Reasoning Under Possibility in Open-World Robotics: Handling Uncertainty, Actions, and Hypothetical States

Nikoletta Dimitrova Nikolova

MSc Robotics, Faculty Mechanical Engineering

Delft, The Netherlands

University Supervisor: Dr. Joris Sijs

29-07-2025

Reasoning Under Possibility in Open-World Robotics: Handling Uncertainty, Actions, and Hypothetical States

Nikoletta Dimitrova Nikolova
Master Thesis
MSc Robotics, Faculty Mechanical Engineering
Delft, The Netherlands
29-07-2025

***Abstract*—Robotic systems are more commonly deployed in open-world environments, where safe and reliable operation is challenged by uncertainty, dynamism, and incomplete knowledge. To explore new approaches for solving those challenges, this work focuses on possibility theory. It explores the use of possibility theory as a framework for reasoning under such conditions. A reasoning solution is proposed, that processes scene graph observations, compares them to existing knowledge, and generates hypotheses using possibility distributions as the foundation for modeling and representing uncertainty. This enables flexible belief updates and action-based inference. The system is evaluated through a set of test scenarios that qualitatively assess its reasoning performance. While still an early-stage implementation, the work highlights the potential of possibility-based reasoning in robotics and lays a foundation for future research in this direction.**

***Index Terms*—possibility theory, reasoning, sequence, robots, open-world**

I. Introduction

Robots are becoming increasingly integrated into everyday life. They expand beyond traditional industrial applications and enter dynamic, unstructured, and human-populated environments. From healthcare to disaster response to domestic assistance to collaborative manufacturing, robots are expected to operate in different situations without posing danger to the outside world. In the past robots were predominantly developed and deployed for closed-world situations. In those explicitly modeled environments robots operate with comprehensive prior knowledge, rely on well-defined tasks and operate within a limited pre-defined scope. This is common for example in industrial automation, where robots have to perform repetitive tasks in structured and controlled spaces.

However, with the new applications emerging, robots expand into open-world environments. Here it is possible to encounter unforeseen events, unfamiliar objects, incomplete information, changing context. Information may be missing, ambiguous, or even contradictory. In such settings, robots must not just cope with uncertainty about outcomes, but about the very structure of the world they are in.

In open worlds robots cannot rely on exhaustive prior models or predefined actions. They need to instead be capable of interpreting observations actively, revise internal beliefs, reason under uncertainty, often with only partial and evolving knowledge. Environments can be unstructured and require interaction or collaboration with people. All of this demands for robots to be able to perceive, interpret and reason about their world.

Robots must make decisions without complete and/or reliable information. Prior assumptions may be contradicted, observations may be noisy or incomplete, and the consequences of actions may depend on hidden or shifting variables.

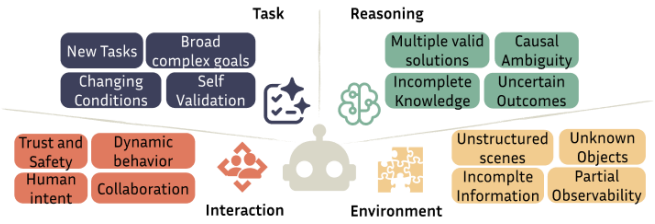


Fig. 1: Types of challenges that robots can face in open-world scenarios, split in categories

As Figure 1 illustrates, the range of challenges that robots face in open-world situations can be broad. Robots must navigate unfamiliar environments, infer meaning from limited data, interact with humans or other agents, and pursue abstract, goal-driven tasks (e.g., *"Find the injured person"*) that involve both perception and reasoning. These scenarios demand decisions based on partial evidence, the handling of conflicting signals, and the capacity to act safely even when the world is only partially understood.

A robot needs to be able to reason with what it knows, revise what it thinks it knows, and explore what it does not know, all while maintaining safe operations and advancing on its given goal. This entails integrating new observations that may introduce unfamiliar entities or contradict prior beliefs, while maintaining a coherent internal representation of the world. The description of the world (often called world model), needs to be structured and capable of evolving over time. In this context, knowledge graphs can provide a principled way to

represent symbolic information about the world in a graph format.

A robot must also hypothesize about unobserved causes or transitions, inferring the most plausible sequence of events that could explain a set of unordered or incomplete observations. Such reasoning enables both context understanding and decision making, even when direct sensory input is sparse or unreliable.

To handle all those new challenges, it is valuable to look to other domains and the solutions that have been used to tackle decisions under uncertainty. This is not a new problem, and there are solutions that exist but are not yet explored in robotics. One such example, and the focus of this research, is the concept of possibility theory. It focuses on using possibility instead of probability, to represent ignorance and uncertainty, and gives a new perspective to how problems of this type can be tackled.

This work focuses on designing a reasoning solution, that uses possibility theory as its core tool to handle, represent and propagate uncertainty. The goal is to enable a robot to interpret observations over time, maintain and revise internal models, and reason about the possibility of knowledge. Given a sequence of four images, the task is to understand the sequence of events and determine what is the most possible combination possible. The solution is constructed using a knowledge graph (and ontology) for defining the world model, possibility theory as base uncertainty measure, and scene graphs as observation input. In doing this work, the aim is to provide a first look at how possibility can be integrated in the internal reasoning loop of a robot and to assess its potential added value.

This work is organized as follows - Section II focuses on providing the relevant background for both robotics and possibility theory. Next, Section III defines the exact problem that is being tackled in this work, and Sections III, IV, VI provides a deep dive into the construction of the developed solution. Lastly Sections VII andVIII provide an overview of results and experimentation process, followed by discussion and conclusion.

II. Background

A. Open World Robotics - the context

Robotics research in recent years has increasingly been dealing with the concept of reasoning. [1] define, that the aim of research is to create *"an autonomous robot that can actively explore the real environment, acquire knowledge, and learn skills continuously"*. This would mean creating a solution that is able to operate in dynamic, uncertain environments. While there are multiple different aspects of reasoning that can be considered, in the context of this work there are three main ones: context understanding, environment description, and decision-making.

In the area of context understanding, current research is looking at elements such as recognizing new (novel, previously unseen) objects [2] or semantic localization [3]. Others focus on correct and detailed modeling of the world, looking at discovering unknown domain dynamics [4] or using hypergraphs as a way to represent general knowledge [5].

Considering environment description, one could think about how to model knowledge and information about the world. [6] for example present a way to split information in three groups: instance, common and diagnostic. The aim is to distinguish between the different types of information that can be within an environment, and to allow a robot to represent not only how the world changes based on actions, but also what are its internal beliefs. Other approaches for world modeling are focused on using language to understand and describe a scene, such as temporal grounding graphs with language introduced by [7].

In the context of decision-making, robotic applications can strive towards autonomous intelligence [8]. This could include different types of reasoning capabilities. An example is the one defined by [9], who distinguish between three different processing capabilities of robots: innate (present at the beginning), self-developed (developed through interaction with the environment), and socially developed (developed by others). Robots need to be able to take decisions in uncertain environments, continuously adapt their understanding of the world, and handle the growing uncertainty and unpredictability in a safe manner. Solutions such as the one by [10] offer an approach, where the robot learns while performing a particular task to handle new situations, by using deep reinforcement learning algorithms and *imagining* situations. New solutions such as Large Language Models (LLMs) also provide promising capabilities. They can be used to equip robots with Vision-Language-Action Models [11] allowing better generalization of actions, or development of open knowledge-based frameworks [12]).

However none of the existing solutions appears to be a universal one and the topic of reasoning in open world is still open research with multitude of challenges. Models, especially LLMs, tend to hallucinate [13]. Others are developed to work only in closed-world environments (e.g. [6] who focus on indoor environments) or partial solutions (e.g. [3] looking only at localization). Given the complexity of this task (consider Figure 1), it is valuable to consider alternative approaches which are used in different tasks.

B. From Uncertainty to Possibility

Uncertainty is a fundamental aspect of reasoning in complex and dynamic environments. It arises from incomplete, imprecise, or conflicting information. Two main types are commonly distinguished: epistemic uncertainty, resulting from a lack of knowledge, and aleatoric uncertainty, arising from inherent randomness or noise in the system [14]. Aleatoric uncertainty is often related to sensor limitations or hardware variability and is typically irreducible [15]. In contrast, epistemic uncertainty can be reduced through knowledge acquisition and inference. For tasks involving exploration, reasoning, and knowledge propagation in open-world robotic systems, epistemic uncertainty is the primary concern.

To address uncertainty formally, a mathematical framework is required. It needs to be one that enables the representation, quantification, manipulation, and update of incomplete information [16]. Two major frameworks offer such formalisms: probability theory and possibility theory. Probability theory models uncertainty through likelihoods ($P(A)$), assuming well-defined priors and sufficient statistical data. In contrast, possibility theory focuses on plausibility ($\Pi(A)$) and certainty ($N(A)$), making it better suited to contexts involving partial, vague, or missing information.

Possibility theory was first introduced by Zadeh [17] and later formalized by Dubois and Prade [18]. It provides a lightweight, expressive alternative to probabilistic reasoning, particularly effective when prior probabilities are unavailable or incomplete. The distinction between possibility (how plausible an event is) and necessity (how certain it is) aligns well with systems that reason incrementally over time. [19] argue that the problem with probabilistic knowledge is the lack of distinction between lack of belief in proposition and belief in its negation, which is crucial for cases when there is uncertain or incomplete knowledge. This is where possibility theory has an advantage, as it is developed to represent incomplete information in either numeric or qualitative form. Moreover, it naturally supports non-monotonic reasoning, graded belief, and qualitative decision-making—key capabilities for flexible, context-aware reasoning in artificial agents [20]. For these reasons, possibility theory forms the foundation of the reasoning framework proposed in this work. A concise overview of the key concepts and definitions in possibility theory is provided in Appendix A.

Central to possibility theory is the concept of the possibility distribution π, which encodes an agent's knowledge state over a domain $\mathcal{X}$ by assigning a plausibility value $\pi(x) \in [0, 1]$ to each element $x \in \mathcal{X}$ [21]. A value of $\pi(x) = 1$ indicates that x is fully plausible, while $\pi(x) = 0$ signifies that it is ruled out. Figure 2 illustrates a simple example of how such a distribution might be constructed over a finite domain.

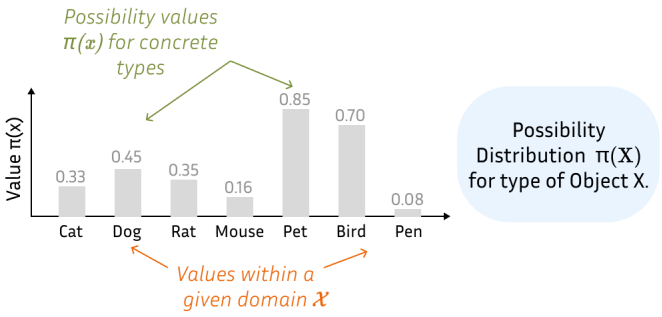


Fig. 2: Example of how a possibility distribution $\pi(X)$ can be constructed. Along the horizontal axis are the different values in a given domain $\mathcal{X}$, whereas the vertical axis represents the possibility values $\pi(x)$ for every type.

