Answer me this: Constructing Disambiguation Queries for Explanation Generation in Robotics

Tiago Mota
Electrical and Computer Engineering
University of Auckland
Auckland, NZ
tmot987@aucklanduni.ac.nz

Mohan Sridharan
School of Computer Science
University of Birmingham
Birmingham, UK
m.sridharan@bham.ac.uk

***Abstract*—Our architecture seeks to enable robots collaborating with humans to describe their decisions and evolution of beliefs. To achieve the desired transparency in integrated robot systems that support knowledge-based reasoning and data-driven learning, we build on a baseline system that supports non-monotonic logical reasoning with incomplete commonsense domain knowledge, data-driven learning from a limited set of examples, and inductive learning of previously unknown axioms governing domain dynamics. In the context of a simulated robot providing on-demand, relational descriptions as explanations of its decisions and beliefs, we introduce an interactive system that automatically traces beliefs, and addresses ambiguity in the human queries by constructing and posing suitable disambiguation queries. We present results of evaluation in scene understanding and planning tasks to demonstrate our architecture's abilities.**

***Index Terms*—Explainable reasoning and learning, non-monotonic logical reasoning, deep learning, HRI**

I. Introduction

Consider a robot estimating the occlusion of objects and stability of object structures while arranging objects in desired configurations on a table; some example simulated scenes are shown in Figure 1. The robot extracts information from camera images, reasons with this information and incomplete domain knowledge, and executes actions to achieve desired outcomes. The robot also learns previously unknown axioms governing domain dynamics, and provides on-demand *explanatory* descriptions of its decisions and beliefs in the form of relations between domain attributes, robot attributes, and robot actions. Let the goal in Figure 1(left) be to have the yellow cylinder on the occluded green block. One plan is to move the objects on the yellow cylinder to the table, move the yellow block (on the green block) to the table, and then move the yellow cylinder on the green block. When asked to justify a plan step, e.g., "why do you want to put the yellow duck on the table first?", the robot answers "the yellow duck is on the yellow cylinder that I need to put on the green block". However, the question "why did you pick up the yellow object?" posed after plan execution is ambiguous in the object and time step being referred to. Recognizing this ambiguity, the robot constructs and poses a question likely to resolve the ambiguity, e.g., "are you referring to the yellow cylinder or yellow block?", and uses the answer to respond to the original query.

US Office of Naval Research (N00014-20-1-2390), Asian Office of Aerospace Research and Development (FA2386-16-1-4071), and the UK Engineering and Physical Sciences Research Council (EP/S032487/1)

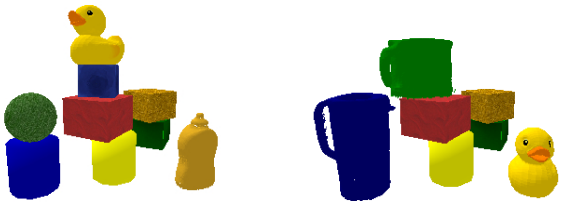

Fig. 1. Simulated scenes in which reference to a "yellow object" is ambiguous.

We seek to enable such on-demand *explanations* of a robot's decisions and beliefs in the form of descriptions of relations between object attributes, actions, and robot attributes. This is difficult to achieve with integrated robot systems that include knowledge-based reasoning and data-driven learning methods. Inspired by cognitive systems research that indicates the benefits of coupling different representations and reasoning schemes [1], [2], our architecture combines the complementary strengths of knowledge representation tools and data-driven methods to provide transparent decision making. It builds on our prior work that combined the principles of non-monotonic logical reasoning and deep learning for scene understanding in simulated images, and demonstrated the use of learned state constraints to partially describe some decisions [3]–[5]. The new contributions of this paper significantly extend the explanation generation capabilities by:

- Using knowledge representation tools to automatically trace the evolution of any given belief by inferring the application of a suitable sequence of known or learned axioms governing domain dynamics.
- Automatically constructing disambiguation queries by introducing new heuristic measures of ambiguity, human confusion, and the relative utility of attributes, to address ambiguity in human queries.

We illustrate the architecture's capabilities in the context of a robot: (i) computing and executing plans to arrange objects in desired configurations; and (ii) estimating occlusion of objects and stability of object configurations, in simulated scenes.

II. Related Work

Early work on explanation generation drew on research in psychology and linguistics to characterize explanations in

terms of generality, objectivity, connectivity, relevance, and content [6]. Studies with human subjects supported these findings [7], and computational methods were developed for explaining unexpected outcomes [8].

