Cutting Events: Towards Autonomous Plan Adaption by Robotic Agents through Image-Schematic Event Segmentation

Kaviya Dhanabalachandran*
Institute of Artificial Intelligence
Bremen University, Germany
kaviya@uni-bremen.de

Vanessa Hassouna
Institute of Artificial Intelligence
Bremen University, Germany
hassouna@cs.uni-bremen.de

Maria M. Hedblom†
Jönköping Artificial Intelligence
Laboratory
Jönköping University, Sweden
maria.hedblom@ju.se

Michaela Kümpel
Institute of Artificial Intelligence
Bremen University, Germany
michaela.kuempel@uni-bremen.de

Nils Leusmann
Institute of Artificial Intelligence
Bremen University, Germany
leusmann@uni-bremen.de

Michael Beetz
Institute of Artificial Intelligence
Bremen University, Germany
beetz@uni-bremen.de

ABSTRACT

Autonomous robots struggle with plan adaption in uncertain and changing environments. Although modern robots can make popcorn and pancakes, they are incapable of performing such tasks in unknown settings and unable to adapt action plans if ingredients or tools are missing. Humans are continuously aware of their surroundings. For robotic agents, real-time state updating is time-consuming and other methods for failure handling are required. Taking inspiration from human cognition, we propose a plan adaption method based on event segmentation of the image-schematic states of subtasks within action descriptors. For this, we reuse action plans of the robotic architecture CRAM and ontologically model the involved objects and image-schematic states of the action descriptor ‘cutting.’ Our evaluation uses a robot simulation of the task of ‘cutting bread’ and demonstrates that the system can reason about possible solutions to unexpected failures regarding tool use.

CCS CONCEPTS

• **Computer systems organization** → *Robotic autonomy*.

KEYWORDS

situational assessment, cognitive robotics, plan adaption, autonomous agents, event segmentation, image schemas

ACM Reference Format:

Kaviya Dhanabalachandran, Vanessa Hassouna, Maria M. Hedblom, Michaela Kümpel, Nils Leusmann, and Michael Beetz. 2021. Cutting Events: Towards Autonomous Plan Adaption by Robotic Agents through Image-Schematic Event Segmentation. In *Proceedings of the 11th Knowledge Capture Conference (K-CAP '21), December 2–3, 2021, Virtual Event, USA*. ACM, New York, NY, USA, 8 pages. <https://doi.org/10.1145/3460210.3493585>

* All authors have equal contribution.

† Main corresponding author.

This work is licensed under a Creative Commons Attribution International 4.0 License.

K-CAP '21, December 2–3, 2021, Virtual Event, USA.

© 2021 Copyright is held by the owner/author(s).

ACM ISBN 978-1-4503-8457-5/21/12.

<https://doi.org/10.1145/3460210.3493585>

1 INTRODUCTION

Robotics research and engineering have seen great advancement in the past decades. Modern robotic agents are experts at performing well-defined activities in static environments (e.g. [15, 21, 28]) and research on vaguely-defined actions in static environments has also seen improvement in the past years [17]. However, when asked to perform even the simplest of tasks in unfamiliar and fast-changing environments, the performance of artificial agents is, at best, inefficient and, at worst, dangerous. As the robotic industry is getting closer to taking part in people’s homes as household appliances or companions, this problem needs to be properly addressed.

From an engineering point of view, the current limitation manifests as two distinct problems. On the one hand, robotic agents lack a sophisticated ability to reason about the involved objects, agents and environment in actions. They are (generally) unable to adapt to new situations in a way that respects the rules of the environment, changes in object states or takes the inconsistency of human behaviour into account. For instance, imagine a robotic agent in a kitchen scenario which is given instructions to bring the milk. It needs to have information about the most likely storage location of milk: the refrigerator. If there is no milk in the refrigerator, the robot should know to check for UHT milk in one of the cupboards as a substitute. Additionally, it needs to have the commonsense knowledge of the functional relationship that as milk is a liquid, it needs to be transported inside a solid container. Even more importantly, if a robot encounters a human during task execution, it must stop performing any actions to avoid accidents while also being able to continue its action execution when the human is no longer in the way.

This displays the second problem, namely, that the agent needs to have an ability for situational awareness in the form of real-time updates of the states of the world. However, real-time state updates of the whole environment are both time-consuming and cost-inefficient, making them unfeasible in the current state of robotic research. Instead, new methods to detect changes in object states such as a leaking container, or humans blocking the agent’s path, needs to be developed.