As discussed in [21], possibility distributions can be defined in either quantitative or qualitative terms. This work adopts the qualitative approach, where values are interpreted as relative rankings rather than precise numeric scores [22]. This choice reflects the nature of the available input. It can be limited, symbolic, and often uncertain, which can be typical in open-world tasks.

Recent work has begun to explore the integration of possibility theory into machine learning [23] and robotics, though applications remain sparse. Notable efforts include its use in sonar modeling for mobile robots [24] and in multi-agent coordination and task allocation [25], [26]. However, there is still a significant lack of practical implementations that use possibility theory as a core reasoning mechanism in robotics. This research aims to address that gap by developing a theoretical foundation and proof-of-concept implementation of a reasoning loop grounded in possibility theory.

III. Problem Definition

As input the system gets four images, represented as scene graphs, and has to understand their sequence. Robots observe the world sequentially, so to mimic that the developed system needs to also process the input sequentially. Therefore, the scene graphs are processed one by one by the reasoning process.

Every image from the given sequence is turned into an observation, which is given in the form of a scene graph. A scene graph represents a snapshot of the world at a given time t_n. It specifies what is currently happening by providing entities, attributes and relations that exist at that moment.

The system does not have information about which concrete sequence of scene graphs is the correct one: scene graph at t_1 may not be the real first image. This is why the solution processes all possible sequences and analyses at the end which order can be the correct one. Therefore the task given to the system can be formulated as answering the question *Do you know what happened here?*. The aim is to establish which sequence of scene graphs is the one with least uncertainty at the end. The expectation is that non-logical sequences (ones which a human would not choose) would yield lowed possibility values and higher uncertainty compared to logical ones. To be able to answer such questions, the solution needs at least partial understanding over what happens between images and what action is more possible than others.

The problem starts with a world model $\mathcal{W}$, containing predefined knowledge and instances, all captured within an ontology $\mathcal{K}$. $\mathcal{K}$ contains information about how the knowledge is structured, specifying entities $\mathcal{E}$, attributes $\mathcal{D}$, and relations $\mathcal{R}$.

$$\mathcal{K} = (\mathcal{E}, \mathcal{D}, \mathcal{R}) \tag{1}$$

At every time step t_n individuals $\mathcal{I}_{t_n} = \{\mathcal{E}_{t_n}, \mathcal{D}_{t_n}, \mathcal{R}_{t_n}\}$ are added and updated. As the consideration is for a particular time step t_n, then the world is also considered for a specific moment, denoted as $\mathcal{W}_{t_n}$.

$$\mathcal{W}_{t_n} = (\mathcal{K}, \mathcal{I}_{t_n}) \tag{2}$$

At each time step t_n, the system receives an input scene graph $\mathcal{G}_{t_n}$, which may introduce novel, uncertain, or even contradictory information with respect to the previous world model $\mathcal{W}t_{n-1}$.

$$\mathcal{G}_{t_n} = (\mathcal{E}'_{t_n}, \mathcal{D}'_{t_n}, \mathcal{R}'_{t_n}) \tag{3}$$

$\mathcal{G}_{t_{n+1}}$ may introduce novel information in the form of new entities, types, or relations previously unknown. It may contain facts that challenge prior knowledge (e.g. human is no longer holding a brush). Therefore, the world at time t_{n+1} is a function $\mathcal{F}$ of the current world knowledge $\mathcal{W}_{t_n}$ and $\mathcal{G}_{t_{n+1}}$:

$$\mathcal{W}_{t_{n+1}} = \mathcal{F}(\mathcal{W}_{t_n}, \mathcal{G}_{t_{n+1}}) \tag{4}$$

Given a set of unordered scene graphs $\mathcal{G}_1, \ldots, \mathcal{G}_n$—each representing a partial and potentially uncertain observation of the world at some unknown time step—there exist multiple possible temporal orderings of these observations. Let $\text{Perm}(\mathcal{G})$ denote the set of all permutations over the scene graphs. The task is to determine the most possible temporal sequence σ^*, such that the corresponding sequence of world states maximizes a global possible criterion under the reasoning model.

$$\sigma^* = [\mathcal{G}_{\sigma(1)}, \ldots, \mathcal{G}_{\sigma(n)}] \text{ such that,} \mathcal{W}_1 \to \mathcal{W}_2 \to \cdots \to \mathcal{W}_n \tag{5}$$

Each transition $\mathcal{W}_i \to \mathcal{W}_{i+1}$ is determined via the function $\mathcal{F}$. Every sequence has an overall possibility criterion (for the whole world) denoted as $\Pi(\sigma)$.

$$\sigma^* = \arg \max_{\sigma \in \text{Perm}(\mathcal{G})} \Pi(\sigma) \tag{6}$$

How exactly is this criterion determined, what is the function $\mathcal{F}$ and how is $\mathcal{K}$ constructed is explained in the following sections.

IV. World Model

At the core of the reasoning process lies the world model, implemented as an ontology $\mathcal{K}$, defining all entities, their attributes and relations over time. This representation forms the foundation for interpreting and evolving knowledge across time steps. The following section describes the key underlying concepts, while Appendix C visualizes the complete ontology.

A. Ontology Structure

The ontology is composed of physical and abstract entities $\mathcal{E}$. Physical entities include `actors` (e.g., humans, dogs) capable of executing actions, and `objects` (e.g., bowls, bags) that are acted upon. Abstract entities model auxiliary concepts such as `states` or `colors` necessary for reasoning.

Each entity may possess attributes, which are either qualitative (e.g., `color`, `pose`) or quantitative (e.g., `location-x`, `location-y`), and are defined as attribute-value pairs with associated confidence.

There are two types of relations present. There are spacial relations (e.g. `isLocated`) and actions (e.g. `holding`). The latter has a special construction.

B. Actions

Actions $\mathcal{A}$ are represented as relations connecting three roles, fulfilled by entities: `actor`, `object`, and `action-state`.

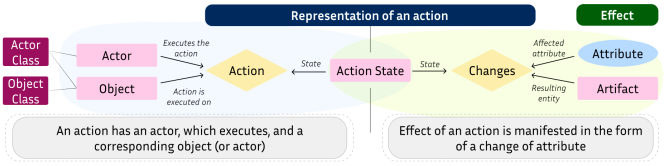


Fig. 3: Action representation in $\mathcal{K}$, showing its key components

An action $a \in \mathcal{A}$ is represented as two separate ontological components. It has a relation `action` (e.g. `painting`, or `holding`) and corresponding entity `action state`. The connection and construct is shown in Figure 3. An action relation connects an `actor`, which executes the action, an `object` which action is executed on, and a `state`. The `action state` is an entity, which represents the state of the given action. It also serves as a way to connect the concrete action to its expected effect on the world. This is done via a `changes` relation. This links action states to expected attribute changes.

C. Time

To enable reasoning across multiple time steps, it is necessary to have a temporal extension of the world representation, which allows to distinguish $\mathcal{W}_{t_n}$. This is achieved by annotating each entity with a `time-step` attribute. This indicates the discrete time step t_n to which a given knowledge belongs. This enables tracking the evolution of individuals across time (i.e., world states $\mathcal{W}_t$) and supports time-dependent reasoning.

D. Possibility

Uncertainty in knowledge is encoded structurally using dedicated attributes that store possibility distributions π. Each distribution is stored as a JSON string within the ontology and interpreted externally.

The key attribute is the `possibility distribution`, which encodes the system's belief about the type of a given entity. For example, `Object_X` may have a distribution over types such as `can` or `bottle`, with associated degrees of plausibility.

$$\pi(\texttt{Object\_X}) = [\texttt{can} : 0.7,\ \ \texttt{bottle} : 0.4,\ \ \texttt{bag} : 0.9]$$

Additional possibility-focused attributes are introduced for certain reasoning contexts, containing information about properties such as color and quantity:

- `color possibility distribution` captures uncertainty about an entity's color.
- `quantity possibility distribution` captures uncertainty in an entity's content quantity (e.g., food amount, paint amount)

These attributes are integral to the reasoning pipeline: they enable action validation, track hypothesis evolution, and model

uncertainty propagation over time. The actual formalization of the possibility distributions is defined in Section V, whereas their usage within the reasoning structure is described in Section VI.

V. Possibility Distributions

At the base of the proposed reasoning approach stands the concept of a *possibility distribution* π, a formalism from possibility theory that assigns a degree of possibility to alternatives, without requiring complete or probabilistic knowledge.

Formally, a possibility distribution $\pi_{\mathcal{X}} : \mathcal{D}_{\mathcal{X}} \rightarrow [0, 1]$ assigns to each value x in the domain $\mathcal{D}_{\mathcal{X}}$ a plausibility score $\pi(x)$. Within the created reasoning solution, $\pi_{\mathcal{X}}$ is used for different elements and for different operations, as shown in Figure 4, to represent graded uncertainty across multiple layers of the knowledge graph. The different equations, presented in the following sections, are inspired by the different methods for constructing a possibility distribution defined by [21].

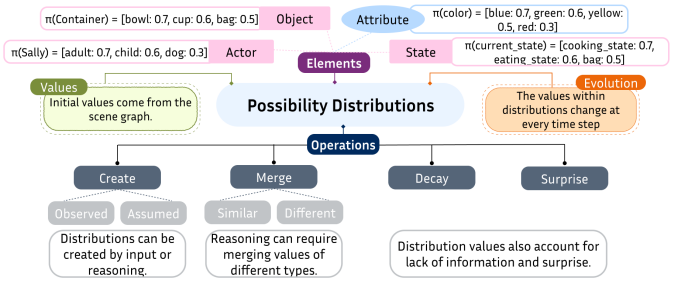


Fig. 4: The key types of possibility distributions present (top part of figure) and corresponding operations that can be performed (bottom part of figure)

A. Types

Possibility distributions are maintained across multiple conceptual layers of the ontology. Every entity may carry distributions over its type (e.g., `can`, `bucket`) and observable attributes (e.g., `pose`, `color`). These distributions persist across time steps and are updated iteratively. Action-related possibility structures, including expected effects, follow the same construct, but follow different rules (see Section VI-C for detailed explanation of this process and differences).