There is much interest in understanding the operation of AI and machine learning methods, and making automation more acceptable [9]. Recent work on *explainable AI/planning* can be broadly categorized into two groups. Methods in one group modify or transform learned models or reasoning systems to make decisions interpretable, e.g., by tracing decisions to inputs [10], learning equivalent interpretable models [11], or biasing a planning system towards making decisions easier for humans to understand [12]. Methods in the other group provide descriptions that make decisions more transparent, e.g., describing planning decisions [13], using rules with monotonic operators to define *proof trees* that provide a declarative view (i.e., explanation) of computation [14], or describing solutions obtained through non-monotonic logical reasoning [15]. These methods are often agnostic to how an explanation is structured or assume comprehensive domain knowledge. Methods are also being developed to make the operation of deep networks more interpretable, e.g., by computing gradients and *heat maps* of relevant features [16] or using neuro-symbolic methods to answer questions about images of scenes [17].

To provide explanations in response to a human query, the robot system needs the ability to identify and use the information relevant to the query. The robot can address ambiguity in the query by constructing clarification questions. Researchers have evaluated how the type of question posed by an agent affects the quality of human responses [18], the ability to learn from answers [19], or the ability to minimize ambiguity in the human response [20]. These methods measured the accuracy of the information obtained, or the ability to learn from the human response; they do not jointly explore reasoning with incomplete domain knowledge (to construct questions) and improving the quality of the explanations.

The focus of our work is on integrated robot systems that use knowledge-based and data-driven algorithms to represent, reason with, and learn from incomplete commonsense domain knowledge and noisy observations. We seek to enable such robots to generate accurate relational descriptions of decisions and evolution of beliefs, capabilities that are not supported by existing systems [9], [21].

III. Architecture

Figure 2 depicts the overall architecture for knowledge representation, explainable reasoning, and interactive learning. This architecture first attempts to perform non-monotonic logical reasoning with incomplete commonsense domain knowledge, which is encoded as an Answer Set Prolog (ASP) [22] program, to complete any given visual scene understanding or planning task. If it is unable to do so, it automatically identifies relevant regions of interest (ROIs) in the images to guide the training of deep network models for this task. Information from these ROIs is also used to induce previously unknown axioms that are used for subsequent reasoning. The

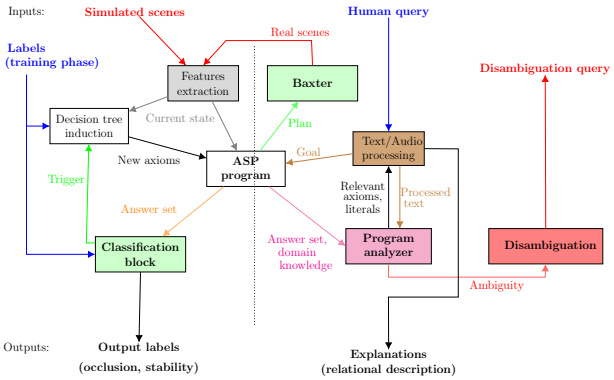

Fig. 2. Architecture combines complementary strengths of non-monotonic logical reasoning, deep learning, and inductive learning. Robot resolves ambiguity in explanatory queries by automatically constructing and posing relevant clarification question(s).

program analyzer takes the parsed human input, and triggers reasoning and/or constructs an explanation of the desired decisions and evolution of beliefs, including the construction of the disambiguation queries. Below, we summarize all components of the architecture, but focus on the new explanation generation capabilities supported by enabling the robot to trace the evolution of beliefs, and by constructing and posing disambiguation questions, while reasoning with and learning contextual information. We do so in the context of the following running example.

Example Domain 1: [Assistive Robot (AR) Domain] Consider a robot analyzing scenes of objects stacked in different configurations to: (i) rearrange object structures according to human requirements; and (ii) provide on-demand, relational descriptions of decisions and beliefs before, during, or after planning and execution. The robot has the ability to visually recognize objects and them to achieve desired object configurations. The robot's prior domain knowledge includes some object attributes such as $size$ (small, medium, large), $surface$ (flat, irregular) and $shape$ (cube, apple, duck), and the spatial $relation$ between objects (above, below, front, behind, right, left, near). It also includes axioms governing domain dynamics but some axioms may be unknown, e.g.:

- Placing an object on top of an object with an irregular surface causes instability;
- Removing all objects in front of an object causes this object to be not occluded.

These axioms can be learned, revised, and used for reasoning.

A. Knowledge Representation and Reasoning

To represent and reason with domain knowledge, we use CR-Prolog, an extension to ASP; we use the terms CR-Prolog and ASP interchangeably. ASP is a declarative language that can represent recursive definitions, defaults, causal relations, and constructs that occur frequently in non-mathematical domains, and are difficult to express in classical logic formalisms.

A domain's description in ASP comprises a *system description* $\mathcal{D}$ and a *history* $\mathcal{H}$. $\mathcal{D}$ comprises a *sorted signature* Σ and axioms. Σ comprises *sorts* arranged hierarchically; *statics*, i.e., domain attributes that do not change over time;

fluents, i.e., domain attributes whose value can be changed; and *actions*. In the AR domain, sorts include $object$, $robot$, $size$, $relation$, $surface$, and $step$. Statics include object attributes such as *obj_size(object, size)* and *obj_surface(obj, surface)*. The Fluents include spatial relations between objects, *obj_relation(relation, object, object)*, e.g., *obj_relation(above, A, B)* implies that object A is *above* object B (the last argument is the reference object), and other relations, e.g., *in_hand(robot, object)*. Actions of the AR domain include *pickup(robot, object)* and *putdown(robot, object, location)*, and relation *holds(fluent, step)* implies that a particular fluent holds true at a particular time step.