In [14], these issues were suggested to be approached through the integration of formal representations of conceptual building

blocks for object states that are commonly used for reasoning, referred to as image schemas [12, 16]. Following this research agenda, we develop this idea further by specifically looking at the role situational awareness plays when a robot fails to perform a particular action (i.e. is unable to reach its goal state).

The framework we use is built on a large knowledge-based system for cognitive robotics [3]. It combines the Cognitive Robot Abstract Machine (CRAM) [6] with the Socio-physical Model of Activities (SOMA) [7], both part of the Open EASE project¹, together with the Failure Interpretation Ontology [8]. Our method to enrich this system with plan adaptation in unexpected situations takes inspiration from event segmentation theory based on image-schematic states [13, 34]. In our work, this takes the form of dividing robot action descriptors into smaller subtasks that are described in terms of functional relationships and image-schematic states of the involved objects. Our thesis states, that if the state of a particular subgoal is different from the expected state based on the plan descriptor, then the agent is required to re-calibrate and perform a plan adaption to ‘intelligently’ choose its next course of action in order to reach the requested goal state. While failures are detected using CRAM, the connected ontologies allow for reasoning and plan adaption.

As household activities are plentiful and of increasingly complex character, to demonstrate the ideas in this paper, we will reduce the problem space by looking at the action descriptor for *cutting food*. To evaluate the success of our modelling, we use CRAM’s robot simulation to demonstrate how a robot would be able to successfully perform cutting actions even when faced with unexpected tool failures.

The main contributions of this work are: first, we ontologically model an image-schematic event segmentation for the cutting action. Second, we include a continuous situation assessment through a task-specific failure interpretation. Third, we simulate the reasoning capabilities of a virtual robot.

2 FOUNDATION AND SYSTEM OVERVIEW

2.1 Autonomous Robots in Uncertain Environments

In controlled environments, modern robots can do a wide variety of household tasks, such as serving breakfast [23], sweeping the floor [33], preparing pancakes [5] or setting and cleaning the table [18].

Many of these approaches rely on actions in state transition systems, in which different actions are treated as atomic transitions between states [9], an example is the planning domain definition language (PDDL) [19]. Within such state-based systems, situational awareness is usually restricted to updates between action performance, as a means to update the internal states of the world. For predictable and unchanging environments, this is highly efficient, as actions can be executed without considerations to what might go wrong or what might take place outside the scope of that particular action. In these systems, task execution is usually done through computing collision-free paths rather than motions that achieve certain effects [22].

However, in uncertain environments where variables and states change independently of the actions performed by a robot, other methods for situational awareness needs to be employed. One such method is to divide the actions into subtasks. It has been shown that uncertainty in task specification can be reduced by a task decomposition into interchangeable, generalised subtasks [30]. Moreover, an action decomposition into primitive tasks enables robots to reliably and flexibly make decisions [4]. Given an action decomposition into primitive tasks, primitive tasks can be coupled into various actions for successful motion planning and action execution.

This is of particular importance to interpret the reasoning for failures in automated motion planning and execution [8]. In the next section, the inspiration for a cognitively-inspired approach to deal with this problem is presented.

2.2 Image-Schematic Event Segmentation

Unlike artificial agents, humans are well-adapted to deal with unfamiliar events set in uncertain environments. Repeated exposure to particular situations early teaches humans how to distinguish different events from one another and to ‘break apart’ action segments based on pauses or conceptually different structures [2]. Based on a combination of analogical reasoning to similar situations and by performing commonsense reasoning, humans can quickly adapt and satisfactorily achieve goals even with the most under-specified instructions. As one event floats seamlessly into another, event segmentation aims to explain how humans make conceptual cuts in the flow of time [34].

The cuts can be based on state-altering activities within the event, such as particular movement patterns or alterations in the relational states of the involved objects. For instance, the `SOURCE_PATH_GOAL`² (SPG) movement of a knife moving towards a loaf of bread, or the `CONTACT`-relationship between the knife and the bread just before it is cut into parts (the dynamic `SPLITTING` state of the `PART-WHOLE` image schema). These kinds of functional relationships are (by some) called image schemas [16] that have been formalised in different formats (e.g. [12, 20]). In short, image schemas are spatiotemporal relationships between objects that are learned from sensorimotor experiences. In addition to those mentioned, common examples include `SUPPORT`, `VERTICALITY` and `CONTAINMENT`, where related members of that particular functionality are embedded within that concept (e.g. going `IN`). In previous research of looking at event segmentation for formal systems, the image-schematic states have been argued to map onto the conceptual cuts of events [13].