B. Key Operations

The following operations govern how possibility distributions are initialized, updated, and pruned during the reasoning process.

a) Creation from Observation: Observed entities and attributes are initialized using the confidence $\phi(x) \in [0, 1]$ presented in the scene graph $\mathcal{G}$:

$$\pi(x) = \phi, \forall x \in \mathcal{G}_t \tag{7}$$

b) Creation from Rules: For inferred states, such as actions, possibility is computed based on different satisfied conditions (called preconditions, see Section VI-C). Here the value(s) of $\pi(x)$ are calculated using the following formula:

$$\pi_{\text{action}} = \beta \cdot \min\left(c_1 \cdot \max_x \pi_{p_1}(x),\ \ldots,\ c_n \cdot \max_x \pi_{p_n}(x)\right) \tag{8}$$

In here $\beta \in [0, 1]$ represents the *intention factor* of an actor to execute a given action. The overall possibility of a precondition (e.g. $\exists$`HumanA` or $\exists$`Brush`) is denoted as p_i. As preconditions can be different types of statements, p_i is directly extracted from the distributions of the different statements. Lastly, c_i is the confidence or importance weight (typically $c_i = 1$) in the different preconditions.

c) Merging: To integrate multiple sources of information (e.g., prior belief π_{old} and new input π_{new}), distributions are merged as follows:

$$\pi_{\text{new}}(x) = \begin{cases} (1-\alpha) \cdot \pi_{\text{old}}(x) + \alpha \cdot \pi_{\text{obs}}(x), & x \in \pi_{\text{obs}} \\ \gamma \cdot \pi_{\text{old}}(x), & x \notin \pi_{\text{obs}} \end{cases} \tag{9}$$

The weight $\alpha \in [0, 1]$ controls the trust in the new observation relative to prior knowledge and represents a particular policy:

- $\alpha = 0$: full trust in prior knowledge
- $\alpha = 1$: full trust in new knowledge
- $\alpha = 0.5$: equal weighting (compromise)
- $0 < \alpha < 0.5$: conservative approach prioritizing current knowledge
- $0.5 < \alpha < 1$: prioritizes new knowledge

The parameter $\gamma \geq 1$ applies discount to missing elements, increasing uncertainty.

d) Decay: If no evidence is observed, distributions are gradually degraded by a decay factor $\delta > 1$ to reflect increasing uncertainty over time:

$$\pi'(x) = \begin{cases} \pi(x), & \text{if } x = \arg\max_{x'} \pi(x') \\ \min(1,\ \pi(x) * \delta), & \text{otherwise, where } \delta > 1 \end{cases} \tag{10}$$

e) Surprise: When incoming observations directly contradict a current belief, conflicting entries in the distribution are discounted or removed using factor λ. The specific discounting mechanism depends on the entity type and the available contextual information.

$$\pi(x) \leftarrow \lambda \cdot \pi(x), \quad \text{with } \lambda \ll 1 \tag{11}$$

VI. Reasoning Process

Building upon the foundation of possibility distributions, the reasoning process is structured into three main modules: (1) input process, (2) action evaluation, and (3) hypothesis generation. These modules operate sequentially over each time step, enabling the solution to integrate new information, assess possible developments, and maintain a coherent up-to-date world state representation.

Figure 5 illustrates reasoning loop. It begins from scene graph input, passing through the core reasoning modules, and finishing in an updated state, ready for the next cycle.

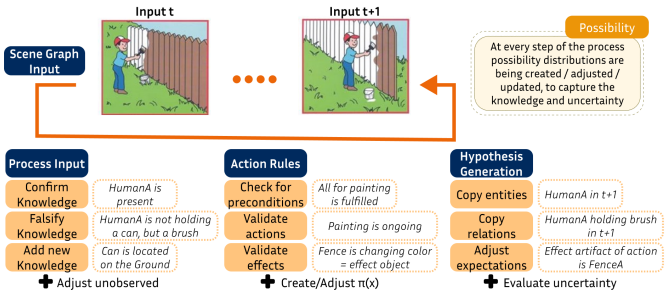


Fig. 5: This figure showcases a single step in the reasoning process. It starts with a scene graph input, which describes the information contained within an image. This is processed by comparing incoming to existing knowledge. Afterwards action rules are applied and hypotheses are generated. As a result at the end of the loop there is set of hypotheses that describe how the world should look like at the next time step.

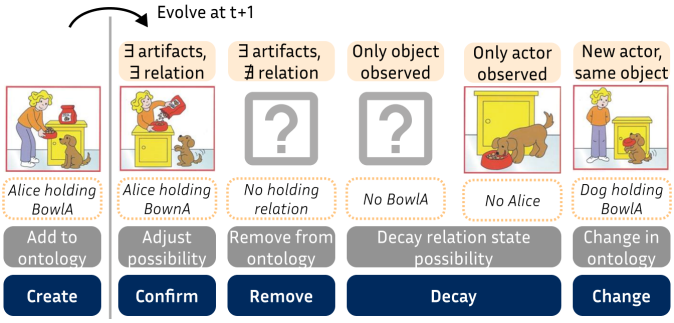


Fig. 6: On the left is the observation provided at time t, which represents the current knowledge. On the right are the possible next observations that can happen, considering the concrete piece of information. The orange block represents the reasoning context, white block is the relevant statement, gray is what happens, and blue is the actual compare relation operation.

A. Input

First step of the process shown in Figure 5 begins at the scene graph input. A single scene graph represents a single instance in time. It contains information about detected entities, attributes, and relationships in a single image frame. In this context each scene graph is interpreted as a snapshot of the world state. Refer to Appendix B for example of how the scene graph itself is constructed.

B. Process Input

To process the information from the scene graph (Process Input step in Figure 5), the system compares it against the current hypothesis and beliefs it has. Knowledge is either confirmed, falsified, or new. This step involves iterating over every statement in the scene graph, and aligning it to the current information in the ontology.

a) Compare Entities: When evaluating entities, the system first checks if an entity with the same unique `id` exists in the knowledge base. Here it is assumed that the scene graph would couple every individual entity it observes with a unique `id` which is kept throughout the whole sequence (e.g. an actor `Sally` always has `id: Sally`). Using this, the system can assess whether the incoming information is about the same individual. If the `id` is the same, then the system checks the observed type. It can be that previously the object was observed as `human`, but is now `child` or `statue`. In such cases, the associated `possibility distribution` is updated to reflect this new evidence. If the entity does not exist in the ontology already, it is added directly - a logic that is also applied to newly introduced attributes.

b) Compare Relations: Evaluating for the two types of relations (special and actions), which can be observed, requires different processing than entities. As shown in Figure 6 (dark blue blocks), there are different outcomes that can come from processing incoming relations.

- **Create**: If the relation is new, create a new instance of the relation (e.g. `painting`) and a corresponding state (e.g. entity `painting state`). The possibility of the relation is encoded within the possibility distribution of the state.
- **Confirm**: If a relation is confirmed (it exists as it is expected), then the possibility distributions are merged according to Eq. 9.
- **Remove**: If explicit evidence appears that a relation does not exist anymore, then it is removed. It is important to note that determining when something does not exist requires clear definition and extra logic.
- **Decay**: When one of the entities is not present (e.g. as shown in Figure 6 the human is no longer visible in case 5), then the possibility of the relation state is decayed according to Eq. 10.
- **Change**: When there is a clear change in the relation (e.g. both human and bowl are present, but someone else is now holding the bowl), then the relation is changed within the ontology. The possibility is updated accordingly.

c) Evaluate Unobserved: If certain elements from the current hypothesis are not observed in the new scene graph (e.g. `Human` is no longer visible), their possibility distributions are decayed following Eq. 10).

C. Action Rules

Next step of the reasoning process in Figure 5 is action evaluation. The core of the reasoning process lies in the generation, tracking and validation of actions. Those serve as hypothesis about possible interactions or events that can occur in the environment. Their modeling relies on a rule-based approach structured in four main steps: preconditions, creation, effect, and validation. Each one of those is explained in the following paragraphs.

a) Preconditions: Each action rule defines a set of preconditions. Those can be logical or relational requirements, that must hold in the current world state for an action to be possible. Here, as seen in Figure 7, preconditions can be relations (e.g. Human holding Brush) or existence of entities

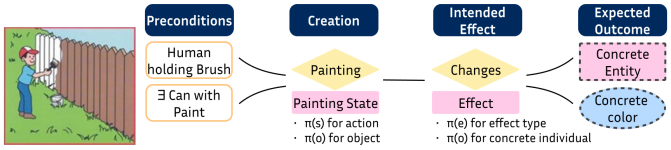


Fig. 7: Example of how the rule of the action painting functions - steps and type of artifacts generated

(e.g. ∃`Paint`). Preconditions can be defined directly within the ontology in the form of rules or constraints. They can also be within another part of the reasoning system (e.g. Python functions).

b) Creation: If there are fulfilled preconditions, then the particular action is considered possible. In this case the assumed action instance is created by adding the following to the ontology:

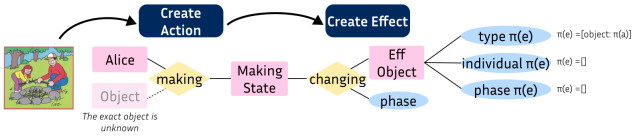
- `Action state` entity (e.g. `painting state`, which represents the assumption that the action is taking place. Its possibility is calculated following Eq. 8.
- `Action` relation between state and key actor. Depending on the concrete action the object of the action is unknown (e.g. what object is being painted) unless otherwise specified.

c) Effect: If an action is assumed possible, its intended effect is generated alongside it. The form of the expected effect of an action is knowledge, which is pre-defined within the system (either in the ontology or as a rule in another form). The effect is characterized by two main things: (1) type of entity that is affected, and (2) property that is affected. Therefore to generate an effect (Figure 8a), the following is added to the ontology:

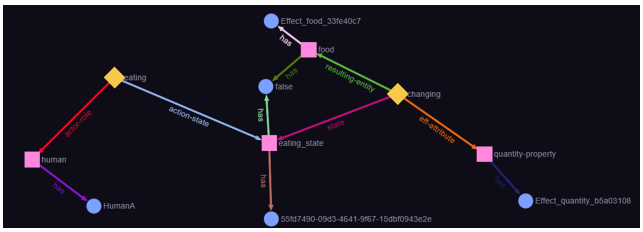
- A property, which is expected to change. An example can be `color` or `quantity`. Depending on the concrete action rules defined within the knowledge, its value can be either unknown (e.g. expected quantity) or derived from the preconditions (e.g. color should be white due to presence of white paint).
- A new entity, introduced as a result of the action. This is defined to be a generic prespecified type (e.g. `object`).

As seen in the right side of Figure 8a, the effect is a special type of entity. It has three distinct possibility distributions, which serve different purposes.

- **Type Distribution:** It defines the type an entity can be (e.g. `can`, `fence`, etc). At definition, unless specified otherwise, this distribution contains only one element: $\pi_{\texttt{effect}} = [\texttt{object}: \pi_{\texttt{action}}]$
- **Instance Distribution:** It defines which is the exact individual within the current world, which fulfills the role of the expected effect (e.g. `Sally` or `Fence_A`). This distribution is initially created empty and it evolves during the reasoning iterations.
- **Attribute Distribution:** It defines the possibility over the expected attribute values (e.g. possible colors after painting). Unless specified otherwise, it is created empty as there is no knowledge available.



(a) Steps to create an effect from a given action



(b) Example of generated assumed effect for Eating action for scenario 3, Image 4 (person holding a hot dog)

Fig. 8: Illustration of effect modeling: (a) steps in creating an effect for a particular action, visualizing the concrete entities and relations developed. (b) generated assumed effect for an action in the reasoning process using the developed ontology.

d) Validation: Once an action and effect are assumed, they must be continuously evaluated against the new information. As every new scene graph introduces new knowledge, so the assumptions need to be checked. There are two mechanisms that are used to govern this process:

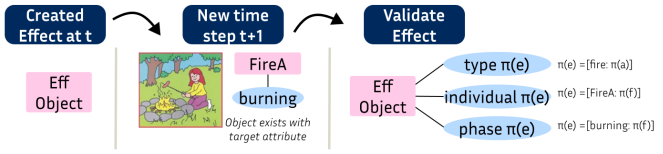


Fig. 9: To validate a generated effect, the system looks at the available knowledge and updates the different possibility distributions accordingly.