The domain's transitions are described by axioms encoded in an action language and translated automatically to ASP statements, e.g., axiom for the AR domain include:

$$\begin{aligned} holds(in\_hand(robot, object), I+1) \ &\leftarrow \\ &occurs(pickup(robot, object), I) \\ holds(obj\_relation(above, A, B), I) \ &\leftarrow \\ &holds(obj\_relation(below, B, A), I) \\ \neg occurs(pickup(robot, object), I) \ &\leftarrow \\ &holds(in\_hand(robot, object), I) \end{aligned}$$

which represent a causal law, a state constraint, and an executability condition respectively. The spatial relations extracted from images are also converted to ASP statements. In addition, we include axioms encoding generic (commonsense) domain knowledge, e.g., statements of the form "larger objects placed on smaller objects are typically unstable".

History $\mathcal{H}$ includes records of observations received and actions executed by the robot at particular time steps. This can be expanded to represent initial state defaults, i.e., statements that are initially assumed to be true in all but a few exceptional circumstances. For example, we encode "a book is usually in the library; if not there, it is usually found in the office", and exceptions, e.g., cookbooks are in the kitchen [23].

To reason with domain knowledge, the robot automatically constructs the CR-Prolog program $\Pi(\mathcal{D}, \mathcal{H})$, including helper axioms for reasoning; our program for the AR domain is in our open-source repository [24]. Planning, diagnostics, and inference can then be reduced to computing *answer sets* of Π; each answer set is a possible world comprising beliefs of the robot associated with Π. We use the SPARC system [25] to compute answer set(s) of ASP programs, and extract relevant literals (e.g., plans) as needed. In other work we combined ASP-based non-monotonic logical reasoning with probabilistic reasoning for more precise action execution at a finer granularity [23]. For ease of understanding and to focus on the interplay between reasoning and learning, we limit ourselves to logical reasoning at a coarser resolution in this paper.

B. Features Extraction, Classification, and Learning

Next we describe the extraction of features from images and their use in performing estimation and learning tasks.

Feature extraction: The main inputs are RGB images of simulated scenes (e.g., Figure 1) with different object configurations. For any image, we extract spatial relations between objects using our prior work on incrementally revising the grounding (i.e., meaning in physical world) of spatial relations encoded by prepositional words such as "above", and "behind" [26]. We also extract object attributes, e.g., color, shape, and size using probabilistic algorithms, with the most likely outcome encoded as ASP statements with complete certainty.

Classification and Learning: The *classification block* comprises three sub-components, and encodes a processing strategy. For any given image, the agent first attempts to address the classification task (e.g., estimate object occlusion and stability of object structures) using ASP-based reasoning with domain knowledge. If an answer is not found, or an incorrect answer is found (during training), the robot automatically extracts relevant regions of interest (ROIs) from the corresponding image. Information from these ROIs is used to train and use a deep (convolutional) neural network, the second sub-component, for the classification task.

Images processed using deep networks are considered to contain information that is missing (or incorrect) in the existing knowledge. Image features and object relations extracted from the ROIs in each such image, along with the ground truth label for occlusion and stability, are used by the third sub-component to incrementally learn a decision tree (during training) that summarizes the experiences of state transitions. Branches in this decision tree with sufficient support among the training examples are used to induce axioms that are merged with existing axioms and used for reasoning.

C. Answering Explanatory Questions

Next, we describe the components that provide explanatory descriptions of decisions and beliefs.

Text and audio interface: Human (verbal) input is first transcribed using existing software [27], labeled using a part-of-speech tagger, and normalized with the lemma list [28] and their synonyms and antonyms retrieved from WordNet [29]. The processed text helps identify the type of request, which may be task execution or an explanation. In the former case, the related goal is passed to the ASP program for planning. In the latter case, the "Program Analyzer" module (see below) automatically infers and extracts relevant literals to compose an answer. These literals are used with generic sentence templates to produce human-understandable (textual) explanations; this can be converted to synthetic speech [30].

Beliefs tracing: Our architecture's ability to construct relational (explanatory) descriptions depends on the ability to infer the sequence of axioms that explain the evolution of any given belief. We adapt existing methods for generating "proof trees" [14] to our non-monotonic (logical) reasoning formulation. For any belief of interest, i.e., a positive or negative literal of a fluent or an action, we proceed as follows:

1) Select axioms whose head matches the belief of interest.
2) Ground literals in the body of each selected axiom and check whether these are supported by the answer set.