[14] introduce the idea of using image-schematic states for action selection by arguing how it would enrich robot action descriptors with semantic components. One main benefit that was argued, was that as image schemas are both intrinsically meaningful and exist in a finite number, using them in robotic planning offers intelligent reasoning methods to find alternative plans in unexpected state changes.

In the upcoming sections, we introduce the framework for event representation within the scope of this paper and demonstrate how

¹<http://www.openease.org>, Open EASE is an open-source resource platform for research in cognitive robotics.

²Following convention, image-schematic states and transitions are written in capital uppercase letters.

```

(perform
  (an action
    (type cutting)
    (?object (an object (type object-to-slice) (pose ...) ...))
    (?object (an object (type slicing-object) (pose ...) ...))
    (?arm (a body_part (type arm-to-slice) (pose slice-position) ...))
    (?arm (a body_part (type arm-to-hold) (pose (on object-to-slice)) ...))
    (goal (cut object-to slice)))
  )

```

Figure 1: CRAM representation of cutting

```

Cutting  $\sqsubseteq$  PhysicalTask
PhysicalTask  $\sqsubseteq$  Task
Task  $\sqsubseteq$  ( $\forall$ classifies.Action)
Action  $\sqsubseteq$  ( $\exists$ hasParticipant.PhysicalAgent
   $\sqcap$   $\exists$ hasParticipant.PhysicalObject)

```

Figure 2: SOMA representation of cutting.

subtasks are inspired by the notion of image-schematic cuts in the event.

2.3 Action Descriptors in CRAM

Robots typically use cognitive architectures for action planning and execution. This work employs the Cognitive Robot Abstract Machine (CRAM) as the cognitive architecture to execute robot actions in human environments [6]. CRAM combines symbolic descriptions of actions with numerical, sub-symbolic data from its perception system and has been used to advance robotic performance in everyday household activities (e.g. [5, 18]). The high-level plan executive CRAM uses KnowRob [3], a knowledge representation and reasoning system, to perform commonsense reasoning. KnowRob, in turn, employs different domain ontologies to reason about and interpret robot actions, object affordances, environments and situations.

As a demonstration, Figure 1 shows the CRAM action description of *cutting*. The description commands the robot to perform the action of type *cutting* on some previously perceived object of type *object-to-slice* by using a *slicing-object* with the two body parts *arm-to-slice* and *arm-to-hold*. The terminal state goal of the action is to have cut the *object-to-slice* into two separate objects.

CRAM uses three different types of designators: *object-*, *action* and *location designators*. The *object designator* is a symbolic descriptor for the representation of an object, where the detailed information about the object is later obtained from the perception system within CRAM or derived from the knowledge processing system KnowRob. The *action designators* are symbolic action descriptions for which properties and rules can be specified for the actions. The third designator is the *location designator*, which describes poses of the candidates (e.g. objects, robot, kitchen) and verifies if the poses of objects/robots enable action execution (e.g. spatial information of ‘collision-free’ locations to stand in order to successfully execute the actions). Within the designators, identification keys are used to distinguish between different objects. For example, if the system detects a knife, it will produce an object designator with the corresponding key-value pair (type, name and location).

Needed information for task execution is extracted from the object designator and missing information is inferred from the reasoning engine KnowRob. For example, information on the angle of the object in relation to the robot and how much force is needed for specific tasks can be inferred. In the example action of *cutting*, the knife would be assigned the object designator *object-to-cut* as a unique key-value pair.

Through KnowRob, the robot can update its CRAM plan according to the current state representation of the world.

2.4 Segmented Event Information in SOMA

CRAM’s action descriptors include high-level action plans. However, everyday activities like cooking, setting the table or preparing a meal are difficult tasks for a robot. To enrich the system with more detailed information, the Socio-physical Model of Activities (SOMA) was designed to provide robots with environment and activity knowledge so that they could achieve a larger variety of goals in any given situation [7]. Written in the Web Ontology Language (OWL) [25], SOMA is an upper ontology for the representation of activities from the perspective of household robots. It is based on the DOLCE+DnS Ultralite (DUL) foundational framework and its definitions of descriptions and situations [24]. As such, it provides a more detailed structure for the representation of activities and their subtasks, participating agents and objects as well as their properties and roles in the given actions. The SOMA ontology can offer valuable information about where objects can be found, which action descriptors are best for particular actions and which action can be executed, given a particular set of objects. This information is derived from commonsense rules for activities, which are based on a combination of object properties, object affordances³ and the functional relationships between them.