- **Action Validity Check:** At each subsequent time step, the system should re-verify whether the preconditions for assumed actions still hold. If not, the action hypothesis is invalidated and decayed. It is not be fully removed from the ontology, as it may still have occurred in the previous time step, but it will not be generated as possible for the next time step.
- **Effect Concretization** The system checks whether an observed object reflects the expected outcome of an action. This is done by monitoring the presence and change in the target attribute in existing entities (Figure 9). The belief that an object is the result of the concrete action is updated based on the value of the key attribute.
  - High possibility if the key attribute is present and there is an expected change (e.g. 0.9).
  - Medium possibility if the key attribute is present, but there is no change yet (e.g. 0.4).

– Los possibility if there is no key attribute (e.g. 0.01).

Based on the attribute value of the compared `object`, the instance is added to the instance distribution of the effect (as the target object can be the created effect). The type and attribute distribution of the effect are updated by merging the information from the `object` with that of the effect.

D. Hypothesis Generation

Once the action rules are evaluated, the last step (right side of Figure 5) is to generate the expected hypothesis for next time step $t+1$. This is derived directly from the current knowledge and the updated possibility distributions. The key assumption underlining this process is that if no plausible actions are currently ongoing, then the world is expected to remain unchanged. The generation process follows these steps for each individual in the knowledge base $\mathcal{K}$:

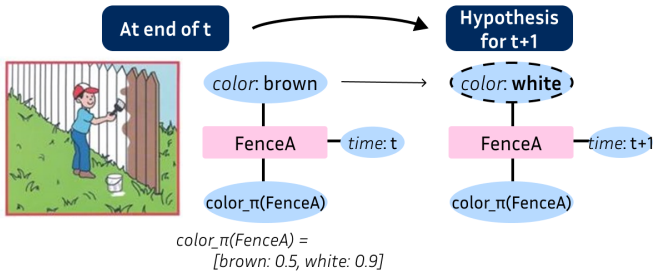


Fig. 10: This image shows how hypothesis generation uses possibility distribution information to determine the next values. In the example case the highest possibility value for the attribute color is white, so the new Fence instance will be created with expected color white instead of brown.

1) **Evaluate Possibility Distributions:** For each entity and its attributes, extract the most plausible value from their respective possibility distributions (as shown in Figure 10. This includes:
   - *Entity Type* (e.g., `can`, `dog`, `bowl`)
   - *Attribute Values* (e.g., `color = white`, `pose = sitting`)

   For entities or attributes which are present, but may not include a possibility distribution (e.g. `location_x`), the entities and attributes are directly copied to the next time step without changes.
2) **Generate Hypothesis:** Use the selected values to create a new instance (or version) of the individual at time $t+1$, annotated with the extracted attributes and relations.
3) **Mark as Assumed:** All generated individuals are tagged as `assumed`, with `observed = false` and timestamp `time-step` $= t+1$, indicating their hypothetical nature.

This approach allows the system to maintain a consistent timeline of evolving hypotheses. The result is that different versions of the same individual across time steps are tracked and linked, enabling temporal reasoning, backtracking, and the evaluation of whether projected expectations were met or contradicted.

VII. Proof of Concept

A first proof of concept is developed based on the described principles in Section VI.

A. Construction

The core functionality of the proposed reasoning system is implemented as a Python script. The underlying ontology is constructed using TypeDB, where the schema defines the structure of entities, attributes, and relations. For each time step, all observed instances are stored as separate instances in the ontology.

Input is provided in the form of pre-processed hand-crafted scene graphs, formatted as JSON files (See Appendix B for examples). These contain entities, attributes, and relations extracted and annotated for a concrete time step. The created ontology contains entities, grouped in four main classes: `actor`, `object`, `state`, and `property`. The first three are used to represent the needed information for action implementation, whereas the latter supports the definition of effects. There are 6 action rules: painting, eating, cooking, putting, giving, making (for concrete definition of each one, refer to Appendix D). There are multiple different attributes, focused on defining possibility distributions and other relevant properties (see Appendix C for complete list).

The main reasoning components, described in Section VI and illustrated in Figure 5 are implemented as modular Python functions. These functions encode the domain rules, action preconditions and effects, and manage the propagation and revision of knowledge over time. Possibility distributions are represented as lists and handled entirely in Python, then stored in the ontology as string attributes for reference and persistence.

B. Parameters

Within the system's reasoning process, there are five key parameters coming from the possibility distribution operations (See Section V-B). Their values directly influence how the system handles uncertainty and belief revision, and are selected to support flexible yet consistent behavior during testing:

- **Confidence of Preconditions** ($c_n = 1$): All preconditions are treated as equally important, due to the lack of differentiated priority information defined within the rules.
- **Decay Factor** ($\delta = 1.1$): It is chosen to introduce gradual uncertainty increase without drastically changing the distribution in a single step.
- **Intention Weight** ($\beta = 1$): This assumes all actions are equally possible, reflecting the absence of contextual intention indicators in the ontology $\mathcal{K}$ or in the scene graph $\mathcal{G}_n$.
- **New Information Weight** ($\alpha = 0.5$): Incoming information from scene graphs is considered equally certain when compared to existing hypotheses.

- **Missing Information Discount** ($\gamma = 1.1$): Similar to the decay, the uncertainty in the partial missing information for a possibility distribution (when there is incoming information about only part of the values present in the current distribution) is gradually introduced.

These parameters form the foundation for how possibility is propagated, merged, or revised as new information arrives. In future iterations, their values could be adjusted dynamically based on observation confidence or environmental context.

C. Test Cases

Three test sequences (Figure 11) evaluate system performance under increasing complexity. Each sequence represents a distinct level of complexity in terms of observed interactions, temporal gaps, and ambiguity.

Fig. 11: The three test cases considered, outlining different scenarios in four pictures. Source: https://www.preschoolmomandkids.pk/product/story-sequencing-4-scene-set-1-2/

- **Basic:** A fully observable, single-action scenario with clear, deterministic transitions. Serves as a baseline to verify core reasoning functionality.
- **Medium:** A two-actor interaction introducing ambiguity in agency and participation. Tests the system's capacity for relational reasoning and distributed effects..
- **Complex:** A temporally sparse, multi-step sequence involving compound actions (e.g., fire, cooking, eating). Evaluates reasoning under uncertainty, incomplete observability, and multiple plausible interpretations.

VIII. Results

To evaluate the performance of the developed solution, a qualitative approach is chosen. As there are no other solutions, which do directly the same type of task, one of the chosen measures is that of an expert. It can be argued, that if a human can explain why a sequence is logical, then the solution should also get high possibility indicators. It is about understanding, making sensible and stable results.

A. Validation Parameters

To validate the process, there are 4 main parameters that are considered for every sequence over all four images:

1) **Confirmed Hypotheses**: the number of hypotheses whose values are explicitly supported or matched in the subsequent observation. In a correct sequence this should be high.
2) **Falsified Hypotheses**: the number of hypotheses contradicted by later observations (e.g., predicted 'color = blue', observed 'color = red'). In a correct sequence this should be low.
3) **Decayed Hypotheses**: hypotheses that remained unconfirmed over time and were gradually discounted due to lack of supporting evidence.
4) **New Knowledge**: the quantity of statements introduced in the new observation that were not present or inferred in the prior state. In a correct sequence the quantity of new (surprise) information should gradually decrease as the system gathers more knowledge about the world.

These metrics allow for analyzing the alignment between the evolving internal world model and the actual incoming scene graphs. High confirmation and low falsification indicate coherent and plausible reasoning. Conversely, a high volume of unexpected new knowledge or decays signals more uncertainty.

B. Overall Quantity

Figure 12 provides a graphical overview of the variation of the parameters, described in the previous section, over the different sequences for Scenario 3 (for the results of Scenario 1 and 2, refer to Appendix E).

As can be observed, there is no clear or consistent pattern that distinguishes one sequence as significantly better than the others. Interestingly, the sequence that achieves the highest number of confirmed hypotheses is not the one that is expected to be correct.

A key insight can be drawn from the behavior of the red line, which represents new or surprising information. Intuitively, in a correct sequence, one would expect that the amount of surprising information is high at the beginning and then gradually decreases. This would reflect the system acquiring most of the new knowledge early and refining it over time. Therefore, in (wrong) sequences like 14 or 20, the opposite happens — the red line spikes or increases over time, suggesting accumulating surprises, which is a strong indicator of inconsistency. In Sequence 20, for instance, the system receives more new information as time progresses, which contradicts the expected temporal flow. This corresponds with expectations as those are incorrect sequences. In contrast, Sequence 1 (the correct sequence) shows a gradual decline in surprises, which better fits the expected behavior. This pattern, however, does not appear to be consistent over all wrong sequences.

It is also important to note that the example in Figure 12 is from the most complex test case and thus expected to have the worst performance. Looking at the results in Appendix E, similar limitations are visible across scenarios. In Scenario 1, the new information (red line) generally decreases over time,

