

# Integrated Knowledge-based Reasoning and Data-driven Learning for Explainable Agency in Robotics

Mohan Sridharan

Intelligent Robotics Lab  
School of Computer Science, University of Birmingham, UK  
m.sridharan@bham.ac.uk

## Abstract

This book chapter describes an integrated architecture for robots that combines the complementary strengths of knowledge-based and data-driven methods for transparent reasoning and learning. Specifically, the architecture builds on the principle of step-wise iterative refinement to support non-monotonic logical reasoning and probabilistic reasoning with tightly-coupled transition diagrams of the domain at different resolutions. Reasoning with prior domain knowledge triggers and guides the interactive learning of previously unknown domain knowledge in the form of axioms governing domain dynamics. Furthermore, the interplay between these components is used to embed the principles of explainable agency, enabling a robot to provide on-demand relational descriptions of its decisions and beliefs in response to different types of questions. The architecture’s capabilities are evaluated in the context of visual scene understanding and planning tasks performed in simulation and on physical robots.

## 1 Motivation

Consider an assistive robot that has to: (a) estimate the occlusion of objects and stability of object configurations in specific scenes—Figure 1; and (b) compute and execute plans to achieve desired configurations. To perform these tasks, the robot extracts information from on-board sensors (e.g., camera), and reasons with this information and prior domain knowledge. The uncertainty in its perception and actuation is represented probabilistically, e.g., “I am 90% certain I saw the robotics book in the study”. The robot’s prior knowledge includes knowledge of some domain attributes (e.g., the arrangement of rooms); some object attributes (e.g., *shape*, *surface*); grounding of some prepositional words (e.g., *above*, *in*) that represent the spatial relations between objects; some axioms governing actions and change in the domain (e.g., “picking up an object will cause it to be in the robot’s hand”); and default statements (e.g., “books are usually in the study”) that hold in all but a few exceptional circumstances (e.g., “cookbooks are in the kitchen”). Furthermore, the existing knowledge has to be revised over time, and the robot has to answer questions about its decisions and beliefs during or after planning and execution. For example, if the goal in Figure 1 (right) is to have the



Figure 1: Scenario for some robot experiments.

yellow ball on the orange block, and the plan is to move the blue block to the table before placing the ball on the orange block, the robot may be asked “why do you want to pick up the blue block first?”, “why did you not pick up the pig?”, or “what would happen if you rolled the ball?”.

Our architecture seeks to jointly address the knowledge representation, reasoning, learning, and control challenges posed by the motivating scenario. In this chapter, we focus on the ability to provide on-demand *explanations* of decisions and beliefs in the form of relational descriptions of relevant objects, object attributes, actions, and robot attributes. Providing such explanations can help improve the algorithms and establish accountability, but it is difficult to do so in integrated robot systems that use knowledge-based reasoning methods (e.g., for planning) and data-driven learning methods (e.g., for object recognition). It requires the associated architecture to support the key functional capabilities of *explainable agency*, e.g., provide on-demand justification of decisions made during (or after) plan generation and execution by considering alternative choices; present information at a suitable level of abstraction; and communicate information such that it makes contact with human concepts such as beliefs and goals (Langley et al. 2017). Our architecture draws on cognitive systems research, which highlights the benefits of coupling different representations, reasoning schemes, and learning methods (Laird 2012; Winston and Holmes 2018), to implement these functional capabilities. Specifically, our architecture:

- Combines the principles of non-monotonic logical reasoning and deep learning for decision making, and automatically learns previously unknown axioms of state constraints, action preconditions, and action effects;
- Leverages the interplay between representation, reasoning, and learning to embed the principles of explainable agency, enabling a robot to provide on-demand relational descriptions of its decisions and beliefs.

These capabilities are evaluated in the context of a robot arranging objects in desired configurations, and estimating occlusion of objects and stability of object configurations, in simulated scenes and in the real world. Results indicate the ability to: (i) incrementally learn previously unknown axioms governing domain dynamics; and (ii) construct explanations reliably and efficiently by automatically identifying and reasoning with the relevant knowledge. We begin with a discussion of related work (Section 2), followed by a description of the architecture (Section 3), some experimental results (Section 4), and conclusions (Section 5).

## 2 Related Work

Early work on explanation generation drew on research in cognition, psychology, and linguistics to characterize explanations in terms of generality, objectivity, connectivity, relevance, and information content (Friedman 1974). Subsequent studies involving human subjects have also indicated that the important attributes of good explanations include coherence, simplicity, generality, soundness, and completeness (Read and Marcus-Newhall 1993). In parallel, fundamental computational methods were developed for explaining unexpected outcomes by reasoning logically about potential causes (de Kleer and Williams 1987).

In recent years, the increasing use of AI methods in different domains has renewed the interest in understanding the decisions of these methods, with many dedicated workshops and special tracks at premier conferences. This understanding can be used to improve the underlying algorithms, and to make automated decision-making more acceptable or trustworthy to humans (Anjomshoae et al. 2019; Miller 2019). Existing work in *explainable AI* can be broadly grouped into two categories (Sreedharan, Kulkarni, and Kambhampati 2022a). Methods in one category modify or map learned models or reasoning systems to make their decisions more interpretable, e.g., by tracing decisions back to input data (Koh and Liang 2017) or explaining the predictions of any classifier by learning equivalent interpretable models (Ribeiro, Singh, and Guestrin 2016), or biasing a planning system towards making decisions easier for humans to understand (Chakraborti, Sreedharan, and Kambhampati 2018). The other category of methods provide descriptions that make a reasoning system’s decisions more transparent, e.g., explaining planning decisions (Borgo, Cashmore, and Magazzeni 2018), providing causal and temporal relations (Seegebarth et al. 2012), or reconciling the differences between the planner’s decisions and the human expectations (Sreedharan, Kulkarni, and Kambhampati 2022b). Much of this research is agnostic to how an explanation is structured or assumes comprehensive domain knowledge.

Since deep networks represent the state of the art for different robotics/AI problems, methods have been developed specifically to understand the operation of these networks, e.g., by computing a heatmap of features most relevant to a deep network’s outputs (Assaf and Schumann 2019). There has also been work on reasoning with learned symbolic structure, or with a learned graph encoding scene

structure, in conjunction with deep networks to answer questions about images of scenes (Norcliffe-Brown, Vafeais, and Parisot 2018; Yi et al. 2018). However, these approaches do not fully integrate reasoning and learning to inform and guide each other; or use the rich commonsense domain knowledge for reliable and efficient reasoning, learning, and the generation of explanations.