In SOMA, actions are defined as “*events performed by agents*.” Such actions follow a CRAM plan that involves at least one task. For example, the action of grasping a knife would include the tasks of *approaching* the knife with the gripper, *grasping* the knife with an appropriate enclosing movement, and *lifting* the knife of its supporting surface. Each of these tasks can be divided into detailed subtasks of increasingly atomic details until they reach the image-schematic level.

Used as the example domain in this paper, the task of *cutting* is classified as a physical task in SOMA (see Figure 2). As an action, it is further specified to have two participants: one physical agent and one physical object. For a *cutting* task, an agent needs two objects of specific dispositions and/or affordances. Namely, 1) a *cutter* such as a sharp object like a knife, and 2) a *cuttable* object that can be divided into pieces, such as a loaf of bread. To specify this, we extended the ontology to represent this particular domain information (see Section 3.1).

One noteworthy comment regarding this task definition is that a robotic agent could use another object to perform the action as long as they share the same dispositions. SOMA does not specify that it is a knife that has to be used for the cutting nor that it is a bread that is being cut. Through the KnowRob reasoner, it could also infer

³Affordances in the Gibsonian sense [10].

that a saw could do the task on a vegetable or a wooden log as long as these objects have been defined with the same dispositions. This allows the robot to fulfil CRAM action goals even in changing environments, by adapting the actions based on available options.

2.5 Action Failure Interpretation

The combination of motion and task planning in CRAM with detailed information from SOMA can help robots to find alternative plans to their actions. Another resource KnowRob includes, is an ontology of common failures that helps to diagnose unsuccessful actions and to find alternative paths to recover from them [1, 8]. The failure interpretation ontology in [8], covers a wide variety of possible failures that often take place in autonomous planning and execution. The failure ontology includes both the failure classification as well as the reasoning for why the action has failed. Some examples are failures based on location and geometry, direct hardware failures of the robot, as well as failures dependent on faulty reasoning skills and shortcomings in cognitive ability.

For instance, if a robot is not able to grasp an object because it cannot reach it, the ontology diagnoses it as a *reachability failure*, which is a subclass of a *capability failure*. The *capability failure* is, in turn, a subclass of an *inception failure* and classifies the grasping pose of the robot (see definition in Figure 3). This hierarchical classification defines the failures. In case the robot was unable to reach the object with the previously applied side grasp, the failure handling would propose to re-adapt its action plan by trying another grasping pose, such as a top grasp. By having access to failure knowledge, robotic agents can reason about their failures to reach the desired goal states of subtasks in the action plan and can re-calibrate their actions to succeed with the plan.

We reuse this failure interpretation ontology for a repeated state assessment and plan adaption and extend it with failure termination rules. In the next section, we present our case study by doing a formal event analysis of the action description of *cutting* and connecting it to the image-schematic states of the event.

Reachability Failure $\sqsubseteq$ *Capability Failure*
Capability Failure $\sqsubseteq$ *Inception Failure*
Capability Failure $\sqsubseteq \exists \text{classifies.GrASPingPose}$
Inception Failure $\sqsubseteq \exists \text{classifies.Task}$

Figure 3: Reachability Failure from the failure interpretation ontology.

3 EVENT REPRESENTATION OF CUTTING

In this work, we use the action of *cutting* as an example for task segmentation with continuous, autonomous state assessment. Cutting is a simple, yet dynamic, action that requires different tools and movements based on what is cut, e.g. bread is sawed, meat sliced and cucumbers chopped. If an agent is given the task to ‘*cut the cucumber*’, situation assessment is crucial for successful task execution. There might be different knives available to choose from, and should the most optimal tool be missing, the robot should be able to reason what tool makes for an appropriate substitute.