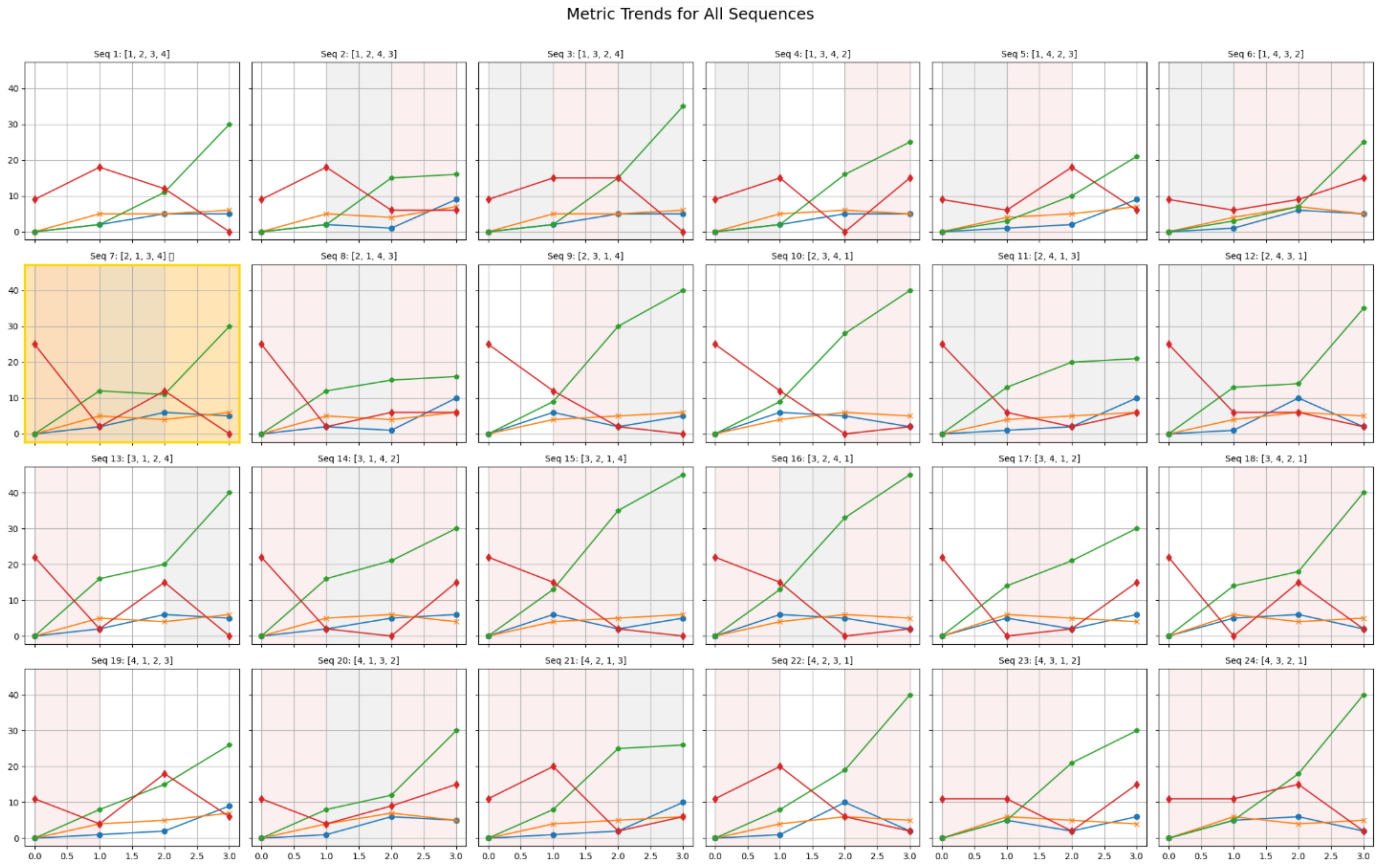


Fig. 12: Analysis for all sequences for Scenario 3. Each graph shows four different metrics: red represents new (surprising) hypotheses; green represents decay; blue represents confirmed hypotheses; orange represents falsified hypotheses. The shading of every graph shows the sequence consistency: white background means the images are in expected consecutive order (e.g. image 1 followed by 2), gray means a step is missed (e.g. image 2 followed by image 4), read means wrong sequence (e.g. image 4 followed by image 3). The graph highlighted in orange is the graph with the highest amount of confirmed hypotheses.

as expected, yet the confirmation and falsification patterns remain too similar across sequences to be informative.

The dominance of the green line, representing decay, is also noteworthy. It likely results from the large temporal gaps between the images. When much changes between time steps, the system struggles to maintain continuity, and information from the previous step is degraded. This stands in contrast to the decay trend seen in Figure 14, where the scene changes less, and observed information remains stable. However, while interesting to observe, it also does not provide a measure based on which one can compare and distinguish sequences. Similarly, the difference between confirmed and falsified hypotheses appears to be too similar between the different cases to be able to create a clear distinction.

Overall, the plots reveal that many sequences produce similar metric trends, especially in terms of rising decay. Confirmed and falsified information does not vary clearly enough to distinguish correct from incorrect temporal orderings. This suggests two possibilities: either the reasoning system is not sufficiently sensitive to temporal correctness, or the current set of metrics is not expressive enough to reflect meaningful differences in sequence quality.

C. Analysis

The overall results offer initial insights, but appear to fall short of providing definitive conclusion about which reasoning sequence is correct. As visible in Figure 12, ambiguity remains, and many variations appear equally plausible, where to a human that would not be the case (even if for some cases it can be argued that many image sequences can be true). This can be explained by looking at three main factors.

a) Evaluation metrics: Current metrics count confirmed and falsified hypotheses equally, regardless of the underlying possibility values. A minor confirmation ($\pi = 0.1$) is weighted the same as a strong one ($\pi = 0.7$), weakening the distinction between plausible and implausible outcomes. The assumption that correct sequences yield more confirmations does not consistently hold.

b) Variability: Possibility parameters (See Section VII-B) are fixed across time and context. For instance, incoming data and internal hypotheses are weighted equally, and all actions are treated as if they have equal intention (a human is just as likely to paint as they are to

eat). This uniformity reduces sensitivity to context and limits the system's ability to prioritize.

c) Action Under-Specification: The current system includes only six predefined actions with simplistic rules. These lack contextual constraints such as object affordances or resource availability (beyond simple existence of a resource). As a result, actions are often inferred in cases where a human would not expect them. Limited test case diversity further restricts meaningful validation.

IX. Discussion

This work presents a novel approach to scene-based reasoning using possibility theory. While the proposed framework demonstrates promising capabilities, the research on this topic is only at the beginning. There are several limitations that must be acknowledged which point to specific directions for future research.

A. Limitations

The current implementation constitutes a proof of concept developed under a number of simplifying assumptions and design decisions. While this supports initial experimentation, it limits expressiveness and real-world applicability.

a) Input Generation: Scene graphs are manually created and assumed to be complete and accurate. This introduces bias and overlooks the uncertainty and noise present in computerized data extraction. The current design therefore may assume a higher (or lower) fidelity and information density than would likely be encountered in practical settings.

b) Ontology Consistency: Reasoning is implemented procedurally in Python, with limited to no integration into the ontology itself. As a result, consistency checks and contradiction detection are limited to explicit assertions and do not account for implicit inconsistencies inferred from the ontology structure.

c) Possibility Modeling: Although possibility distributions form the core of the reasoning approach, the parameters that control their behavior (e.g., α for belief merging) are not yet extensively tuned or explored. This limits the system's ability to reflect nuanced belief dynamics.

B. Future Work

Future work can address current limitations through enhancements across evaluation, modeling, and system architecture.

a) Validation Metrics: The current evaluation, while qualitative, is based on simplistic metrics. Introducing possibility values into the metrics evaluation would provide a clearer overview of the performance. Distinguishing between different types of confirmation and falsification, based on the difference in possibility values (e.g. 0.1 versus 0.6) can provide a less binary view on hypothesis. Additional metrics can also be introduced such as trends in $\pi(x)$, changes in values and consistency in results. This can introduce an extra quantitative exploration of the achieved behavior.

b) Richer Parameter Dynamics and World Representation: Future implementations should tune key parameters $(\alpha, \beta, \gamma, \delta, c_n)$ dynamically. This can be done based on additional defined knowledge within the ontology. This can for example further define what is more or less expected (e.g. human painting fence versus human painting grass) Alternative could be assessing available incoming information (e.g. contextual factors, such as observation confidence, human intent, or known sensor reliability). For instance, if a perception module is known to misclassify the `holding` relation, its impact on hypothesis confidence should be appropriately discounted.

c) Enhanced Action Modeling and Constraints: As actions are at the base of the reasoning algorithms, their extension should be a focus in future research. There can be different directions for that. One can introduce compound actions (e.g. feeding = giving + eating). There can be conditional effects based on preconditions (e.g. color of paint influences expected color change). Affordance-based filtering can help generate more concrete effects (e.g. only certain objects can be painted). Action can have different types of effects depending on extra conditions that occur in different time steps. Actions can also be inferred based on an observed effect rather than preconditions (a fence has changed color compared to previous time step, so painting must have occurred). Such extensions could enhance interpretability and support adding capabilities such as abductive reasoning.

X. Conclusion

This work introduced a reasoning framework based on possibility theory. Its goal was to show how concepts from possibility theory can be applied to open-world reasoning tasks, where information is often uncertain or incomplete. The main challenge was scoped to a concrete task. The aim was to develop a solution, which can determine the correct sequence of incoming scene graphs by understanding what happens in the world and what is realistically possible.

The developed solution relies on a structured world model, built as an ontology, which provides knowledge about how the world could look like. This model is extended with possibility distributions that capture beliefs about the types and attributes of entities, providing a way to model uncertainty along with knowledge. This ontology is used as a starting point for a reasoning process, which can process an incoming scene graph. It compares the incoming information to its current knowledge, applies action rules, and generates new hypotheses for the next time step. At each step, possibility distributions guide how the system evaluates plausibility, tracks belief changes, and responds to new information.

While the final results were not as strong as expected, the work marks an important first step. It shows that possibility theory can offer a structured and flexible foundation for reasoning in uncertain environments. Even in its early form, the approach demonstrates potential. This opens several directions for future work—both in improving the technical implementation and exploring more advanced reasoning use cases.

ACKNOWLEDGMENT

I would like to express my thanks to my supervisor Dr. Joris Sijs, for his invaluable guidance, thoughtful constructive feedback, and the many inspiring and insightful discussions that shaped this work. I am also deeply grateful to my partner and family for his continuous support and encouragement throughout this project.

REFERENCES

[1] T. Taniguchi, S. Murata, M. Suzuki, D. Ognibene, P. Lanillos, E. Ugur, L. Jamone, T. Nakamura, A. Ciria, B. Lara, and G. Pezzulo, "World models and predictive coding for cognitive and developmental robotics: frontiers and challenges," *Advanced Robotics*, vol. 37, no. 13, pp. 780–806, Jul. 2023. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/01691864.2023.2225232

[2] B. Liu, S. Mazumder, E. Robertson, and S. Grigsby, "AI Autonomy: Self-initiated Open-world Continual Learning and Adaptation," *AI Magazine*, vol. 44, no. 2, pp. 185–199, Jun. 2023. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/aaai.12087

[3] N. Akai, T. Hirayama, and H. Murase, "Semantic Localization Considering Uncertainty of Object Recognition," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4384–4391, Jul. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9103264/

[4] M. Sridharan and B. Meadows, "Should I do that? using relational reinforcement learning and declarative programming to discover domain axioms," *2016 Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, pp. 252–259, Sep. 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7846827/

[5] J. Sijs and J. Fletcher, "A robotic knowledge base to model and update real-world information from indoor environments," in *2022 25th International Conference on Information Fusion (FUSION)*, Jul. 2022, pp. 1–8. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9841262

[6] M. Hanheide, M. Göbelbecker, G. S. Horn, A. Pronobis, K. Sjöö, A. Aydemir, P. Jensfelt, C. Gretton, R. Dearden, M. Janicek, H. Zender, G.-J. Kruijff, N. Hawes, and J. L. Wyatt, "Robot task planning and explanation in open and uncertain worlds," *Artificial Intelligence*, vol. 247, pp. 119–150, Jun. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S000437021500123X

[7] R. Paul, A. Barbu, S. Felshin, B. Katz, and N. Roy, "Temporal Grounding Graphs for Language Understanding with Accrued Visual-Linguistic Context," Nov. 2018, arXiv:1811.06966 [cs]. [Online]. Available: http://arxiv.org/abs/1811.06966

[8] Y. LeCun, "A path towards autonomous machine intelligence," Jun. 2022. [Online]. Available: https://openreview.net/forum?id=BZ5a1r-kVsf

[9] M. Stefik and R. Price, "Bootstrapping Developmental AIs: From Simple Competences to Intelligent Human-Compatible AIs," Apr. 2024, arXiv:2308.04586 [cs]. [Online]. Available: http://arxiv.org/abs/2308.04586

[10] P. Wu, A. Escontrela, D. Hafner, P. Abbeel, and K. Goldberg, "DayDreamer: World Models for Physical Robot Learning," in *Proceedings of The 6th Conference on Robot Learning*. PMLR, Mar. 2023, pp. 2226–2240. [Online]. Available: https://proceedings.mlr.press/v205/wu23c.html