This chapter focuses on integrated robot systems that combine knowledge-based and data-driven methods to reason with and learn from incomplete commonsense domain knowledge and observations. We describe an architecture that enables such a robot to generate relational descriptions of its decisions and beliefs in response to different kinds of questions. This architecture builds on our previous refinement-based architecture that represents and reasons at two coupled resolutions (Sridharan et al. 2019). It implements a theory of explanations (Sridharan and Meadows 2019), and leverages the interplay between knowledge-based reasoning and data-driven learning (Sridharan and Mota 2023), to enable the functional capabilities of explainable agency (Langley et al. 2017).

## 3 Architecture Description

Our architecture encodes the principle of *step-wise iterative refinement*. It is based on tightly-coupled transition diagrams at different resolutions. These transition diagrams are described using an action language  $\mathcal{AL}_d$  (Gelfond and Incelesan 2013), which has a sorted signature with statics, fluents, and actions, and supports causal laws, state constraints, and executability conditions; the fluents can be non-Boolean and axioms can be non-deterministic.

The architecture may be viewed as a logician, statistician, and an explorer working together, as shown in Figure 2. For ease of understanding, we will limit our discussion of the architecture to two resolutions. For any given goal, the logician performs non-monotonic logical reasoning at the coarse resolution based on commonsense domain knowledge to provide a sequence of abstract actions. Each abstract transition is implemented by the statistician as a sequence of finer-granularity actions, incorporating probabilistic models of uncertainty (e.g., in perception) and communicating the outcomes to the logician. In addition, the explorer revises the existing knowledge (e.g., of action capabilities) when needed (e.g., in response to unexpected action outcomes). The interplay between the architecture’s components enables the desired functional capabilities of explainable agency, with the robot providing relational descriptions of its decisions and beliefs at the desired resolution in response to queries from a human. We will use the following example to describe the architecture’s components.

**Example: Assistive Robotics (AR) Domain.** A robot: (i) estimates occlusion of scene objects and stability of object structures, and computes and executes plans to achieve desired object configurations; and (ii) provides on-demand relational descriptions of decisions and evolution of beliefs. There is uncertainty in the robot’s perception and actuation; probabilistic algorithms are used for visual object recognition and to move objects. The robot has some prior domain

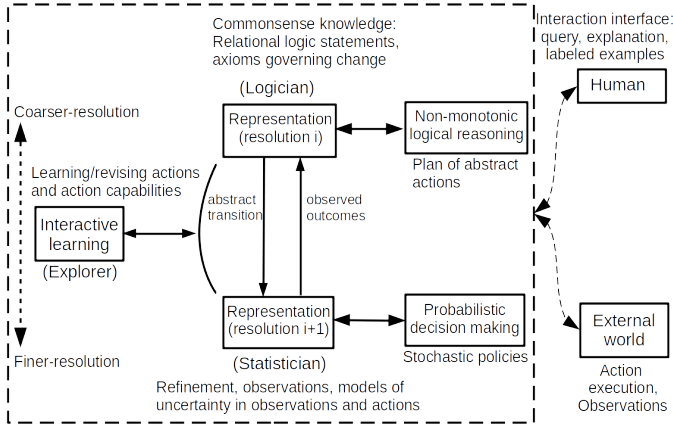


Figure 2: Our architecture leverages strengths of declarative programming, probabilistic reasoning, and interactive learning to represent, reason, and learn at different resolutions.

knowledge, which includes object attributes such as *shape* and *surface*; spatial relations between objects (e.g., above, below, behind, in); some domain attributes; and some axioms governing domain dynamics such as:

- Placing an object on top of an object with an irregular surface results in an unstable object configuration.
- Removing all objects blocking the view of an object’s frontal face causes this object to be not occluded.
- An object below another object cannot be picked up.

This knowledge may need to be revised over time, e.g., some axioms and the value of some attributes may be unknown or may change, as described in Section 3.2.

### 3.1 Knowledge Representation and Reasoning

The coarse resolution domain description comprises system description  $\mathcal{D}_c$  of transition diagram  $\tau_c$ , a collection of  $\mathcal{AL}_d$  statements, and history  $\mathcal{H}_c$ . The subscript “c” refers to the coarse resolution.  $\mathcal{D}_c$  comprises sorted signature  $\Sigma_c$  and axioms. For the AR domain,  $\Sigma_c$  includes basic sorts such as *place*, *thing*, *robot*, *person*, *object*, *cup*, *surface*, and *step*; statics such as *next\_to(place, place)* and *obj\_surface(obj, surface)*; fluents such as *loc(thing, place)*, *obj\_rel(relation, object, object)*, and *in\_hand(entity, object)*; and actions such as *move(robot, place)* and *give(robot, object, person)*. Axioms in  $\mathcal{D}_c$  (for the AR domain) are statements such as:

*move(rob<sub>1</sub>, P)* **causes** *loc(rob<sub>1</sub>, P)*  
*putdown(rob<sub>1</sub>, Ob<sub>1</sub>, Ob<sub>2</sub>)* **causes** *obj\_rel(on, Ob<sub>1</sub>, Ob<sub>2</sub>)*  
*loc(O, P)* **if** *loc(rob<sub>1</sub>, P)*, *in\_hand(rob<sub>1</sub>, O)*  
**impossible** *give(rob<sub>1</sub>, O, P)* **if** *loc(rob<sub>1</sub>, L<sub>1</sub>)*  $\neq$  *loc(P, L<sub>2</sub>)*

that correspond to two causal laws, a state constraint, and an executability condition respectively. We also include axioms in  $\mathcal{D}_c$  to encode theories of intention and affordance.

The history  $\mathcal{H}_c$  of a dynamic domain is typically a record of fluents observed to be true or false at a particular time step, *obs(fluent, boolean, step)*; and of actions

that “happened”, i.e., were executed at a particular time step, *hpd(action, step)*. This definition is expanded to represent prioritized defaults describing the values of fluents in the initial state, e.g., “books are usually in the library; if not there, they are in the office”, along with exceptions (if any).