3) Create a new branch in a proof tree (with target belief as root) for each axiom supported by the answer set, and store axiom and supporting ground literals in nodes.
4) Repeat Steps 1-3 with the supporting ground literals in Step 3 as target beliefs in Step 1, until all branches reach a leaf with no further supporting axioms.

The paths from the root to the leaves in these trees help construct the desired explanations.

Program Analyzer: Our approach automatically identifies and reasons with relevant information to construct relational descriptions for four types of *explanatory* questions or requests. The first three were introduced as question types to be considered by any explainable planning system [31]; we also consider questions about the robot's beliefs at any point in time. The shortest answer is selected when multiple explanations are found; if multiple explanations of the same complexity exist, one is chosen at random.

1) **Plan description** When asked to describe a plan, the robot parses the related answer set(s) and extracts a sequence of actions such as *occurs(action1, step1), ..., occurs(actionN, stepN)* to construct the response.
2) **Action justification: Why action X at step I?** To justify an action's execution at a particular time step:
   a) For each action that occurred after time step I, the robot examines relevant executability condition(s) and identifies literal(s) that would prevent the action's execution at step I. For picking up the red_block in Figure 1(right), assume that the executed actions are *occurs(pickup(robot, green_mug), 0)*, *occurs(putdown(robot, green_mug, table), 1)*, and *occurs(pickup(robot, red_block), 2)*. If the focus is on the first $pickup$ action, an executability condition related to the second $pickup$ action:

$$\neg occurs(pickup(robot, A), I) \leftarrow holds(obj\_rel(below, A, B), I)$$

   is ground in the scene to obtain *obj_rel(below, red_block, green_mug)* as a literal of interest.
   b) If any identified literal is in the answer set at the time step of interest (0 in this example) and is absent (or its negation is present) in the next step, it is a reason for executing the action of interest.
   c) The condition modified by executing the action of interest is paired with the subsequent action to construct the answer. The question "Why did you pick up the green mug at time step 0?", receives the answer "I had to pick up the red block, and the red block was below the green mug".

   A similar approach is used to justify the selection of any action in a plan that has not been executed.
3) **Hypothetical actions: Why not action X at step I?** For questions about actions not selected:
   a) The robot identifies executability conditions with the hypothetical action in the head, i.e., conditions that prevent selection of the action during planning.
   b) For each such executability condition, if the literals in the body are satisfied by the corresponding answer set, they form the answer.

   Continuing with our example, for the question "Why did you not put the green mug on the yellow duck at time step 1?", the following axiom is identified:

$$\neg occurs(putdown(robot,\ A,\ B), I) \leftarrow has\_surface(B,\ irregular)$$

   which implies that an object cannot be placed on another with an irregular surface. Since the robot knows that the yellow duck has an irregular surface, it answers "Because the yellow duck has an irregular surface".
4) **Belief query: Why belief Y at step I?** To explain any particular belief, the robot uses *belief tracing* to identify supporting axioms and literals that form the answer. For instance, to explain the belief that object ob_1 is unstable in step I, the robot finds the axiom:

$$\neg holds(stable(ob_1), I) \leftarrow holds(small\_base(ob_1), I)$$

   Assume that the current beliefs include that ob_1 has a small base. Tracing this belief identifies the axiom:

$$holds(small\_base(ob_1), I) \leftarrow holds(relation(below,\ ob_2,\ ob_1),\ I), has\_size(ob_2,\ small),\ has\_size(ob_1,\ big)$$

   When asked "why do you believe ob_1 is unstable at step I?", the robot answers "Because ob_2 is below ob_1, ob_2 is small, and ob_1 is big".

D. Disambiguation Queries

Questions posed by humans may be ambiguous in terms of the objects and time steps that they refer to. Our architecture enables the robot to address this ambiguity by constructing and posing clarification questions based on the domain attributes. Since different disambiguation queries can be formed based on the attributes characterizing domain objects, our approach draws inspiration from findings in psychology and cognitive science [6], [7] to construct queries most likely to address the ambiguity. This approach is based on three heuristic measures applied in the following order:

1) **Unambiguity:** this measure selects attributes that match the least number of ambiguous objects in the context of the human query and the current scene.
2) **Human confusion:** based on the understanding that queries with many attributes are more likely to confuse a human, this measure selects questions based on as few attributes as possible.
3) **Attribute/Feature rank:** this measure identifies questions comprising more "useful" attributes. The rank of

TABLE I
EXAMPLE OF COMPUTING THE RANK OF ATTRIBUTES.

features	human preference	detection complexity	rank
Color	0.5	0.9	0.66
Size	0.3	0.8	0.5
Shape	0.2	0.6	0.36

each attribute is a linear combination of its *human preference* and *detection complexity*:

$$\textbf{Attribute rank} \;=\; \alpha \times (human\ preference) \;+\; \beta \times (detection\ complexity).$$

where the values of α and β are dynamically updated to reflect the relative importance of the measures. Here, *human preference* seeks to capture the preference of humans to use certain attributes for describing certain objects, whereas the *detection complexity* reflects the level of difficulty a robot has in detecting each attribute. These are domain-specific measures whose values are determined from prior knowledge and statistics collected in an initial training phase. For instance, suppose a robot is able to detect *color*, *size* and *shape* of objects, and the current values for α and β are 0.6 and 0.4 respectively. Illustrative examples of the values of *human preference* and *detection complexity*, computed experimentally, and the resulting *rank* of each attribute, are summarized in Table I. The computed values can also be revised over time, although we do not do so in our experiments.