3.1 Ontological Modelling of Objects Involved in Cutting Bread

The created ontology is an extension of the SOMA ontology [7], modelled in Description Logic. To demonstrate the cutting action, we have modelled the objects involved in cutting (bread, knives and cutting board) in relation to their dispositional properties (the full ontology is published on Github⁴). The dispositional property of an object varies based on the requirements of the environment. For example, a *CuttingBoard* can be used to provide *SUPPORT* to any object which undergoes cutting, with a disposition type *Deposition*. However, the same *CuttingBoard* can also be used as a cover for a *CONTAINER*, as in the case in which there is no lid available for a pot, as both have the disposition type of *Coverage* [29]. As mentioned in previous sections, the purpose to model the dispositions of objects is to enable the robot to infer the potential tasks afforded by the objects and what actions are possible given a set of objects, similarly to how a human can reason about the possibilities with particular objects.

Following Turvey’s definition of dispositions [31], for the disposition type *Cuttability* the objects of type *CuttingTool* act as *bearer* and the objects of type *Food* act as *trigger* for the task of cutting. In our kitchen environment, the most essential objects include the knives that play the role of the *Tool* that affords cutting and has the disposition type *Splitting*, which alters the goal state of the object they cut. The bread plays the role of *ShapedObject* as it changes shape while being cut, given through the disposition of type *Cuttability*. In our scenario, the cutting board takes on the disposition of type *Deposition* and plays the role of type *Deposit* as it supports our cutting objects.

The axioms in Figure 4 are a small subset of our TBox ontology. The first axiom states that Bread is a *Food* and that it possesses the dispositions that affords the task of type *Cutting*, which in turn trigger an object of type *CuttingTool*.

From the second axiom representing some features of *Bread Knife*, the same task of cutting is triggered, but with the role *ShapedObject*.

Important to note is that the roles that the objects can take vary in relation to the task progression. For instance, the knife assumes a role of *HeldObject* when the subtask of *AssumingArmPose* is executed, where the knife is passive and is held by the gripper

⁴https://github.com/kaviyachandran/situational_awareness_example/blob/master/owl/situational_awareness_example.owl

Bread $\sqsubseteq$ *Food* $\wedge \exists \text{hasDisposition.Cuttability} \wedge \exists \text{hasRole.ShapedObject}$
Cuttability $\sqsubseteq (\exists \text{affordsTask.Cutting} \wedge \exists \text{affordsTrigger.Tool})$
Tool $\sqsubseteq \exists \text{classifies.CuttingTool}$

BreadKnife $\sqsubseteq$ *Knife* $\wedge \exists \text{hasDisposition.Splitting} \wedge \exists \text{hasRole.Tool}$
Splitting $\sqsubseteq (\exists \text{affordsTask.Cutting} \wedge \exists \text{affordsTrigger.ShapedObject})$
ShapedObject $\sqsubseteq \exists \text{classifies.Food}$

Figure 4: Representation of Bread and Bread Knife.

Figure 5: Task segmentation with SOMA's image-schematic state representation of the subtask *Lower cutter object*.

of the robot. Later, the same knife will take on the active role of *CuttingTool* when performing the subtask of *PushingDown* when the knife pushes through (ie. cutting) the bread while moving towards the cutting board. The distinctions between these subtasks are introduced in the next section.

3.2 Cutting as an Event in CRAM

The task to *cut the bread* can be segmented as depicted in Figure 5. For this task, the robot needs to use both hands; one to hold the bread and one to hold a knife that cuts the bread. The diagram shows the task segmentation for the hand holding the knife. First, this hand needs to *grasp* an object of type *CuttingTool* (henceforth called *knife*), then *cut* the “cuttable” object of type *ShapedObject* (henceforth called *bread*) until the knife reaches the cutting board, and finally *place* the knife back on the table. The cutting action is (row two) segmented into the more granular tasks to *approach* the bread with the knife, to *lower* the knife ‘through’ the bread and then to *lift* the knife.

In Figure 5, the continuous state assessment and plan adaption is visualised with example failures like “*Grasp not successful?*” that will be explained in the following sections. If the robot was unable to grasp the object, a reachability failure, as described in Figure 3, would be diagnosed, leading to a plan adaption in grasping pose and starting a new grasping task.