To reason with domain knowledge, we construct program  $\Pi(\mathcal{D}_c, \mathcal{H}_c)$  in CR-Prolog, a variant of Answer Set Prolog (ASP) that incorporates consistency restoring (CR) rules (Gebser et al. 2012).  $\Pi$  includes  $\Sigma_c$  and axioms of  $\mathcal{D}_c$ , inertia axioms, reality checks, closed world assumptions for actions, and observations, actions, and defaults from  $\mathcal{H}_c$ .  $\Pi$  also includes statements encoding information extracted from sensor inputs (e.g., spatial relations, object attributes) with sufficiently high probability. ASP is based on stable model semantics, and supports *default negation*, *epistemic disjunction*, and non-monotonic logical reasoning. Unlike “ $\neg a$ ” that states *a* is believed to be false, “*not a*” only implies *a* is not believed to be true, i.e., each literal can be true, false or *unknown*. An *answer set* of  $\Pi$  represents the beliefs of the robot associated with  $\Pi$ , with the literals at each time step representing the corresponding *state*. The non-monotonic logical reasoning ability enables recovery from incorrect inferences drawn due to reasoning with incomplete knowledge or noisy sensor inputs. Entailment, planning, and diagnostics can be reduced to computing answer sets of  $\Pi$ ; we do so using the SPARC system (Balai, Gelfond, and Zhang 2013).

For any given goal, reasoning at the coarse-resolution provides a plan of *abstract actions*. To implement the abstract actions, we define a fine-resolution system description  $\mathcal{D}_f$  as a *refinement* of  $\mathcal{D}_c$  such that any given abstract transition between two states in  $\tau_c$  has a path in  $\tau_f$  between a refinement of these two states. In the AR domain, a robot would (for example) reason about grid cells in rooms and parts of objects, attributes previously abstracted away by the designer. To support interaction with the physical world, we extend  $\mathcal{D}_f$  by introducing a *theory of observation* that encodes *knowledge-producing* actions and fluents, and non-determinism ( $\mathcal{D}_{fr}$ ). Since reasoning with  $\mathcal{D}_{fr}$  can become computationally intractable for complex domains, we enable the robot to automatically *zoom* to  $\mathcal{D}_{fr}(T)$ , the part of  $\mathcal{D}_{fr}$  relevant to any given abstract transition  $T$ . Reasoning with  $\mathcal{D}_{fr}(T)$  provides a sequence of concrete actions that implement  $T$ , incorporating relevant probabilistic models of uncertainty (e.g., in perception or in the outcomes of executed actions) as appropriate. Fine-resolution outcomes with a high probability are committed to the fine-resolution history, and the corresponding coarse-resolution outcomes are added to  $\mathcal{H}_c$ . For a more detailed description of refinement and zooming, and the use of such a knowledge representation and reasoning architecture on physical robots, please see (Sridharan et al. 2019). For an extension of this architecture to encode an adaptive theory of intentions in the coarse resolution, please see (Gomez, Sridharan, and Riley 2021).

### 3.2 Interactive Learning

Reasoning with incomplete knowledge, e.g., to fetch target objects or estimate the occlusion of objects, can result in incorrect outcomes. The state of the art for learning previously

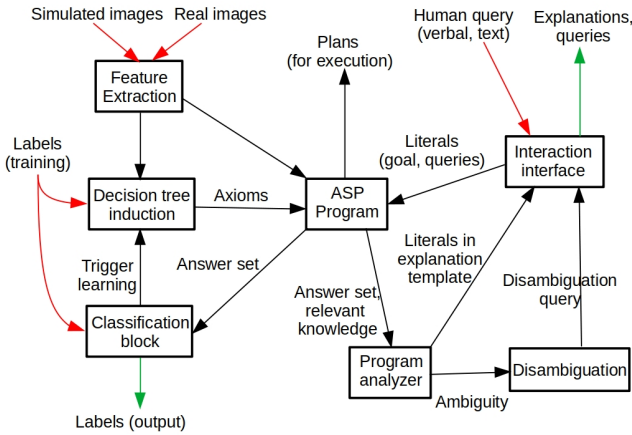


Figure 3: Non-monotonic logical reasoning guides interactive (e.g., deep, inductive) learning of previously unknown domain knowledge to complete desired estimation tasks, and to provide relational descriptions of knowledge and beliefs.

unknown actions, axioms, or object models, are based on “end to end” data-driven methods that require many labeled examples; it is difficult to provide such examples in complex domains or to interpret the decisions made. Figure 3 shows the components for learning and explanation generation in our architecture; red (green) arrows denote inputs (outputs) from (to) humans. The robot first attempts to use ASP-based logical reasoning to complete the desired tasks (e.g., planning, estimation). If this reasoning does not provide any outcome (e.g., no plan), or provides an incorrect outcome (e.g., incorrect stability label), it is considered to indicate that the knowledge is incomplete or incorrect, triggering learning. The learning component’s implementation is described below in the context of the estimation tasks.

**Classification Block (CNNs).** The main sensor inputs are RGB/D images that are processed to extract spatial relations and other attributes that are encoded as ASP statements. The extraction of spatial relations is based on our prior work that incrementally revises the physical world grounding of prepositional words (e.g., “in”, “above”) representing these relations (Mota and Sridharan 2018). For any given image, the robot tries to estimate the occlusion and stability of objects using ASP-based reasoning. If an answer is not found, or an incorrect answer is found (for training images), the robot automatically extracts relevant regions of interest (ROIs) from the image. Parameters of existing Convolutional Neural Network (CNN) architectures (e.g., Lenet (Lecun et al. 1998), AlexNet (Krizhevsky, Sutskever, and Hinton 2012)) are tuned to map information from each ROI to the corresponding labels. The robot automatically identifies and reasons with the relevant axioms and relations to determine the ROIs; the notion of relevance is also expanded to construct explanations efficiently in Section 3.3.

**Decision Tree induction.** The images used to train the CNNs are considered to contain information about missing or incorrect constraints related to the estimation tasks (occlusion, stability). Image features and spatial relations ex-

tracted from ROIs in each such image, and the known occlusion and stability labels (during training), are used to incrementally learn a decision tree summarizing the corresponding state transitions; this process repeatedly splits nodes based on unused attributes likely to provide the highest entropy reduction. Trees are learned separately for different actions, and branches of a tree that satisfy minimal thresholds on purity at the leaf and on the level of support from labeled examples, are used to construct candidate constraints. Candidates without a minimal level of support on unseen examples are removed and similar axioms are merged. Specifically, axioms with the same head and some overlap in the body are grouped. Each combination of one axiom from each group is encoded in an ASP program along with axioms that are not in any group. This program is used to classify ten labeled scenes, only retaining axioms in the program that provides the highest accuracy on these scenes. Also, axioms that cease to be useful over time are identified and removed by associating each axiom with a *strength* that decays exponentially if it is not used or learned again.