A simple method for constructing disambiguation queries would be to consider all possible combinations of attributes not included in the human (input) query. The three measures could then be applied to these queries to select the question(s) to be posed to address the ambiguity. Such an approach would be computationally expensive in complex domains. Instead, our architecture uses the belief tracing approach to automatically identify information that can be used to address the current ambiguity. Recall that any human query is translated to literals compatible with the current knowledge (i.e., the knowledge base). For simplicity, assume that there is a single such literal; this literal is ground for each entity in the current scene that matches the query. The negation of such literals are used as the initial beliefs in the beliefs tracing approach to identify information not supported by the knowledge base. For instance, in the scene in Figure 1 (right), the human request "Put the green mug on the yellow object", is ambiguous since there are three yellow objects in the scene. Our approach finds three negated action literals of interest: $\neg\ occurs(putdown(rob1, mug, yellow\_duck), I)$, $\neg\ \ occurs(putdown(rob1, mug, yellow\_cylinder), I)$, and $\neg\ occurs(putdown(rob1, mug, yellow\_block), I)$. Since the first two literals are supported by the knowledge base, i.e., the robot knows these actions cannot be executed in the current state given the existing axioms, the robot prioritizes the yellow cube as being the object of interest and biases the disambiguation question towards confirming this intuition; in this example, the candidate disambiguation query is: "Do you want the mug on top of the yellow block?". Section IV-B describes an example of using this approach.

IV. EXPERIMENTAL SETUP AND RESULTS

We first describe the setup for evaluating the architecture's abilities (Section IV-A), followed by the execution traces (Section IV-B) and quantitative results (Section IV-C).

A. Experimental Setup

The reasoning and learning capabilities of our baselines architecture have been described in our prior work [3], [4]. Here we focus on belief tracing and generation of disambiguation queries, and evaluate the following hypotheses:

H1 : Our disambiguation approach reduces the number of queries posed by the robot and the attributes used in the queries, and increases the accuracy of the explanatory responses after the first disambiguation question; and

H2 : The contextual information retrieved by belief tracing enables the robot to construct queries better suited to address the ambiguity in the human query or request.

Experimental trials considered simulated images of the AR domain. We used a real-time physics engine (Bullet) to create 200 simulated images, each with $7-15$ objects stacked in different configurations or spread on a flat surface. Objects included cylinders, spheres, cubes/blocks, a duck, and five household objects from the Yale-CMU-Berkeley dataset (apple, pitcher, mustard bottle, mug, and box of crackers). We considered questions containing $2-10$ ambiguous objects, with $2-10$ attributes available for disambiguation. One hundred images containing up to 10 objects were used for questions containing up to six ambiguous entities whereas the other 100 were used for question with more than six ambiguous entities. For each ambiguous question, we stored the number of attributes and interactions required, and the accuracy in the robot's responses after posing the disambiguation queries. We compared the proposed method with a baseline approach that randomly and incrementally selects attributes from the set of attributes considered to be relevant to the original human query, until the ambiguity is eliminated or all available attributes are exhausted. The baseline initially uses the same number of attributes as our method, and then adds one attribute at a time; each such addition is considered an additional interaction.

B. Execution Trace

The following execution traces illustrate our architecture's ability to construct and use disambiguation queries, and to provide relational descriptions in response to the human queries.

Execution Example 1: [Disambiguation Example]
Consider the following interaction in the scenario in Figure 1(right); object attributes are color, shape, and size.

- **Human**: "Please pick up the yellow object."
  This is an ambiguous statement because there are multiple yellow objects in the scene.
- The baseline disambiguation strategy randomly chooses and uses one of the two unused attributes to ask a clarification question:
  Robot: "What is the size of the yellow object?"
  In this case, the three yellow objects are of comparable

size (medium), so the robot would need at least one more question for disambiguation.
- Our disambiguation approach chooses the best attributes to construct queries. Assume that all possible combinations of the two unused attributes are considered to construct candidate disambiguation queries, i.e., it considers size, shape, and size and shape respectively.
- Using the *unambiguity* measure, the robot chooses attribute(s) resulting in the least number of matching entities. Since yellow objects are of a similar size (medium), no candidate query is constructed based just on size.
- Based on the *human confusion* measure, the robot seeks to construct queries based on the minimum number of attributes. In our example, the candidate query containing only the shape attribute is preferred over the other combining size and shape. As a result, only one disambiguation question is constructed:
  Robot: "What is the shape of the yellow object?"
- Only two measures were used to select a disambiguation query in this example. However, when two or more queries are constructed in more complex situations, the third (*attribute/feature rank*) measure will also be used.