3.3 Analysing the Image-Schematic States of Cutting

All events can be segmented into a series of transitions, where each transition comprises an initial scene, an event execution and resulting in a (usually, but not necessarily different) terminal scene. Each transition consists of an image-schematic state of the involved objects, an image-schematic transformation and an updated image-schematic state. For instance, the transition of the task of approaching, in which the knife moves towards the bread, can

be image-schematically defined as a sequence of states such as *CONTACT* (Hand, Knife) followed by a *DOWN-facing SOURCE_PATH_GOAL*(Knife) until the state of *CONTACT* (Knife, Bread) is reached. There the next transition starts (defined in SOMA as *PushingDown*), cutting the bread: *CONTACT* (Knife, Bread) + *PATH_GOAL* (Knife, CuttingBoard) that leads to the end state *CONTACT* (Knife, CuttingBoard) + *SPLITTING*(Bread, Slice). In this fashion, the entire event can be broken into the corresponding spatiotemporal states that are each defined based on features that are part of CRAM’s object, action and location descriptors. In Table 1, an overview of some important action descriptors are presented in relation to the SOMA task and the involved image schema transformations.

More specifically, the definition of a transition depicts how the scene should pan out after a specific task has been executed. The concept of *Transition* defined in SOMA is used to represent situations that occur before and after the execution of an action. A *situation* is considered as a description of a *state*. In a scenario with no failures, the situations manifest as states in the physical world. Any *scene* which has an *is-a* relation with a *situation* is used to denote the pre- and post-conditions of the executed action.

The initial scene before the execution of an action defines the roles that must be part of the state for successful task execution. During the execution phase, the *state* that manifests as a particular *scene*, must have physical objects as participants in the state taking up the roles of the type defined in the corresponding *Scene*. The terminal scene of a *transition* represents the outcome of an action. The roles that objects assume in the initial and terminal scene depict how the action has altered the state of the objects.

In the system, these object descriptors would define an individual *Bread_0* of type *Bread* that acts in a role of type *DepositedObject* prior to the action execution as it has been placed on the cutting board (or other supporting surface). Then in order to slice the bread, a task of type *PushingDown* will be executed and this results in splitting the bread to ‘obtain’ a bread slice. Once the action is executed successfully, the role *ShapedObject* will be assigned to the *Bread_0*, and the obtained bread slice act as role type *CreatedObject*. Figure 5 illustrates the facts that are part of a transition individual when the action *PushingDown* is performed. Each action descriptor in CRAM plan can be connected to the transition individual. The transition individual dictates the pre- and post-conditions for a task to be performed with its associated initial and terminal scene instances.

CRAM	SOMA	Image Schemas
Grasp	Grasping	LINK + CONTAINMENT
Approach	AssumingArmPose	SPG
Lower	PushingDown	DOWN + SPG
Lift	Lifting	UP + SPG
Hold	Holding	CONTAINMENT + LINK
Release	Releasing	¬CONTACT
Place	PuttingDown	SUPPORT

Table 1: CRAM action descriptors with equivalent SOMA tasks and the closest image-schematic states

Moreover, the scene individuals have properties that illustrate the image schemas they satisfy. These properties define the functional and spatial relationships that hold among the objects in the state. Considering the transformed scene obtained by executing the PushingDown task, it satisfies two image schema concepts, namely ContactTheory and SupportTheory. ContactTheory materialises when the knife comes into contact with the bread (and later the cutting board) and the SupportTheory is realised as the bread is supported by the cutting board.

3.4 Reasoning About Event Failures

As discussed in Section 3.3, the representation of the events is segmented into multiple transitions. The modelling in SOMA provides a convenient way to check for failures by confirming whether the outcome of an action, a *State* has satisfied the corresponding *Scene* individual. In our work, a *State* satisfies the *Scene* description, if it includes objects that assume desired roles as represented in the *Scene*. It is essentially matching the object states in the *Scene* with the *State*. The first order formula given in Equation 1, represents the rule to check if the physical state is a manifestation of the scene description. The unary predicates assert individuals of their respective types. The *satisfies* predicate defines the relationship between the scene and a state. This relation holds when all the roles that are part of the *Scene* individual s_0 can classify the objects that are part of the *State* st_0 .

$$\begin{aligned} & \forall st_0, s_0 : \text{PhysicalState}(st_0) \wedge \text{Scene}(s_0) \wedge \text{satisfies}(st_0, s_0) \\ \rightarrow & \forall r_0 \exists o_0 : \text{Role}(r_0) \wedge \text{Object}(o_0) \wedge \text{usesRole}(s_0, r_0) \wedge \\ & \text{hasParticipant}(st_0, o_0) \wedge \text{classifies}(r_0, o_0) \end{aligned} \quad (1)$$

The reasoning is carried out as a part of the KnowRob framework, with a closed-world assumption. This means that the facts that cannot be deduced from the knowledge base are considered false. Therefore, the extended ontologies were loaded into the knowledge base.