In addition to constraints, the robot learns previously unknown causal laws and executability conditions if there is a mismatch between the observed state after action execution and the expected state based on reasoning with existing knowledge. Any expected but unobserved fluent literal indicates missing executability condition(s); any observed unexpected fluent literal suggests missing causal law(s). Examples of learned axioms include:

$$\begin{aligned} \neg stable(A) &\leftarrow obj\_relation(above, A, B), \\ &\quad obj\_surface(B, irregular) \\ \neg pickup(robot_1, Ob_1) &\leftarrow in\_hand(robot_1, Ob_2) \end{aligned}$$

which correspond to a state constraint (“an object placed on another with an irregular surface is unstable”) and an executability condition (“the robot cannot pick up an object if it is already holding another object”) respectively. For more details, see (Sridharan and Mota 2023).

Although we do not describe it in detail in this chapter, our architecture includes a similar learning scheme for planning and diagnostics tasks. This scheme learns actions and axioms from human descriptions of desired behavior, or observations obtained through active exploration or reactive action execution in response to unexpected outcomes. Reasoning automatically limits this learning to states, actions, and observations relevant to task(s) and goal(s) at hand; see (Sridharan and Meadows 2018) for details.

### 3.3 Explanation Generation

We consider an “explanation” to be a relational description of the robot’s decisions or beliefs. This component is based on a *theory of explanations* that maps the postulates of explainable agency to: (i) claims about representing, reasoning with, and learning knowledge to support explanations; (ii) a characterization of explanations along axes based on abstraction, specificity, and verbosity; and (iii) a methodology for constructing explanations (Sridharan and Meadows 2019). This component is described below.

**Interaction interface.** Human interaction with our architecture is through speech or text. Existing software and a controlled vocabulary are used to parse human verbal input and to convert text to verbal response. Specifically, human verbal input is transcribed into text from the controlled vocabulary. This (or input) text is labeled using a part-of-speech (POS) tagger, and normalized with the lemma list (Someya 1998) and related synonyms and antonyms from WordNet (Miller 1995). The processed text helps identify the type of request: a desired goal or a question about decisions, beliefs, or hypothetical situations. Any goal is sent to the ASP program for planning; the robot executes the plan, performing diagnostics and replanning as needed, until the goal is achieved. For any question, the “Program Analyzer” considers the domain knowledge, inferred beliefs, and processed human input to automatically identify relevant axioms and literals. These literals are inserted into generic response templates based on the controlled vocabulary, resulting in (textual) descriptions that make contact with human concepts such as beliefs and goals, which are converted to synthetic speech if needed. Whenever the posed query or request is ambiguous, the robot constructs and poses queries to remove the ambiguity. Some examples of such interactions are provided in Section 4.2.

**Mental Simulations.** In order to explain the evolution of a particular belief or the (non)selection of a particular action at a particular time step, our architecture includes the ability to infer the associated sequence of beliefs and axioms. This capability is used by the “*Program Analyzer*” (below) to construct explanations. We adapt *proof trees*, which have been used to explain observations in the context of classical first-order logic statements (Ferrand, Lessaint, and Tessier 2006), to our formulation based on non-monotonic logic, to obtain the following methodology:

1. Select axioms with the target belief or action in the head.
2. Ground literals in the body of each selected axiom. Check if they are supported by the current answer set.
3. Create a new branch in a proof tree (with the target belief or action as the root) for each selected axiom supported by the answer set, and store the axiom and the related supporting ground literals in suitable nodes.
4. Repeats Steps 1-3 with the supporting ground literals in Step 3 as target beliefs in Step 1, until all branches reach a leaf node without any further supporting axioms.

Paths from the root to the leaves in these trees provide candidate explanations. If multiple paths exist, one of the shortest branches is selected and used to construct answers—see (Mota, Sridharan, and Leonardis 2021) for examples.

**Program Analyzer.** Algorithm 1 describes the approach for automatically identifying and reasoning with the relevant information to construct relational descriptions in response to questions or requests. It does so in the context of four types of *explanatory* requests or questions. The first three types were introduced in prior work as questions to be considered by any explainable planning system (Fox, Long, and Magazzeni 2017), and the fourth type considers the evolution of beliefs:

---

**Algorithm 1: (Program Analyzer) Answer query**

---

```

Input : Literal of input question;  $\Pi(\mathcal{D}, \mathcal{H})$ ; answer templates.
Output: Answer and answer Literals.
// Compute answer set
1 AS = AnswerSet( $\Pi$ )
2 if question = plan description then
   // Retrieve actions from answer set
3   answer_literals = Retrieve(AS, actions)
4 else if question = "why action X at step I?" then
   // Extract actions after step I
5   next_actions = Retrieve(AS, actions for step > I)
   // Extract axioms influencing these
   actions
6   relevant_axioms = Retrieve( $\Pi$ , head =  $\neg$  next_actions)
   // Extract relevant literals from
   Answer Set
7   relevant_literals = Retrieve(AS, Body(relevant_axioms)
    $\in I \wedge \notin I + 1$ )
   // Output literals
8   answer_literals = pair(relevant_literals, next_actions)
9 else if question = "why not action X at step I?" then
   // Extract axioms relevant to action
10  relevant_axioms = Retrieve( $\Pi$ , head =  $\neg$  occurs(X))
   // Extract relevant literals from
   Answer Set
11  answer_literals = Retrieve(AS, Body(relevant_axioms)
    $\in I \wedge \notin I + 1$ )
12 else if question = "why belief Y at step I?" then
   // Extract axioms influencing this
   belief
13  relevant_axioms = Retrieve( $\Pi$ , head = Y)
   // Extract body of axioms
14  answer_literals = Recursive.Examine(AS,
   Body(relevant_axioms))
15 Construct_Answer(answer_literals, answer_templates)

```

---

1. **Plan description:** When asked to describe a particular plan, the robot parses the related answer set(s) to extract a sequence of actions of the form *occurs(action1, step1), ..., occurs(actionN, stepN)* (line 3, Algorithm 1). These actions are used to construct the response.