[11] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, P. Florence, C. Fu, M. G. Arenas, K. Gopalakrishnan, K. Han, K. Hausman, A. Herzog, J. Hsu, B. Ichter, A. Irpan, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, L. Lee, T.-W. E. Lee, S. Levine, Y. Lu, H. Michalewski, I. Mordatch, K. Pertsch, K. Rao, K. Reymann, M. Ryoo, G. Salazar, P. Sanketi, P. Sermanet, J. Singh, A. Singh, R. Soricut, H. Tran, V. Vanhoucke, Q. Vuong, A. Wahid, S. Welker, P. Wohlhart, J. Wu, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich, "RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control," 2023. [Online]. Available: https://arxiv.org/abs/2307.15818

[12] P. Liu, Y. Orru, J. Vakil, C. Paxton, N. M. M. Shafiullah, and L. Pinto, "OK-Robot: What Really Matters in Integrating Open-Knowledge Models for Robotics," Feb. 2024, arXiv:2401.12202 [cs]. [Online]. Available: http://arxiv.org/abs/2401.12202

[13] R. Firoozi, J. Tucker, S. Tian, A. Majumdar, J. Sun, W. Liu, Y. Zhu, S. Song, A. Kapoor, K. Hausman, B. Ichter, D. Driess, J. Wu, C. Lu, and M. Schwager, "Foundation Models in Robotics: Applications, Challenges, and the Future," Dec. 2023, arXiv:2312.07843 [cs]. [Online]. Available: http://arxiv.org/abs/2312.07843

[14] V. Lohweg, K. Voth, and S. Glock, "A Possibilistic Framework for Sensor Fusion with Monitoring of Sensor Reliability," in *Sensor Fusion - Foundation and Applications*. IntechOpen, Jun. 2011. [Online]. Available: https://www.intechopen.com/chapters/14978

[15] H. Aloulou, M. Mokhtari, T. Tiberghien, R. Endelin, and J. Biswas, "Uncertainty handling in semantic reasoning for accurate context understanding," *Knowledge-Based Systems*, vol. 77, pp. 16–28, Mar. 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705114004717

[16] G. J. Klir, *Uncertainty and Information: Foundations of Generalized Information Theory*, 1st ed. Wiley, Nov. 2005. [Online]. Available: https://onlinelibrary.wiley.com/doi/book/10.1002/0471755575

[17] L. A. Zadeh, "Fuzzy sets as a basis for a theory of possibility," *Fuzzy Sets and Systems*, vol. 100, pp. 9–34, Jan. 1999. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0165011499800049

[18] D. Dubois and H. Prade, "Possibility Theory," in *Encyclopedia of Complexity and Systems Science*, R. A. Meyers, Ed. New York, NY: Springer, 2014, pp. 1–20. [Online]. Available: https://doi.org/10.1007/978-3-642-27737-5_413-2

[19] ——, "A crash course on generalized possibilistic logic," in *International Conference on Scalable Uncertainty Management (SUM 2018)*. Milan, Italy: Springer, 2018, pp. 3–18, hAL: hal-02181919. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02181919

[20] ——, "Possibility Theory and Possibilistic Logic: Tools for Reasoning Under and About Incomplete Information," Z. Shi, M. Chakraborty, and S. Kar, Eds., vol. 623. Cham: Springer International Publishing, 2021, pp. 79–89. [Online]. Available: https://link.springer.com/10.1007/978-3-030-74826-5_7

[21] ——, "Practical Methods for Constructing Possibility Distributions: CONSTRUCTING POSSIBILITY DISTRIBUTIONS," *International Journal of Intelligent Systems*, vol. 31, no. 3, pp. 215–239, Mar. 2016. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/int.21782

[22] T. Denœux, D. Dubois, and H. Prade, "Representations of Uncertainty in Artificial Intelligence: Probability and Possibility," P. Marquis, O. Papini, and H. Prade, Eds. Cham: Springer International Publishing, 2020, pp. 69–117. [Online]. Available: http://link.springer.com/10.1007/978-3-030-06164-7_3

[23] D. Dubois and H. Prade, "Reasoning and learning in the setting of possibility theory - Overview and perspectives," *International Journal of Approximate Reasoning*, vol. 171, p. 109028, Aug. 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0888613X23001597

[24] K. Demirli, M. Molhim, and A. Bulgak, "POSSIBILISTIC SONAR DATA MODELING FOR MOBILE ROBOTS," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 07, no. 02, pp. 173–198, Apr. 1999. [Online]. Available: https://www.worldscientific.com/doi/abs/10.1142/S0218488599000118

[25] P. Fuster-Parra, J. Guerrero, J. Martín, and Valero, "New Results on Possibilistic Cooperative Multi-robot Systems," in *Cooperative Design, Visualization, and Engineering*, Y. Luo, Ed. Cham: Springer International Publishing, 2017, pp. 1–9.

[26] D. Dubois and H. Prade, "Possibilistic Logic: From Certainty-Qualified Statements to Two-Tiered Logics – A Prospective Survey," F. Calimeri, N. Leone, and M. Manna, Eds., vol. 11468. Cham: Springer International Publishing, 2019, pp. 3–20. [Online]. Available: http://link.springer.com/10.1007/978-3-030-19570-0_1

[27] D. Dubois, J. Lang, and H. Prade, "Possibilistic Logic," in *Handbook of Logic in Artificial Intelligence and Logic Programming*, D. M. Gabbay, C. J. Hogger, J. A. Robinson, and D. Nute, Eds. Oxford University PressOxford, Mar. 1994, pp. 439–513. [Online]. Available: https://academic.oup.com/book/53007/chapter/421965136

APPENDIX A
POSSIBILITY PRIMER

Possibility theory is a mathematical framework for representing and reasoning with uncertain, imprecise, or incomplete information. It was introduced by [17] as a qualitative counterpart to probability theory. It was formalized by [27] and further developed by [18]. It provides a foundation particularly well-suited to cases where information is vague or incomplete, rather than random.

A. Main Concepts

Possibility theory defines two dual measures of uncertainty for an event A (a subset of a universe of discourse U):

- **Possibility measure** $\Pi(A)$: Indicates the degree to which event A is plausible: i.e., not contradicted by the available knowledge.
- **Necessity measure** $N(A)$: Indicates the degree to which event A is certain: i.e., guaranteed by the available knowledge.

These two measures are connected by the duality:

$$N(A) = 1 - \Pi(\neg A)$$

where $\neg A$ denotes the complement of A in the universal set U. There are key axioms [14] that define the notion of possibility and necessity. For the arbitrary events $A, B \subseteq U$:

- $\Pi(\emptyset) = 0$: The empty event is impossible.
- $N(\emptyset) = 0$: The empty event is never necessary.
- $\Pi(U) = 1$: The universal event is fully possible.
- $N(U) = 1$: The universal event is necessarily true.
- $\Pi(A \cup B) = \max(\Pi(A), \Pi(B))$: The possibility of a union is the maximum of the individual possibilities (maxitivity).
- $N(A \cap B) = \min(N(A), N(B))$: The necessity of a conjunction is the minimum of the individual necessities (minitivity).

B. Possibility Distribution

The possibility theory uses possibility distribution π to represent the state of knowledge of an agent over a variable X taking values in a domain $\mathcal{X}$.

$$\pi : \mathcal{X} \to [0, 1]$$

For each value $x \in \mathcal{X}$, $\pi(x)$ represents the degree of compatibility or plausibility that x is the true value of X, given the available knowledge. A value of $\pi(x) = 1$ indicates full plausibility, while $\pi(x) = 0$ means x is considered impossible. This can be used, following the formalization of [18] to evaluate the possibility, denoted Π with $\Pi : 2^{\Theta} \to [0, 1]$, of an event A using

$$\Pi(A) = \sup_{s \in A} \pi(s)$$

It is used to evaluate consistency where $\Pi(A)$ represents the degree to which A is considered plausible (possible) given the possibility distribution π. Mathematically this means looking for the supremum (sup) within π.

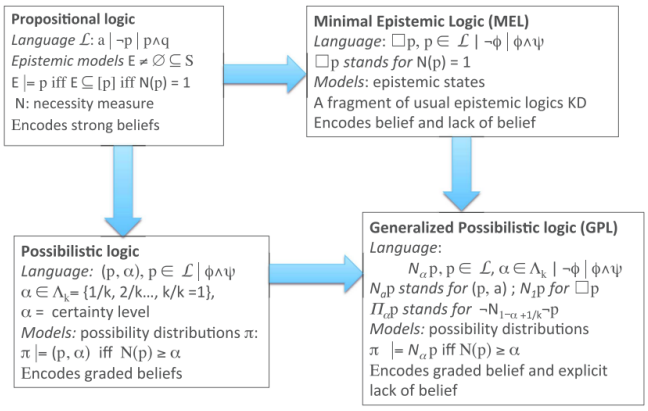


Fig. 13: Comparison of different belief logics in possibility theory; Figure 1 from [19].

Similarly, necessity N quantifies the degree of certainty of an event A using

$$N(A) = \inf_{s \notin A} 1 - \pi(s)$$

This evaluates whether a given event A is implied by the possibility distribution π. Mathematically this means looking for the infimum (inf) within π.

Those two measures are at the foundation of the possibility logic, where formulas are represented as a pair of propositional formula α and certainty degree λ of the form (α, λ).

This notion is formalized by [19] into Generalized possibilistic logic, shown in Figure 13.

C. Comparison with Probability Theory

Possibility theory differs fundamentally from probability theory in how uncertainty is aggregated:

- **Additivity (Probability):** $P(A \cup B) = P(A) + P(B) - P(A \cap B)$
- **Maxitivity (Possibility):** $\Pi(A \cup B) = \max(\Pi(A), \Pi(B))$
- **Ignorance:** Possibility theory can represent complete ignorance by assigning $\pi(x) = 1$ for all x, something probability theory cannot do without assigning a uniform distribution (which implies knowledge of equality).

APPENDIX B
SCENE GRAPH INPUT

This appendix provides the structured scene graph input data used for reasoning and validation throughout the scenarios. Each image is represented as a graph consisting of nodes (entities) and edges (relations) with associated attributes and confidence values.

A. Notation

- **Node:** An entity with ID, type, and attribute list.
- **Edge:** A directional relation between two entities.
- **Attributes:** Named properties with values and confidence scores.

The following pages contain representative JSON-formatted inputs for three example scenarios.