Let us consider the outcome of the PushingDown task in our cutting scenario. The scene individual contains the roles *ShapeObject* and *CreatedObject*. When the task is executed successfully, we expect to infer objects assuming these roles in the *State*. Any deviation will mean that the task execution failed, as the desirable goal has not been reached.

In case of a failure, we limit ourselves to reason about those that are caused by incorrect tool use. As introduced, we use the failure ontology⁵ to classify the failure types and respond with a suitable recovery strategy. For example, if the bread was not cut, this could be due to several different reasons. One such cause could be that the knife slips or collides with the cutting board rather than the bread. In this case, in the terminal state, the role of the bread is of type *Unaltered* as the bread has not undergone any change in shape. We classify the last action leading to this failure as of type *FinishedFailure* as we do not have the desired outcome of the action.

To handle the recovery phase for a detected failure, a reactive behaviour is considered as a primary response. To recover from the failure, a recovery strategy of type RepeatLastAction is performed. If the failure persists through a second trial, KnowRob

infers that the failure is not of type *FinishedFailure* but might be of type *CapabilityFailure* and thus searches for the most suitable substitute object to perform the task with. This is done by finding another object that can act as a *trigger* to realise the disposition possessed by bread for the cutting task. In Equation 2 we represent the disposition property of bread. This disposition is further used to deduce the role assumed by the trigger object. In our case, with the bread being the trigger, the bearer role needs to be taken by an object of type CuttingTool. Thus, the next action with a strategy of ReplaceResource is carried out by fetching an object that *is-a* cutting tool. If no optional failures can be inferred from KnowRob, action execution is terminated after two trials.

$$\begin{aligned} & \forall b_0 \exists t_0 : \text{Bread}(b_0) \wedge \text{Object}(t_0) \wedge \text{triggerOf}(t_0, b_0) \\ \rightarrow & \exists d_0, r_0 : \text{Disposition}(d_0) \wedge \text{Role}(r_0) \wedge \text{hasDisposition}(b_0, d_0) \wedge \\ & \text{affordsTrigger}(d_0, r_0) \wedge \text{classifies}(r_0, t_0) \wedge \text{CuttingTool}(t_0) \end{aligned} \quad (2)$$

4 EVALUATION THROUGH A ROBOT SIMULATION

Simulations offer a method to evaluate robot performance in a controlled yet realistic environment as limitless trials can be run without any risk of damaging the system. As software systems, they offer a controlled, predictive environment to evaluate the accuracy of conceptual modelling designed to successfully execute robot action plans.

To evaluate the conceptual modelling in this paper, we use the Unreal Engine⁶, as it can create a realistic virtual kitchen environment [11] with a robotic agent [27] that simulates a physical robot with its sensors and ‘physical’ constraints.

The objects within the simulation are a direct representation of objects modelled as part of the ontology. This means, for instance, that every object from our ontology with a disposition of type *Cutability* inherits properties from the cuttable class. Likewise, the objects with disposition *Splitting* inherit from the cutter class. Several properties of the simulated agent play a role in the outcome of particular actions, such as the orientation of tools and grippers and points of contact. As an example, during the execution of the cutting action, any cuttable object will be separated into two distinct pieces with different size and shape properties in relation to the orientation of the cutter. Hence, within the simulation, it is possible to fine-tune particular actions regarding the desired terminal state. For bread, this would be thin slices.

4.1 Cutting Bread Simulation

To demonstrate the failure interpretation and plan adaption capabilities of our framework, we conducted an experiment in which a simulated robot was assigned with the task to cut the bread⁷.

In the initial state, the robot is presented with a loaf of bread and two different knives in operating distance. The purpose of the scene was to evaluate how well our system can handle and overcome unexpected failures. To introduce a situation in which

⁵<https://github.com/ease-crc/failureont>

⁶<https://www.unrealengine.com/en-US/>

⁷A video of the simulation is available here: <https://www.youtube.com/watch?v=c94A1GPD-Ro>

Figure 6: The different stages of the simulation of cutting bread where each stage represents a subtask. The first row demonstrate a failure, and the readaption of the action plan is presented on the second row.

one of the subtasks would be unsuccessful, and the action fails, we defined one knife as ‘dull’ by removing the *Splitting* disposition.