2. **Action justification: Why action X at step I?** To justify the execution of any particular action at step I:

- (a) For each action  $A$  that occurred after time step  $I$ , the robot examines relevant executability condition(s) and identifies literal(s) that would prevent  $A$ 's execution (lines 5-7). For the goal of placing the orange block on the table in Figure 1 (right), assume that the actions executed include *occurs(pickup(robot, blue\_block), 0)*, *occurs(putdown(robot, blue\_block), 1)*, and *occurs(pickup(robot, orange\_block), 2)*. If the focus is on the first *pickup* action, an executability condition related to the second *pickup* action:

$$\neg \text{occurs}(\text{pickup}(\text{robot}, A), I) \leftarrow \text{holds}(\text{obj\_relation}(\text{below}, A, B), I)$$

is ground in the scene to obtain *obj\_relation(below, orange\_block, blue\_block)* as a literal of interest.

- (b) If any identified literal is in the answer set at the time step of interest (0 in current example), and is absent or negated in the next step, it is a reason for executing the action ( $X$ ) being considered (line 7).
- (c) The condition modified by the execution of the action of interest ( $X$ ) is paired with the subsequent action ( $A$ ) to construct the answer (line 8). For instance, the question “Why did you pick up the blue block at time step 0?”, receives the answer “I had to pick up the orange block, and it was located below the blue block”.

A similar approach is used to justify the selection of any particular 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 for execution:

- (a) The robot identifies executability conditions with action  $X$  in the head, i.e., conditions that (if true) would prevent  $X$  from being included in plans (line 10).
- (b) For each identified executability condition, the robot examines whether literals in the body are satisfied in the corresponding answer set (line 11). If so, these literals are used to construct the answer.

Suppose action  $putdown(robot, blue\_block, table)$  occurred at step 1 in Figure 1 (right). For the question “Why did you not put the blue block on the tennis ball at step 1?”, the following executability condition is identified:

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

which implies that an object cannot be placed on another object with an irregular surface. The answer set indicates that the tennis ball has an irregular surface. The robot answers “Because the tennis ball has an irregular surface”. This process uses the *belief tracing* approach above.

### 4. Belief query: Why belief Y at step I? To explain any particular belief, the robot uses the *belief tracing* approach to identify the supporting axioms and relevant literals to construct the answer. For example, to explain the belief that object $ob_1$ is unstable in step $I$ , the robot finds the support 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. Searching for why  $ob_1$  is believed to have a small base identifies the axiom:

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

Asking “why do you believe object  $ob_1$  is unstable at step  $I$ ?” would provide the answer “Because object  $ob_2$  is below object  $ob_1$ ,  $ob_2$  is small, and  $ob_1$  is big”.

**Disambiguation.** Questions or requests posed by humans may be ambiguous in terms of the objects or the time step that they reference. Our architecture includes a method to

automatically construct questions to address such ambiguities. Inspired by findings in psychology and cognitive science (Friedman 1974; Read and Marcus-Newhall 1993), this method enables the robot to construct queries comprising the set of object attributes most likely to address the ambiguity. The method is based on three heuristic rules: (i) select attributes that match with a minimum number of ambiguous objects for the query and scene under consideration; (ii) since queries with many attributes are more likely to confuse a human, select questions with the minimum number of attributes; (iii) assign higher priority to attributes that are preferred by humans (if known) and are easy for the robot to detect. We only summarize this capability here to provide a complete description of the architecture. For more details about the heuristic rules and their use to construct disambiguation queries, see (Mota and Sridharan 2021).

## 4 Experimental Results

Section 4.1 describes the experimental setup, followed by execution traces in Section 4.2 and quantitative results in Section 4.3. We evaluated the ability to learn axioms and construct relational descriptions of decisions and beliefs in response to different types of questions.

### 4.1 Experimental Setup

We experimentally evaluated the following hypotheses:

**H1** : our architecture supports reliable learning of unknown axioms, improving the quality of plans generated; and

**H2** : leveraging the links between reasoning and learning improves the accuracy of the explanatory descriptions.

Experimental trials considered images from the robot’s camera and simulated images. Real world images contained 5–7 objects of different colors, textures, shapes, and sizes in different locations of the AR domain. The objects included cubes (blocks), a pig, a bell pepper, a tennis ball, cups, an apple, an orange, and a pot. These objects were either stacked on each other or spread on a table in different locations—see Figure 1 (left). A total of 40 configurations were created, each with five different goals for planning and four different questions for each plan (one for each question type), resulting in a total of 200 plans and 800 questions. We used a Baxter robot to manipulate objects on a tabletop.

Since it is difficult to explore a wide range of objects and scenes with physical robots, we also used a real-time physics engine (Bullet) to create 40 simulated images, each with 7–9 objects (3–5 stacked and the remaining on a flat surface). Objects included cylinders, spheres, cubes, a duck, and five household objects from the Yale-CMU-Berkeley dataset (apple, pitcher, mustard bottle, mug, and box of crackers). We once again considered five different goals for planning and four different questions (one for each type) for each plan, resulting in the same number of plans (200) and questions (800) as with the real world data.

To explore the interplay between reasoning and learning, we focused on the effect of learned knowledge on planning and constructing explanations. Specifically, we prepared a knowledge base in which some axioms governing

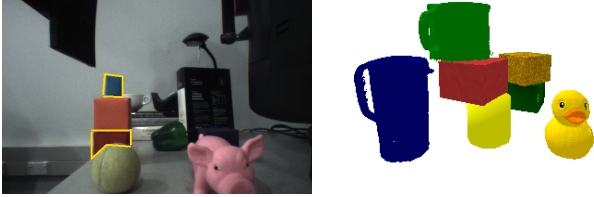


Figure 4: (Left) relation between blue cube and red cube is important for the explanation in Execution Example 2; (Right) simulated image used in Execution Example 3.

the domain dynamics were missing. We then ran experiments in which our architecture learned the missing axioms over time, as described in Section 3.2, and used them for planning and explanation generation. The baseline for comparison in these experiments included the reasoning and explanation generation components of our architecture but did not support any learning, i.e., it only used the initial knowledge base with some axioms missing. During planning, the performance measures included the number of optimal, sub-optimal, and incorrect plans, and the planning time. An "optimal" plan is a minimal plan, i.e., the quality of a plan was measured in terms of the ability to compute plans with require the least number of actions to achieve the goal. The quality of an explanation was measured in terms of precision and recall of the literals in the answer provided by our architecture in comparison with the expected ("ground truth") response provided manually (by the designer). Any claims of statistical significance were based on a paired t-test.

Note that the experimental setup described above does not include any studies with human subjects evaluating the quality of the explanations provided by our architecture. Such studies provide important feedback that can be used to evaluate and improve the architecture, but we leave such studies for future work. Instead, we present some execution traces describing the operation of our architecture, followed by a discussion of quantitative experimental results.