Execution Example 2: [Disambiguation and Axioms]
We continue with the previous example as described below.

- **Human**: "Please move the mug on top of the yellow object." Similar to Execution Example 1, this statement is ambiguous because the robot is unsure which of the three yellow objects the human is referring to.
- Unlike Execution Example 1, we consider the known axioms to provide contextual information that reduces the search space for constructing disambiguation queries.
- Assume that the robot knows the following axioms:

$$\neg holds(stable(Ob1), I) \leftarrow holds(obj\_relation(above, Ob_1, Ob_2), I), has\_surface(Ob_2, irregular) \quad (2a)$$

$$\neg occurs(putdown(rob_1, Ob_1, Obj2), I) \leftarrow holds(obj\_relation(below, Ob_2, Ob_3), I) \quad (2b)$$

  Statement 2(a) eliminates the duck as a possible target location (for the mug) since it is known to have an irregular surface. Statement 2(b) favors the yellow block (on top of the green block) as the possible target location.
- It is possible to place the mug on the yellow cylinder after removing the red block, but the yellow block offers a simpler solution (based on the unambiguity measure). The following disambiguation query is thus constructed:
  Robot: "Should I move the mug on top of yellow block?"

Execution Example 3: [Disambiguation and Explanation]
Next, consider a different request in the same scenario.

- **Human**: "Move the yellow object on the green cube."
  Once again, there is ambiguity in the reference to an yellow object. As the yellow cube is already in the desired position, and the yellow cylinder is below other objects, the yellow duck would be the simpler solution based on the *ambiguity* measure, resulting in the question:
  Robot: "Should I move the yellow duck on top of the green cube?"
  Human: "No. Please move the yellow cylinder on top of the green cube."
- To attend to this request, the robot computes and executes the plan: *pick up the mug; put down the mug on the table; pick up the red cube; put down the red cube on the table; pick up the yellow cube; put down the yellow cube on the table; pick up the yellow cylinder; put down the yellow cylinder on the top of the green cube*.
- The human asks a question related to the plan:
  Human: "Why did you put the cube on the table?"
  This is an ambiguous question because the red and yellow cubes have been moved to the table at different time steps. Since these cubes have similar size and shape, color is used automatically for disambiguation.
  Robot: "What is the color of the cube?"
  Human: "Yellow."
- The question about the executed plan is now answered.
  Robot: "I had to put the yellow cylinder on top of the green cube. The green cube was below the yellow cube."
- The human asks about a particular belief.
  Human: "Why did you believe the green cube was below the yellow cube?"
  Robot: "Because I observed the green cube below the yellow cube at step zero."

C. Experimental Results

Next, we discuss quantitative results of evaluating the hypotheses listed in Section IV-A. Unless stated otherwise, *all claims are statistically significant at the 95% significance level*. The first set of experiments was designed to evaluate hypothesis **H1**:

1) 100 initial object configurations were constructed randomly. The information extracted from each image (e.g., object attributes, spatial relations) was included as initial state in the corresponding ASP program.
2) For each initial state, we considered questions in which $2-10$ objects were ambiguous, and $2-10$ attributes were available for the constructing disambiguation queries.
3) The number of attributes used for disambiguation was recorded for the baseline algorithm and for our algorithm. When a sufficient number of attributes were not available for disambiguation, all the attributes were considered to be available.
4) We ran the baseline for the same 100 scenes mentioned above, and considered any extra attribute needed in addition to the number of attributes required by our disambiguation approach as an extra interaction.

Recall that for both the baseline and our approach, only attributes related to the objects in the ambiguous queries are considered. The average number of attributes used is reported as a function of the number of ambiguous objects in Figure 3. The average number of interactions as a function of the number of ambiguous objects is in Figure 4. Our

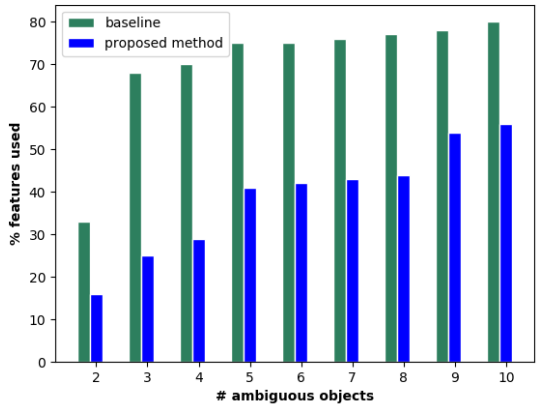

Fig. 3. Percentage of attributes used in disambiguation queries expressed as a function of the number of ambiguous objects in the scene. Our proposed method uses a smaller number of attributes.