B. Scenario 1 - Image 3 - Painting


```
{"nodes":[
   {"id":"HumanA", "type":"human", "confidence":0.98,
    "attributes":[
      {"name":"pose", "value":"standing", "confidence":1.0 }
           ,
      {"name":"location_x", "value":2.9, "confidence":1.0 },
      {"name":"location_y", "value":0.5, "confidence":1.0 }
    ]
   },
   {"id":"CanA", "type":"can", "confidence":0.92,
    "attributes":[
      {"name":"location_x", "value":2.8, "confidence":0.9 },
      {"name":"location_y", "value":0.45, "confidence":0.9 }
    ]
   },
   {"id":"FenceA", "type":"fence", "confidence":0.95,
    "attributes":[
      {"name":"color", "value":"white", "confidence":0.9 },
      {"name":"color", "value":"brown", "confidence":0.2 },
      {"name":"location_x", "value":3.8, "confidence":1.0 },
      {"name":"location_y", "value":0.5, "confidence":1.0 }
    ]
   },
   {"id":"WhitePaint", "type":"paint", "confidence":0.85,
    "attributes":[
      {"name":"color", "value":"white", "confidence":0.8 }
    ]
   },
   {"id":"BrushA", "type":"brush", "confidence":0.92,
    "attributes":[
      {"name":"location_x", "value":2.9, "confidence":0.95 }
           ,
      {"name":"location_y", "value":0.65, "confidence":0.95
           }
    ]
   },
   {"id":"Ground", "type":"ground", "confidence":0.95 }
  ],
  "edges":[
    {"source":"HumanA", "target":"BrushA", "relation":"
        holding", "confidence":0.95 },
    {"source":"HumanA", "target":"FenceA", "relation":"
        painting", "confidence":0.9 },
    {"source":"HumanA", "target":"FenceA", "relation":"facing
        ", "confidence":0.95 },
    {"source":"CanA", "target":"Ground", "relation":"
        isLocated", "confidence":0.9},
    {"source":"CanA", "target":"WhitePaint", "relation":"
        contain", "confidence":0.6 }
  ]
}
```


C. Scenario 2 - 2: Pouring Food


```
{
  "nodes":[
    {"id":"HumanA", "type":"human", "confidence":0.98,
     "attributes":[
       {"name":"pose", "value":"standing", "confidence":0.9 }
            ,
       {"name":"location_x", "value":0.5, "confidence":1.0 },
       {"name":"location_y", "value":0.5, "confidence":1.0 }
     ]
    },
    {"id":"DogA", "type":"dog", "confidence":0.95,
     "attributes":[
       {"name":"pose", "value":"standing", "confidence":0.85
            },
       {"name":"location_x", "value":0.6, "confidence":1.0 },
       {"name":"location_y", "value":0.5, "confidence":1.0 }
     ]
    },
    {"id":"Bowl", "type":"bowl", "confidence":0.85,
     "attributes":[
       {"name":"location_x", "value":0.6, "confidence":1.0 },
       {"name":"location_y", "value":0.45, "confidence":1.0 }
     ]
    },
    {"id":"Shelf", "type":"shelve", "confidence":0.85,
     "attributes":[
       {"name":"location_x", "value":0.6, "confidence":1.0 },
       {"name":"location_y", "value":0.4, "confidence":1.0 }
     ]
    },
    {"id":"Bag", "type":"bag", "confidence":0.9},
    {"id":"FoodA", "type":"food", "confidence":0.8,
     "attributes":[
       {"name":"quantity", "value":"10", "confidence":0.95}
     ]}
  ],
  "edges":[
    {"source":"HumanA", "target":"Bag", "relation":"holding",
         "confidence":0.85},
    {"source":"Bag", "target":"Shelf", "relation":"isLocated"
        , "confidence":0.85},
    {"source":"HumanA", "target":"Bag", "relation":"pouring",
         "confidence":0.75},
    {"source":"Bowl", "target":"Shelf", "relation":"isLocated
        ", "confidence":0.9},
    {"source":"Bag", "target":"FoodA", "relation":"contain",
         "confidence":0.75},
    {"source":"HumanA", "target":"DogA", "relation":"facing",
         "confidence":0.8}
  ]
}
```


D. Scenario 3 - Image 3 - Cooking


```
{
  "nodes":[
    {"id":"HumanA", "type":"human", "confidence":0.98,
     "attributes":[
       {"name":"pose", "value":"sitting", "confidence":0.9 },
       {"name":"location_x", "value":0.5, "confidence":1.0 },
       {"name":"location_y", "value":0.55, "confidence":1.0 }
     ]
    },
    {"id":"Hotdog", "type":"meal", "confidence":0.9,
     "attributes":[
       {"name":"state_content", "value":"uncooked",
            "confidence":0.87}
     ]
    },
    {"id":"FoodA", "type":"food", "confidence":0.9,
     "attributes":[
       {"name":"quantity", "value":"1", "confidence":0.95},
       {"name":"state_content", "value":"uncooked",
            "confidence":0.87}
     ]
    },
    {"id":"Stick", "type":"stick", "confidence":0.85},
    {"id":"Rocks", "type":"rock", "confidence":0.8},
    {
     "id":"Wood", "type":"wood", "confidence":0.9,
     "attributes":[
       {"name":"quantity", "value":"multiple", "confidence":0
            .95 }
     ]
    },
    {"id":"Ground", "type":"ground", "confidence":0.95},
    {
     "id":"Fire", "type":"fire", "confidence":0.9,
     "attributes":[
       {"name":"state_content", "value":"burning",
            "confidence":0.87}
     ]
    }
  ],
  "edges":[
    {"source":"HumanA", "target":"Stick", "relation":"holding
        ", "confidence":0.9 },
    {"source":"Hotdog", "target":"Stick", "relation":"
        isLocated", "confidence":0.85 },
    {"source":"Hotdog", "target":"FoodA", "relation":"contain
        ", "confidence":0.85 },
    {"source":"Rocks", "target":"Ground", "relation":"
        isLocated", "confidence":0.85 },
    {"source":"HumanA", "target":"FoodA", "relation":"cooking
        ", "confidence":0.9},
    {"source":"Fire", "target":"Wood", "relation":"contain",
         "confidence":0.9}
```

```
  ]
}
```

APPENDIX C
ONTOLOGY STRUCTURE

This appendix presents the ontology structure as implemented in TypeDB. The ontology defines the conceptual schema for representing entities, actions, and reasoning artifacts used throughout the system.

ENTITIES: ACTORS AND OBJECTS

```
entity actor;
   entity human sub actor;
      entity adult sub human;
      entity child sub human;
   entity animal sub actor;
      entity dog sub animal;

entity object;
   entity static sub object;
      entity table sub static;
      entity shelve sub static;
      entity ground sub object;
      entity room sub object;
      entity fire sub object,
         owns fase;

   entity constituary sub object;
      entity fence sub constituary;
      entity brush sub constituary;
      entity rock sub constituary;
      entity stick sub constituary;
      entity content-holding sub constituary;
         entity cup sub content-holding;
         entity bowl sub content-holding;
         entity bag sub content-holding;
         entity bucket sub content-holding;
         entity stack sub content-holding;

entity content;
   entity food sub content;
   entity wood sub content;
   entity paint sub content;
```

ACTIONS

```
relation action,
   relates actor-role,
   relates object-role,
   relates action-state;

actor plays action:actor-role;
object plays action:object-role;
actor plays action:object-role;
state plays action:action-state;

   relation holding sub action;
   relation placing sub action;
   relation giving sub action;
   relation eating sub action;
   relation painting sub action;
   relation cooking sub action;
   relation making sub action;
```

STATES

```
entity state,
   entity holidng_state sub state;
   entity placing_state sub state;
   entity giving_state sub state;
   entity eating_state sub state;
   entity painting_state sub state;
   entity cooking_state sub state;
   entity making_state sub state;
```

EFFECT

```
relation changing,
   relates state,
   relates resulting-entity,
   relates eff-attribute;

entity property,
   entity color-property sub property;
   entity quantity-property;
   entity phase-property;

state plays changing:state;
object plays changing:resulting-entity;
actor plays changing:resulting-entity;
property plays changing:eff-attribute;
```

RELATIONS: SPATIAL AND CONTAINMENT

```
relation isLocated,
   relates actor-role,
   relates object-role,
   relates action-state;

object plays isLocated:object-role;
object plays isLocated:actor-role;
actor plays isLocated:actor-role;
isLocated_state plays isLocated:action-state;

relation contain,
   relates actor-role,
   relates object-role,
   relates action-state;

content-holding plays contain:actor-role;
content plays contain:object-role;
contain_state plays contain:action-state;

relation facing,
   relates actor-role,
   relates object-role,
   relates action-state;

actor plays facing:actor-role;
object plays facing:object-role;
actor plays facing:object-role;
facing_state plays facing:action-state;
```

ATTRIBUTES - GENERAL

```
attribute originates, value string;
attribute id, value string;
attribute assumed, value boolean;
attribute observed, value string;
```

```
attribute time_step, value string;
attribute effect, value string;

attribute color, value string;
attribute distance, value string;
attribute fase, value string;
attribute quantity, value string;
attribute pose, value string;

attribute location_x, value string;
attribute location_y, value string;
```

ATTRIBUTES - POSSIBILITY DISTRIBUTION

```
attribute possibility_distribution, value
    string;
attribute possibility_value_state, value
    string;
attribute color_possibility_distribution,
    value string;
attribute pose_possibility_distribution, value
     string;
attribute fase_possibility_distribution, value
     string;
attribute quantity_possibility_distribution,
    value string;
attribute individual_possibility_distribution,
     value string;
```

ATTRIBUTE OWNERSHIP

```
actor owns originates, id, assumed, observed,
    time_step, effect, distance, pose,
    location_x, location_y,
    possibility_distribution,
    individual_possibility_distribution,
    pose_possibility_distribution;

object owns originates, id, assumed, observed,
     time_step, effect, distance, pose,
    location_x, location_y, color, quantity,
    fase, possibility_distribution,
    individual_possibility_distribution,
    color_possibility_distribution,
    quantity_possibility_distribution,
    fase_possibility_distribution;

state owns originates, id, assumed, observed,
    time_step, effect,
    possibility_distribution,
    possibility_value_state;

content owns originates, id, assumed, observed
    , time_step, effect, location_x,
    location_y, possibility_distribution,
    quantity, color,
    color_possibility_distribution, quantity,
    quantity_possibility_distribution;

property owns originates, assumed, id,
    time_step;
```

APPENDIX D
ACTION DEFINITION

There are six core actions, that are defined in the developed proof of concept. For every action, aside from its definition within the ontology, there is a class created in Python. It contains clear definition of the different preconditions, expected effects and validation principles.

A. *Painting Action*

The painting action is the key activity in Scenario 1. It has three main defined preconditions and one effect. For the current implementation it is considered that painting changes the color of an object, disregarding cases where one paints with the same color.

TABLE I: Action Template: Painting

Aspect	**Description**
Name	Painting Action
Definition	The action of changing the color of an object.
Preconditions	(1) ∃ Actor, (2) ∃ Paint, (3) ∃ Brush
Effects	An object with a different color form originally observed.
Validation Principle	Check for object, whose color has changed compared to previous time step. Alternatively look for object with the same color.

B. *Cooking Action*

The cooking action is an activity present in Scenario 3. It has three main defined preconditions and one effect. For here only one method of cooking is considered: fire.

TABLE II: Action Template: Cooking

Aspect	**Description**
Name	Cooking
Definition	Change the state of food from uncooked to cooked.
Preconditions	(1) ∃ Actor, (2) ∃ Food, (3) ∃ Fire
Effects	Food changes state from uncooked to cooked.
Validation Principle	Check for food, whose state has changed from uncooked to cooked. Alternatively look for food that is cooked.

C. *Eating Action*

The eating action is part of Scenario 2 and 3. It can be executed by different actors and it results in change in quantity of food. It has three preconditions.