## 4.2 Execution Traces

The following execution traces demonstrate the capabilities of our architecture.

### Execution Example 1. [Planning and learning]

The robot in the AR domain is in the *study*; it is asked to bring a cup to the *study*, i.e., the goal state contains:  $loc(C, study)$ ,  $not\ in\_hand(robot_1, C)$ , where  $C$  is a *cup*.

- The computed plan of abstract actions is:

$$\begin{aligned} &move(robot_1, kitchen), pickup(robot_1, C), \\ &move(robot_1, study), putdown(robot_1, C) \end{aligned}$$

This plan uses the default knowledge that cups are usually in the *kitchen* that is next to the *study*.

- To implement each abstract transition  $T$ , the robot zooms to  $\mathcal{D}_f(T)$ . For instance, only cells in the *study* and the *kitchen* are relevant to the first *move* action; irrelevant objects and locations are automatically ignored.
- The zoomed description is used to obtain a probabilistic policy that is invoked repeatedly to execute a sequence

of concrete actions that implements the abstract action, e.g., robot is in a cell in the *kitchen* after first *move*. Other actions are executed in a similar manner.

- The robot's attempt to pick up a cup in the kitchen failed. The robot observes that the cup is *heavy* while its arm is *light*. It then learns the following executability condition (written as a statement in  $\mathcal{AL}_d$ ):

$$\text{impossible } pickup(robot_1, C) \text{ if } arm(robot_1, light), \\ obj\_weight(C, heavy)$$

It is also possible to learn and merge other kinds of axioms with the existing knowledge.

- The robot is also able to automatically provide on-demand explanations at a suitable level of abstraction.

**Human:** "Please describe the executed plan in detail."

**Robot:** "I moved to cell  $c_2$  in the *kitchen*. I picked the large cup by its handle from the counter [...] I moved to cell  $c_4$  of the *study*. I put the cup down on the red table."

Next, consider the robot's ability to provide explanations to different kinds of queries.

### Execution Example 2. [Explain plans and beliefs]

In the scene in Figure 4 (left), the robot's goal is to achieve a state in which the red cube is on top of the orange cube. The following interaction takes place *after* the robot has executed a plan to achieve the goal.

- **Human:** "Please describe the plan."

**Baxter:** "I picked up the blue cube. I put the blue cube on the table. I picked up the orange cube. I put the orange cube on the table. I picked up the red cube. I put the red cube on the orange cube."

- The robot may have to justify a particular action.

**Human:** "Why did you pick up the blue cube at step 0?"

**Baxter:** "Because I had to pick up the red cube, and it was below the blue cube."

This answer is also used to automatically highlight the relevant image regions that influenced this answer, providing additional transparency—see Figure 4 (left).

- The robot may have to justify the choice of actions.

**Human:** "Why did you not put down the orange cube on the blue cube?"

**Baxter:** "Because the blue cube is small." In the absence of any reference to a particular time step, the robot considers the single instance (in the executed plan) of putting the orange cube on another cube or surface. The answer is based on learned default knowledge that any structure with a large object on a small object is typically unstable.

- The human may ask the robot to justify beliefs.

**Human:** "Why did you believe that the red cube was below the blue cube in the initial state?"

**Baxter:** "Because I observed the red cube below the blue cube in step 0."

- The robot can run *mental simulations* to answer counterfactual (i.e., hypothetical) questions.

**Human:** "What would happen if the ball is rolled?"

**Baxter:** "The structure of blocks would be unstable".

### Execution Example 3. [Disambiguation]

Consider the simulated scenario in Figure 4 (right).

- **Human:** “Move the yellow object on to the green cube.”  
There is ambiguity in the reference to a yellow object. Since the yellow cube is already on the green cube, and the yellow cylinder is below other objects, the robot poses the following clarification 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.”

- The robot computes the plan: *pick up the green mug; put the green mug on the table; pick up the red cube; put the red cube on the table; pick up the yellow cube; put the yellow cube on the table; pick up the yellow cylinder; put the yellow cylinder on the green cube.*

Note that there are other equally valid plans, e.g., one that moves the yellow cube to the table first.

**Human:** “Why do you want to pick up the green mug?”

**Robot:** “I have to place the yellow cylinder on the green cube, and the yellow cylinder is below the green mug.”

The robot can also trace the evolution of particular beliefs and the application of relevant axioms to answer questions after plan execution.

- **Human:** “Why did you not pick up the red cube at step1?”

**Robot:** “Because the red cube is below the green mug.”

**Human:** “Why did you move the yellow cube onto the table?”

**Robot:** “I had to put the yellow cylinder on top of the green cube. The green cube was below the yellow cube.”

## 4.3 Empirical Results

To evaluate **H1**, we removed five axioms (two causal laws and three executability conditions) from the robot’s knowledge, and ran the learning algorithm 20 times. We measured the precision and recall of learning these axioms in each run; Table 1 summarizes the results. Each run was terminated if the robot executed a number of actions without detecting any inconsistency, or if the number of decision trees constructed exceeded a number. The row labeled “Strict” summarizes results when any variation in the target axiom, i.e., axioms with additional irrelevant literals, was considered to be incorrect. An example of such an axiom in which the second literal in the body is irrelevant is shown below.

$$\begin{aligned} \neg \text{holds}(\text{in\_hand}(R1, O1), I + 1) \leftarrow \\ \text{occurs}(\text{putdown}(R1, O1, O2), I), \\ \neg \text{holds}(\text{in\_hand}(R1, O5), I). \end{aligned} \quad (1)$$

The row labeled “Relaxed” summarizes results when over-specifications were not counted as errors. High precision and recall support hypothesis **H1**.

The next set of experiments further evaluated **H1**.

1. For the 40 initial object configurations (Section 4.1), information extracted from the images corresponding to top and front views (i.e., from the camera on each gripper) was encoded as initial state in the ASP program.

Table 1: Precision and recall for learning previously unknown axioms. Errors under ”Strict” mainly correspond to the inclusion of additional irrelevant literals.

Missing Axioms	Precision	Recall
Strict	69%	78%
Relaxed	96%	95%

Table 2: Number of plans and planning time after including the learned axioms for reasoning (our architecture), expressed as a fraction of the values without including the learned axioms (baseline).