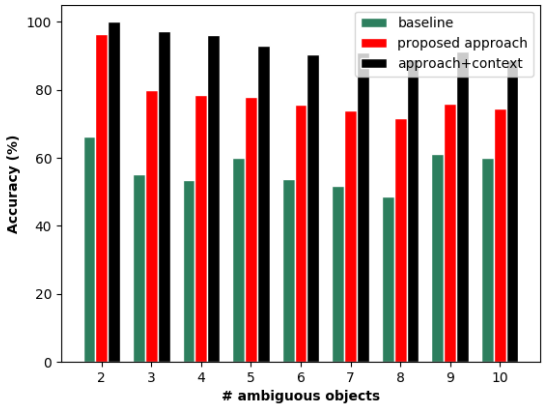

Fig. 5. Accuracy of answers provided by the robot after constructing disambiguation queries using the baseline and the proposed method (with and without contextual knowledge). Using the proposed method improves accuracy, and using contextual information further improves performance.

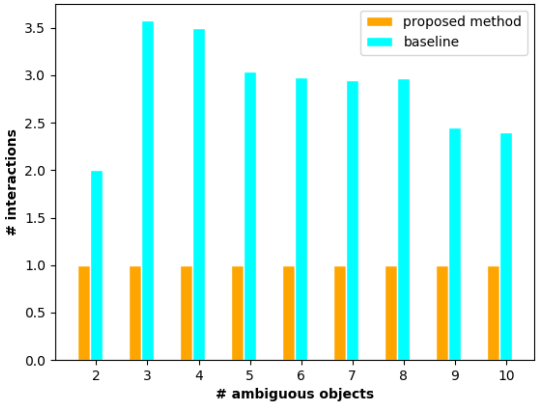

Fig. 4. Average number of interactions required for disambiguation. Our proposed approach uses a significantly smaller number of interactions compared with the baseline.

method significantly reduces the number of attributes used for disambiguation. Also, the baseline approach requires at least two interactions to achieve the expected response whereas our method typically requires only one. These results support **H1**.

The second set of experiments was designed as follows to evaluate hypotheses **H1** and **H2**:

1) 100 initial object configurations were constructed randomly. The information extracted from each such image (e.g., object attributes, spatial relations) was encoded as initial state in the corresponding ASP program.
2) For each initial state, we considered questions with $2-10$ ambiguous objects, and $2-10$ attributes available for the construction of disambiguation queries.
3) The accuracy for the answers provided by the robot (to the original question) after asking the disambiguation question is computed for the baseline and our proposed method; with out method, we also computed accuracy with and without contextual knowledge identified through reasoning (and belief tracing) with existing knowledge. Results are plotted in Figure 5 as *baseline*, *proposed*, and *proposed+context* respectively.

Figure 5 indicates that our method improves accuracy of the responses, which further supports **H1**. We also observe that extracting and using the contextual information to construct the disambiguation queries helps obtain useful information from the human, improving the accuracy of the answers to the original human queries. These results support **H2**.

V. Conclusion

The architecture described in this paper is a step towards greater transparency in reasoning and learning for integrated robot systems that include methods for knowledge-based reasoning and data-driven learning. Our architecture exploits the complementary principles and strengths of non-monotonic logical reasoning with incomplete commonsense domain knowledge, data-driven deep learning from a limited set of examples, and inductive learning of previously unknown axioms governing domain dynamics. We also described a strategy to trace the evolution of beliefs, and construct and pose suitable disambiguation queries. Experimental results using simulated images demonstrate the ability to construct suitable disambiguation queries and provide more accurate relational descriptions (as explanations) of decisions and the evolution of beliefs in response to the human queries.

Our architecture presents multiple directions for further research. We will further explore the interplay between reasoning and learning in the context of explaining decisions and beliefs while performing scene understanding and planning tasks in more complex domains. We will also investigate the use of our architecture on a robot interacting with humans in the physical (i.e., real) world through noisy sensors and actuators, building on other work in our group on combining non-monotonic logical reasoning with probabilistic reasoning at different resolutions [23]. The longer-term objective is to support explainable reasoning and learning in integrated robot systems in complex domains.

REFERENCES

[1] J. E. Laird, *The Soar Cognitive Architecture*. The MIT Press, 2012.

[2] P. H. Winston and D. Holmes, "The Genesis Enterprise: Taking Artificial Intelligence to Another Level via a Computational Account of Human Story Understanding," Massachusetts Institute of Technology, Computational Models of Human Intelligence Report 1, December 2018.

[3] T. Mota and M. Sridharan, "Commonsense Reasoning and Knowledge Acquisition to Guide Deep Learning on Robots," in *Robotics Science and Systems*, 2019.

[4] T. Mota, M. Sridharan, and A. Leonardis, "Commonsense reasoning and deep learning for transparent decision making in robotics," in *European Conference on Multiagent Systems*, 2020.

