and N. Bjørner, "Z3: An efficient smt solver," in *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, 2008.

[57] S. Siva and H. Zhang, "Omnidirectional multisensory perception fusion for long-term place recognition," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[58] S. Siva, Z. Nahman, and H. Zhang, "Voxel-based representation learning for place recognition based on 3d point clouds," in *IEEE/RSJ International conference on Intelligent Robots and Systems*, 2020.

[59] Z. Chen, A. Jacobson, N. Sünderhauf, B. Upcroft, L. Liu, C. Shen, I. Reid, and M. Milford, "Deep learning features at scale for visual place recognition," in *Proc. Int'l Conf. on Rob. and Automation*, 2017.

[60] E. Pepperell, P. I. Corke, and M. J. Milford, "All-environment visual place recognition with smart," in *Proc. ICRA*, 2014.

[61] H. Zhang, F. Han, and H. Wang, "Robust multimodal sequence-based loop closure detection via structured sparsity.," in *Robotics: Science and systems*, 2016.

[62] T. Williams, G. Briggs, B. Oosterveld, and M. Scheutz, "Going beyond literal command-based instructions: Extending robotic natural language interaction capabilities," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[63] G. Briggs, T. Williams, and M. Scheutz, "Enabling robots to understand indirect speech acts in task-based interactions," *Journal of Human-Robot Interaction*, 2017.

[64] T. Williams, S. Acharya, S. Schreitter, and M. Scheutz, "Situated open world reference resolution for human-robot dialogue," in *Proceedings of HRI*, 2016.

[65] T. Williams and M. Scheutz, "Reference in robotics: A givenness hierarchy theoretic approach," in *The Oxford Handbook of Reference*, Oxford University Press, 2019.

[66] T. Williams and M. Scheutz, "Power: A domain-independent algorithm for probabilistic, open-world entity resolution," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[67] T. Williams and M. Scheutz, "A framework for resolving open-world referential expressions in distributed heterogeneous knowledge bases," in *Proceedings of AAAI*, 2016.

[68] T. Brick and M. Scheutz, "Incremental natural language processing for HRI," in *Proc. 2nd ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2007.

[69] J. Dzifcak, M. Scheutz, C. Baral, and P. Schermerhorn, "What to do and how to do it: Translating natural language directives into temporal and dynamic logic representation for goal management and action execution," in *Proc. Int'l Conf. on Robotics and Automation*, 2009.

[70] R. B. Jackson and T. Williams, "Language-capable robots may inadvertently weaken human moral norms," in *Proc. alt.HRI*, 2019.

[71] R. Blake Jackson and T. Williams, "Enabling morally sensitive robotic clarification requests," *arXiv e-prints*, 2020.

[72] T. Williams, "A consultant framework for natural language processing in integrated robot architectures," *IEEE Intell. Informatics Bull.*, 2017.

[73] C. Crick, G. Jay, S. Osentoski, B. Pitzer, and O. C. Jenkins, "Rosbridge: ROS for non-ROS users," in *Robotics Research*, 2017.

[74] R. B. Jackson, R. Wen, and T. Williams, "Tact in noncompliance: The need for pragmatically apt responses to unethical commands," in *AAAI Conf. on Artificial Intelligence, Ethics, and Society*, 2019.

[75] R. B. Jackson, T. Williams, and N. Smith, "Exploring the role of gender in perceptions of robotic noncompliance," in *Proc. Int'l Conf. Human-Robot Interaction*, 2020.

[76] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch, "Human-aware robot navigation: A survey," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1726–1743, 2013.

[77] S. B. Banisetty and D. Feil-Seifer, "Towards a unified planner for socially-aware navigation," *arXiv preprint arXiv:1810.00966*, 2018.

[78] S. B. Banisetty, S. Forer, L. Yliniemi, M. Nicolescu, and D. Feil-Seifer, "Socially-aware navigation: A non-linear multi-objective optimization approach," *arXiv preprint arXiv:1911.04037*, 2019.