Measures	Ratio (with/without)	
	Real scenes	Simulated scenes
Number of steps	1.15	1.23
Number of plans	0.81	1.08
Planning time	0.96	1.02

2. For each initial state, five goals were randomly encoded (one at a time) in the ASP program. The robot reasoned with the existing knowledge to create plans for these 200 combinations (40 initial states, five goals).
3. Plans were evaluated based on the number of optimal, sub-optimal, and incorrect plans, and planning time. Trials were repeated with and without the learned axioms.

Recall that our architecture reasons with a knowledge base that includes the learned axioms whereas the knowledge base used by the baseline does not include these axioms. We conducted paired trials with and without the learned axioms in the ASP program used for reasoning. The initial conditions and goal were identical in each paired trial, but differed between paired trials. We expressed the number of plans and the planning time with the learned axioms as a fraction of the corresponding values obtained by reasoning without the learned axioms. The average of these fractions over all the trials is reported in Table 2. We also computed the number of optimal, sub-optimal, and incorrect plans in each trial as a fraction of the total number of plans; we did this with and without the learned axioms and the average over all trials is summarized in Table 3.

These results indicate that for images of real scenes, reasoning with the learned axioms reduced the search space, resulting in a smaller number of plans and a reduced planning time. The use of the learned axioms did not make any significant difference with the simulated scenes. This is understandable because the simulated images had more objects (than real scenes) with several of them being small objects. This increased the number of plans to achieve any given goal. Also, when the robot used the learned axioms for reasoning, it reduced the number of sub-optimal plans and eliminated all incorrect plans; almost every sub-optimal plan corresponded to a goal that could not be achieved without creating an exception to a default. Without the learned axioms, a larger fraction of the plans were sub-optimal or incorrect, particularly for simulated scenes with multiple objects. These results further support **H1**.

The next set of experiments evaluated **H2**:

1. For each of the 200 combinations from the first set of experiments with real-world data, we considered knowl-



Table 3: Number of optimal, sub-optimal, and incorrect plans expressed as a fraction of the total number of plans. Reasoning with the learned axioms (our architecture) improves performance compared with the baseline that reasons without the learned axioms.

Plans	Real Scenes		Simulated Scenes	
	Without	With	Without	With
Optimal	0.4	0.9	0.14	0.3
Sub-optimal	0.11	0.1	0.46	0.7
Incorrect	0.49	0	0.4	0

Table 4: (**Real scenes**) Precision and recall of retrieving relevant literals for constructing answers to questions with and without using the learned axioms for reasoning. Using the learned axioms significantly improves the ability to provide accurate explanations in all but one type of query.

Query Type	Precision		Recall	
	Without	With	Without	With
Plan description	78.5%	100%	67.5%	100%
Why X?	76.3%	95.3%	66.8%	95.3%
Why not X?	96.6%	96.6%	64%	100%
Why belief Y?	96.7%	99%	95.6%	100%

edge bases with and without the learned axioms and asked the robot to compute plans to achieve the goals.

2. The robot had to describe the plan and justify the choice of a particular action (chosen randomly) in the plan. Then, one parameter of the chosen action was changed randomly to ask why this new action could not be applied. Finally, a belief related to the previous two questions had to be justified—see Execution Example 2.
3. The literals present in the answers were compared against the literals in the "ground truth" response, with the average precision and recall scores shown in Table 4.
4. We also performed these experiments with simulated images, with the results summarized in Table 5.

Tables 4, 5 show that in all but one type of question (i.e., counterfactual) posed about real world scenes, the precision and recall of relevant literals (for constructing explanations) were higher when the learned axioms were used for reasoning compared with the baseline that did not use these learned axioms. The improvement in performance was particularly pronounced when the robot had to answer certain types of questions about certain types of scenes (e.g., justification of action choices). For certain types of questions (e.g., about specific beliefs), the precision and recall rates were reason-

Table 5: (**Simulated scenes**) Precision and recall of retrieving relevant literals for constructing answers to questions with and without reasoning with learned axioms. Using the learned axioms significantly improves the ability to provide accurate explanations for all four types of queries.

Query Type	Precision		Recall	
	Without	With	Without	With
Plan description	70.8%	100%	58%	100%
Why X?	65.6%	93.0%	57%	93.0%
Why not X?	90.5%	96.4%	65.2%	100%
Why belief Y?	92.7%	98.4%	90.3%	99.2%

able even when the learned axioms were not included. This is because not all the learned axioms were needed to answer each question. When the learned axioms were used for reasoning, errors were rare and corresponded to additional literals being included in the explanation (i.e., over-specified explanations). Enabling reasoning and learning to inform each other thus resulted in more accurate relational descriptions of decisions and beliefs in response to different types of questions. These results support **H2**.

## 5 Conclusions

The architecture described in this paper is a step towards greater transparency in reasoning and learning for integrated robot systems. The architecture encodes the principle of stepwise refinement to leverage the complementary strengths of non-monotonic logical reasoning with common-sense domain knowledge, data-driven learning from a limited set of examples, and the inductive learning of previously unknown axioms governing domain dynamics. Once the designer provides the domain-specific information, planning, diagnostics, and execution are automated. In addition, the interplay between representation, reasoning, and learning is used to embed the principles of explainable agency, enabling a robot to reliably and efficiently construct and provide on-demand relational descriptions of its decisions and beliefs in response to different types of questions. Experimental results in described in this chapter, and those described in other related papers (Sridharan and Meadows 2019; Mota, Sridharan, and Leonardis 2021), demonstrate the smooth transfer of control and relevant knowledge between components of the architecture, confidence in the correctness of the robot's behavior, and the applicability of the underlying methodology to different domains.

Our architecture opens up multiple directions of future work. For example, we will further explore how the interplay between representation, reasoning, and learning can be leveraged to support explainable agency in one or more robots assisting humans in dynamic domains. In addition, we will conduct experimental studies with human participants evaluating the quality of our explanations, and use the feedback from these participants to make revisions of our architecture and algorithms. Furthermore, we will investigate whether our architecture can be extended to consider social norms while generating explanations of the decisions and beliefs of a robot assisting humans in complex domains.

## Acknowledgements

This work is the result of research threads pursued in collaboration with Tiago Mota, Heather Riley, Ben Meadows, Rocio Gomez, Michael Gelfond, Jeremy Wyatt, and Shiqi Zhang. This work was supported in part by the U.S. Office of Naval Research Science of Autonomy Awards N00014-13-1-0766, N00014-17-1-2434 and N00014-20-1-2390, the Asian Office of Aerospace Research and Development award FA2386-16-1-4071, and the U.K. Engineering and Physical Sciences Research Council award EP/S032487/1. All conclusions are those of the author.