TABLE III: Action Template: Eating

Aspect	**Description**
Name	Eating
Definition	Consuming quantity of food by an actor
Preconditions	(1) ∃ Actor, (2) ∃ Food, (3) Food is cooked
Effects	Food quantity decreases.
Validation Principle	Check for food, whose quantity has changed (decreased) compared to the previous time step.

D. Placing Action

The placing action is part of Scenario 2 and 3. It can be executed in different actors and it results in change in location of an object. It has three preconditions. The exact placing action can be defined in different ways, depending on the exact goal. In this case this action was aimed to support the action of placing the bag of food on the shelf (as needed in Scenario 2) or the placing of wood on the floor for the fire (as needed in Scenario 3).

TABLE IV: Action Template: Placing

Aspect	Description
Name	Placing
Definition	Change location of an object
Preconditions	(1) ∃Actor, (2) ∃object, (3) Actor is holding object
Effects	Position and/or pose and/or distance of object is changed
Validation Principle	Check change in distance and pose of one object compared to others and the absence of a holding relation.

E. Giving Action

The giving action is a special action, which showcases an interaction between two actors and a key object. It is especially crucial for Scenario 2, where the person is giving food to the dog.

TABLE V: Action Template: Giving

Aspect	Description
Name	Giving
Definition	Provide an object to another actor
Preconditions	(1)∃ActorA, (2) ∃ ActorB, (3) ∃ Object, (4) ActorA is holding Object
Effects	Change location of Object and its proximity to ActorB
Validation Principle	Check change in pose, distance, location of an object compared to ActorB. Check for holding relation with ActorA.

Giving can include transferring of different types of objects between different actors.

F. Making Action

In the context of this work the making action is focused on Scenario 3 and the making of a fire. It involves creating a new object.

TABLE VI: Action Template: Making

Aspect	Description
Name	Making
Definition	Making a new object (fire) from given materials.
Preconditions	(1) ∃ Actor, (2) ∃ Wood
Effects	An object fire is created.
Validation Principle	Check for presence of fire. Secondary check if fire has been lid compared to previous time step.

In the general case making would have different conditions, as it would need to cover making of different objects. In that context making can then be considered as needed an actor to execute the action, an object with a particular affordance, and needed material.

APPENDIX E
FULL SEQUENCE RESULTS

While Section VIII describes the results and presents one of the test cases (Scenario 3), here the complete sequence results are provided for the other two situations. As can be observed, both Figure 14 and Figure 15 show similar results to what is described in Section VIII. There is no clear indication that puts one sequence to be significantly more distinctive than the others. It can be seen that confirmed and falsified hypothesis remain relatively close to each other. At the same time, new (surprising) information (shown in red) is less fluctuating than compared to Scenario 3 in Figure 12.

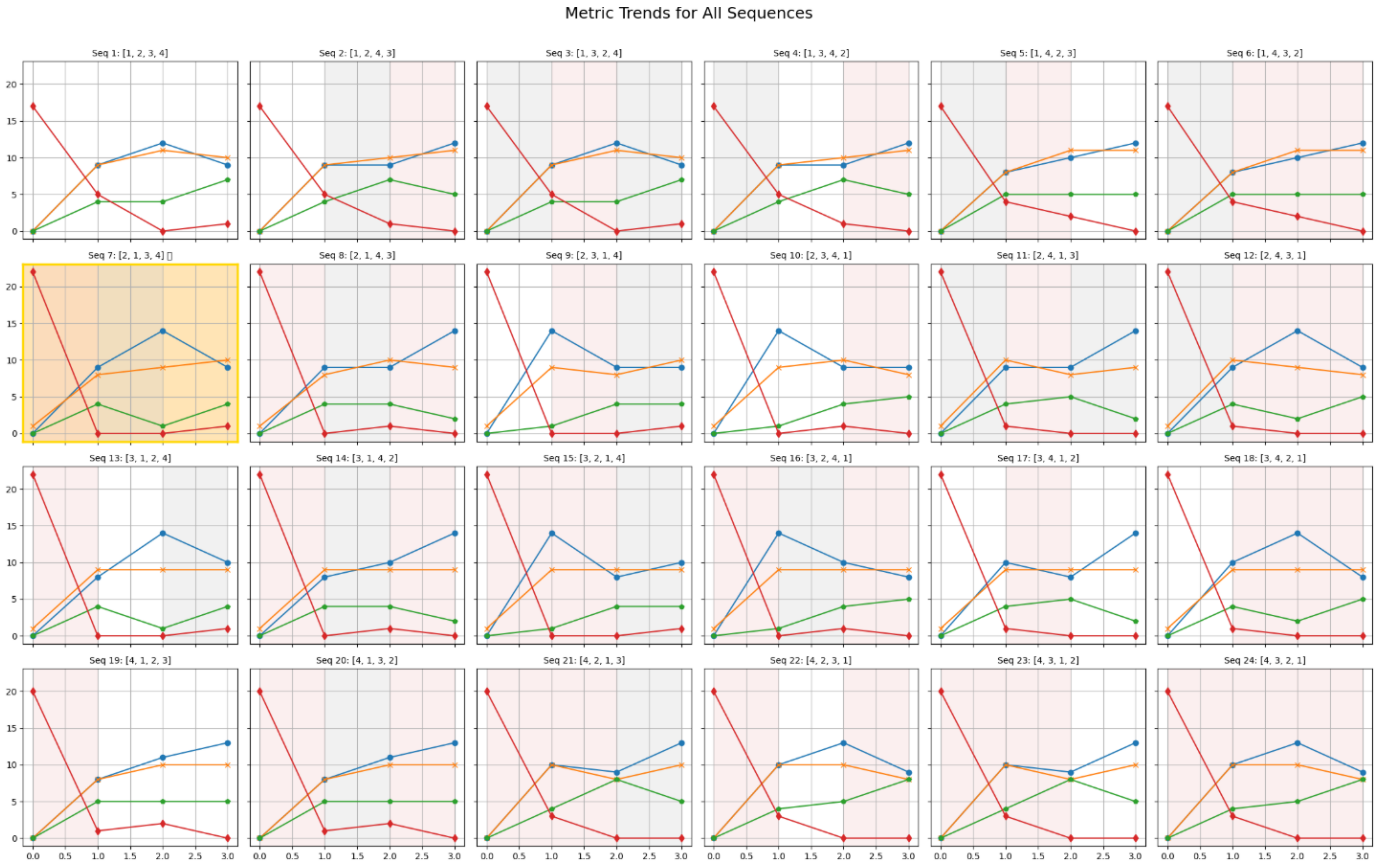


Fig. 14: Analysis for all sequences for Scenario 1. Each graph shows four different metrics: red represents new (surprising) hypotheses; green represents decay; blue represents confirmed hypotheses; orange represents falsified hypotheses. The shading of every graph shows the sequence consistency: white background means the images are in expected consecutive order (e.g. image 1 followed by 2), gray means a step is missed (e.g. image 2 followed by image 4), read means wrong sequence (e.g. image 4 followed by image 3). The graph highlighted in orange is the graph with the highest amount of confirmed hypotheses.

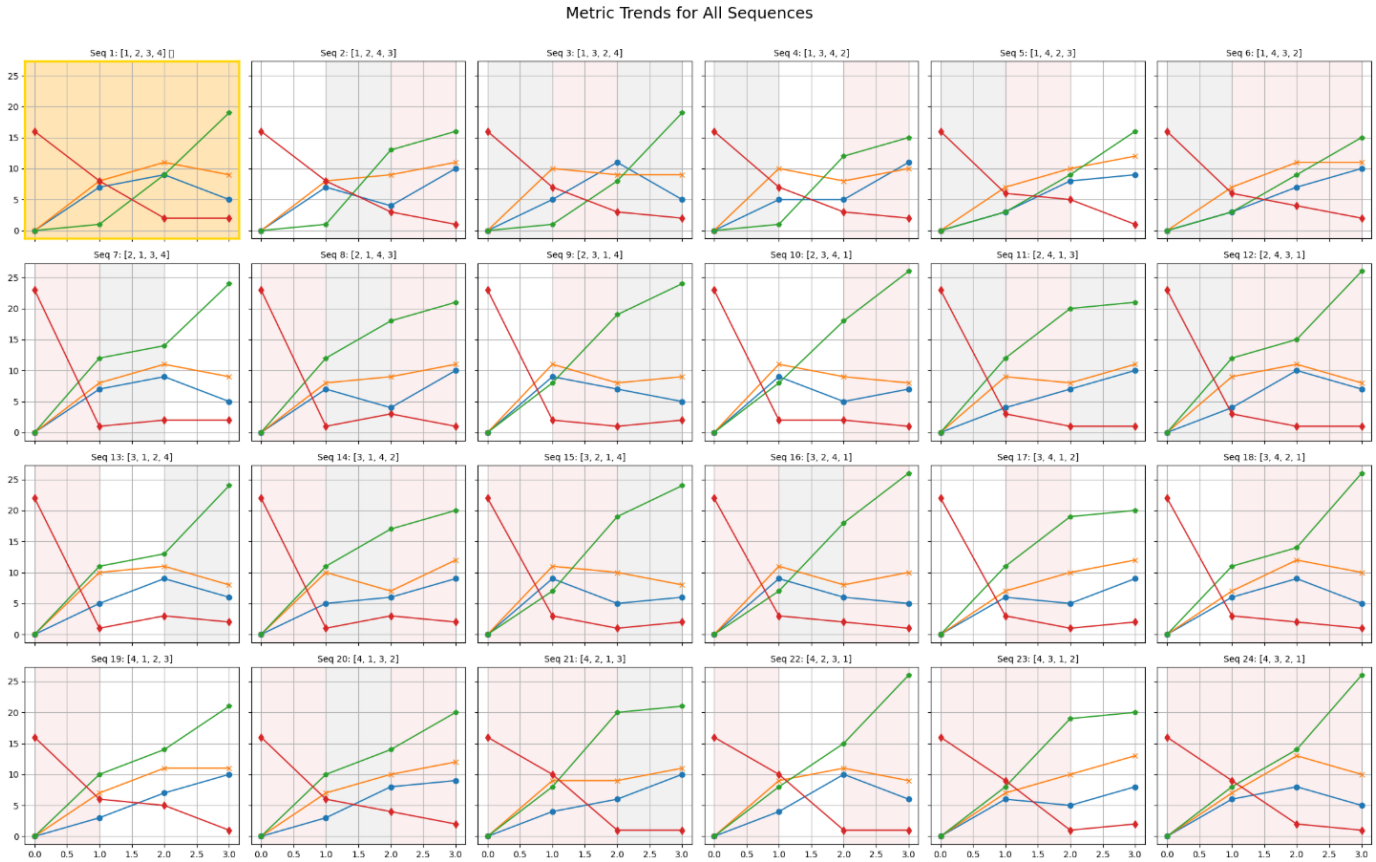


Fig. 15: Analysis for all sequences for Scenario 2. Each graph shows four different metrics: red represents new (surprising) hypotheses; green represents decay; blue represents confirmed hypotheses; orange represents falsified hypotheses. The shading of every graph shows the sequence consistency: white background means the images are in expected consecutive order (e.g. image 1 followed by 2), gray means a step is missed (e.g. image 2 followed by image 4), read means wrong sequence (e.g. image 4 followed by image 3). The graph highlighted in orange is the graph with the highest amount of confirmed hypotheses.