[5] ——, "Integrated Commonsense Reasoning and Deep Learning for Transparent Decision Making in Robotics," *Springer Nature Computer Science*, vol. 2, no. 242, pp. 1–18, 2021.

[6] M. Friedman, "Explanation and scientific understanding," *Philosophy*, vol. 71, no. 1, pp. 5–19, 1974.

[7] S. J. Read and A. Marcus-Newhall, "Explanatory coherence in social explanations: A parallel distributed processing account," *Personality and Social Psychology*, vol. 65, no. 3, p. 429, 1993.

[8] J. de Kleer and B. C. Williams, "Diagnosing Multiple Faults," *Artificial Intelligence*, vol. 32, pp. 97–130, 1987.

[9] T. Miller, "Explanations in Artificial Intelligence: Insights from the Social Sciences," *Artificial Intelligence*, vol. 267, pp. 1–38, 2019.

[10] P. W. Koh and P. Liang, "Understanding Black-box Predictions via Influence Functions," in *International Conference on Machine Learning*, 2017, pp. 1885–1894.

[11] M. Ribeiro, S. Singh, and C. Guestrin, "Why Should I Trust You? Explaining the Predictions of Any Classifier," in *International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1135–1144.

[12] Y. Zhang, S. Sreedharan, A. Kulkarni, T. Chakraborti, H. H. Zhuo, and S. Kambhampati, "Plan explicability and predictability for robot task planning," in *International Conference on Robotics and Automation*, 2017, pp. 1313–1320.

[13] R. Borgo, M. Cashmore, and D. Magazzeni, "Towards Providing Explanations for AI Planner Decisions," in *IJCAI Workshop on Explainable Artificial Intelligence*, 2018, pp. 11–17.

[14] G. Ferrand, W. Lessaint, and A. Tessier, "Explanations and Proof Trees," *Computing and Informatics*, vol. 25, pp. 1001–1021, 2006.

[15] J. Fandinno and C. Schulz, "Answering the "Why" in Answer Set Programming: A Survey of Explanation Approaches," *Theory and Practice of Logic Programming*, vol. 19, no. 2, pp. 114–203, 2019.

[16] R. Assaf and A. Schumann, "Explainable Deep Neural Networks for Multivariate Time Series Predictions," in *International Joint Conference on Artificial Intelligence*, Macao, China, July 2019, pp. 6488–6490.

[17] K. Yi, J. Wu, C. Gan, A. Torralba, P. Kohli, and J. B. Tenenbaum, "Neural-Symbolic VQA: Disentangling Reasoning from Vision and Language Understanding," in *Neural Information Processing Systems*, Montreal, Canada, December 3-8, 2018.

[18] S. Rosenthal, M. Veloso, and A. K. Dey, "Acquiring accurate human responses to robots' questions," *International journal of social robotics*, vol. 4, no. 2, pp. 117–129, 2012.

[19] V. Gonzalez-Pacheco, M. Malfaz, A. Castro-Gonzalez, J. C. Castillo, F. Alonso, and M. A. Salichs, "Analyzing the impact of different feature queries in active learning for social robots," *International Journal of Social Robotics*, vol. 10, no. 2, pp. 251–264, 2018.

[20] B. Myagmarjav and M. Sridharan, "Incremental knowledge acquisition for human-robot collaboration," in *2015 24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2015, pp. 809–814.

[21] S. Anjomshoae, A. Najjar, D. Calvaresi, and K. Framling, "Explainable agents and robots: Results from a systematic literature review," in *International Conference on Autonomous Agents and Multiagent Systems*, Montreal, Canada, 2019.

[22] M. Gelfond and Y. Kahl, *Knowledge Representation, Reasoning and the Design of Intelligent Agents*. Cambridge University Press, 2014.

[23] M. Sridharan, M. Gelfond, S. Zhang, and J. Wyatt, "REBA: A Refinement-Based Architecture for Knowledge Representation and Reasoning in Robotics," *Journal of Artificial Intelligence Research*, vol. 65, pp. 87–180, May 2019.

[24] T. Mota and M. Sridharan, "Scene Understanding, Reasoning, and Explanation Generation," 2020, https://github.com/tmot987/Scenes-Understanding.

[25] E. Balai, M. Gelfond, and Y. Zhang, "Towards answer set programming with sorts," in *International Conference on Logic Programming and Nonmonotonic Reasoning*, 2013.

[26] T. Mota and M. Sridharan, "Incrementally Grounding Expressions for Spatial Relations between Objects," in *International Joint Conference on Artificial Intelligence*, 2018, pp. 1928–1934.

[27] A. Zhang, "Speech recognition (version 3.8)," 2017.

[28] Y. Someya, "Lemma List for English Language," 1998.

[29] G. A. Miller, "WordNet: a lexical database for English," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[30] N. M. Bhat, "Pyttsx3: Text-to-speech x-platform," 2018.

[31] M. Fox, D. Long, and D. Magazzeni, "Explainable Planning," in *IJCAI Workshop on Explainable AI*, 2017.