## References

- Anjomshoae, S.; Najjar, A.; Calvaresi, D.; and Framling, K. 2019. Explainable agents and robots: Results from a systematic literature review. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. Montreal, Canada.
- Assaf, R.; and Schumann, A. 2019. Explainable Deep Neural Networks for Multivariate Time Series Predictions. In *International Joint Conference on Artificial Intelligence*, 6488–6490. Macao, China.
- Balai, E.; Gelfond, M.; and Zhang, Y. 2013. Towards Answer Set Programming with Sorts. In *International Conference on Logic Programming and Nonmonotonic Reasoning*. Corunna, Spain.
- Borgo, R.; Cashmore, M.; and Magazzeni, D. 2018. Towards Providing Explanations for AI Planner Decisions. In *IJCAI Workshop on Explainable Artificial Intelligence*, 11–17.
- Chakraborti, T.; Sreedharan, S.; and Kambhampati, S. 2018. Explicability versus Explanations in Human-aware Planning. In *International Conference on Autonomous Agents and Multiagent Systems*. Stockholm, Sweden.
- de Kleer, J.; and Williams, B. C. 1987. Diagnosing Multiple Faults. *Artificial Intelligence*, 32: 97–130.
- Ferrand, G.; Lessaint, W.; and Tessier, A. 2006. Explanations and Proof Trees. *Computing and Informatics*, 25: 1001–1021.
- Fox, M.; Long, D.; and Magazzeni, D. 2017. Explainable Planning. In *IJCAI Workshop on Explainable AI*.
- Friedman, M. 1974. Explanation and Scientific Understanding. *Philosophy*, 71(1): 5–19.
- Gebser, M.; Kaminski, R.; Kaufmann, B.; and Schaub, T. 2012. *Answer Set Solving in Practice, Synthesis Lectures on Artificial Intelligence and Machine Learning*. Morgan Claypool Publishers.
- Gelfond, M.; and Incelezan, D. 2013. Some Properties of System Descriptions of  $AL_d$ . *Journal of Applied Non-Classical Logics, Special Issue on Equilibrium Logic and Answer Set Programming*, 23(1-2): 105–120.
- Gomez, R.; Sridharan, M.; and Riley, H. 2021. What do you really want to do? Towards a Theory of Intentions for Human-Robot Collaboration. *Annals of Mathematics and Artificial Intelligence, special issue on commonsense reasoning*, 89: 179–208.
- Koh, P. W.; and Liang, P. 2017. Understanding Black-box Predictions via Influence Functions. In *International Conference on Machine Learning (ICML)*, 1885–1894. Sydney, Australia.
- Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Neural Information Processing Systems*, 1097–1105.
- Laird, J. E. 2012. *The Soar Cognitive Architecture*. The MIT Press.
- Langley, P.; Meadows, B.; Sridharan, M.; and Choi, D. 2017. Explainable Agency for Intelligent Autonomous Systems. In *Innovative Applications of Artificial Intelligence*. San Francisco, USA.
- LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86: 2278–2324.
- Miller, G. A. 1995. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11): 39–41.
- Miller, T. 2019. Explanations in Artificial Intelligence: Insights from the Social Sciences. *Artificial Intelligence*, 267: 1–38.
- Mota, T.; and Sridharan, M. 2018. Incrementally Grounding Expressions for Spatial Relations between Objects. In *International Joint Conference on Artificial Intelligence*. Stockholm, Sweden.
- Mota, T.; and Sridharan, M. 2021. Answer me this: Constructing Disambiguation Queries for Explanation Generation in Robotics. In *IEEE International Conference on Development and Learning*.
- Mota, T.; Sridharan, M.; and Leonardis, A. 2021. Integrated Commonsense Reasoning and Deep Learning for Transparent Decision Making in Robotics. *Springer Nature CS*, 2(242): 1–18.
- Norcliffe-Brown, W.; Vafeais, E.; and Parisot, S. 2018. Learning Conditioned Graph Structures for Interpretable Visual Question Answering. In *Neural Information Processing Systems*. Montreal, Canada.
- Read, S. J.; and Marcus-Newhall, A. 1993. Explanatory coherence in social explanations: A parallel distributed processing account. *Personality and Social Psychology*, 65(3): 429.
- Ribeiro, M.; Singh, S.; and Guestrin, C. 2016. Why Should I Trust You? Explaining the Predictions of Any Classifier. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–1144.
- Seegerbarth, B.; Müller, F.; Schattenberg, B.; and Biundo, S. 2012. Making Hybrid Plans More Clear to Human Users: A Formal Approach for Generating Sound Explanations. In *International Conference on Automated Planning and Scheduling*, 225–233.
- Someya, Y. 1998. Lemma List for English Language.
- Sreedharan, S.; Kulkarni, A.; and Kambhampati, S. 2022a. *Explainable Human-AI Interaction: A Planning Perspective*. Springer International Publishing.
- Sreedharan, S.; Kulkarni, A.; and Kambhampati, S. 2022b. *Explanation as Model Reconciliation*, 59–80. Springer International Publishing.
- Sridharan, M.; Gelfond, M.; Zhang, S.; and Wyatt, J. 2019. REBA: A Refinement-Based Architecture for Knowledge Representation and Reasoning in Robotics. *Journal of Artificial Intelligence Research*, 65: 87–180.
- Sridharan, M.; and Meadows, B. 2018. Knowledge Representation and Interactive Learning of Domain Knowledge for Human-Robot Collaboration. *Advances in Cognitive Systems*, 7: 77–96.

Sridharan, M.; and Meadows, B. 2019. Towards a Theory of Explanations for Human-Robot Collaboration. *Kunstliche Intelligenz*, 33(4): 331–342.

Sridharan, M.; and Mota, T. 2023. Towards Combining Commonsense Reasoning and Knowledge Acquisition to Guide Deep Learning. *Autonomous Agents and Multi-Agent Systems*, 37(4).

Winston, P. H.; and Holmes, D. 2018. The Genesis Enterprise: Taking Artificial Intelligence to Another Level via a Computational Account of Human Story Understanding. Computational models of human intelligence report 1, Massachusetts Institute of Technology.

Yi, K.; Wu, J.; Gan, C.; Torralba, A.; Kohli, P.; and Tenenbaum, J. B. 2018. Neural-Symbolic VQA: Disentangling Reasoning from Vision and Language Understanding. In *Neural Information Processing Systems*. Montreal, Canada.