As can be seen in Figure 6, the first task of the robot is to pick up a knife and try to cut the loaf of bread with it. In the first picture, the robot already picked up a knife with its left gripper. The robot then moves the knife to a pre-position above the loaf of bread. Following this, it proceeds to execute the subtask of *PushingDown*, where the knife comes into contact with the loaf of bread. As the knife lacks the disposition of *Splitting*, and the requirement for cutting is the contact relationship between a cuttable object (ie. the bread) and an object with cuttability, a failure occurred and the bread is not split. Through the perception system, the robot identifies that there was no slice produced and following its failure handling protocol described in Section 3.4, it proceeds with the task *RepeatLastAction*. Once again, the robot executes the last subtask, which was *PushingDown* (Figure 6.4). However, the failure remains as the properties of the involved objects have not changed. Therefore, the robot adapts its plan to *ReplaceResource*. In this situation, this corresponds to placing the knife on the table and gripping an alternative tool: the available second knife. After the *ReplaceResource* subtask, the action plan is reassessed, and the robot continues with the original action plan for cutting by executing the subtask *PushingDown* (Figure 6.6). As this knife has the cuttability disposition, the bread is cut and a slice produced. This means, that the simulation could successfully execute the plan adaption presented within the system to reiterate its subtasks and reason about the available tools.

5 DISCUSSION AND RELATED WORK

Our work differs from similar work on automatic planning for household tasks (e.g. [18, 23, 33]) by aiming to identify and formally specify the natural cuts in event and action segments using image schema theory. This approach is suggested to be introduced into the robotic architecture CRAM and its associated knowledge

representation system (SOMA, KnowRob and the failure ontology). However, at this stage, the image-schematic segments are embedded within the formalisation of the robotic action descriptors and not individually represented as in [12, 20]. Similar ideas have been developed (e.g. [29]) and there is a large body of work in robotics devoted to learning affordances (see overviews in [26, 32]), but, to our knowledge, these tend to focus more on object manipulation rather than plan adaption. To extend the usefulness of this idea, these patterns need better represent the states of the subtasks in particular action descriptors.

The hypothesis is that this would lead to robots intelligently adapting their plans when faced with failures by identifying the image-schematic dispositions of objects and the image-schematic transformations of actions. In our simulation, we demonstrated how the system successfully can adapt the subtasks of an action descriptor by using other tools with the same dispositions. To make this more general, it is possible to use learned models to infer parameters. These parameters would be things like grasp poses, object and robot positions as well as human-learned action executions.

One method would be to learn robot-readable models from human activities in the format of narrative-enabled episodic memories (NEEMs) [11]. Such episodic memories can be used to learn and understand the most probable failure situations for a formulation of a suitable recovery strategy and expand the failure ontology. Combining this with the approach in this paper might make it possible to automatically learn alternative plans without the same level of handcrafted modelling.

6 CONCLUSION AND FUTURE WORK

Improving autonomous agents’ behaviour has been an important area of research using both knowledge representation and subsymbolic methods. In this paper, we opted for an ontological approach to divide robotic actions into subtasks of increasingly detailed state descriptions, ultimately ending at the image-schematic level. Our main

hypothesis is that as long as the pre- and post- image-schematic states of executed subtasks are reached, the action can proceed. If not, plan adaption is required.

To demonstrate this idea, we presented a case study on the event *cutting bread* and ontologically modelled the objects in SOMA and used the action descriptor in CRAM. We also introduced appropriate failure management using the CRAM reasoner KnowRob, which has access to an ontology of common failures. In our evaluation using a robot simulation, we demonstrate that the system can reason about alternative paths to reach the terminal state of an action descriptor even when presented with unexpected states.

Future work includes extending the knowledge represented within this system with a particular focus on coring out the image-schematic states of subtasks. Further, to evaluate the method's generalisability, we intend to use the learned model for different cutting tasks in various settings, such as a retail environment to open and unpack a box.

Acknowledgments

The authors would like to thank Mihai Pomarlan for valuable feedback during the development of this work. The research reported in this paper has been partially supported by the Federal Ministry for Economic Affairs and Energy BMWi within the Knowledge4Retail project, subproject semantic Digital Twin 01MK20001M (<https://knowledge4retail.org>) as well as the German Research Foundation DFG, as part of Collaborative Research Center (Sonderforschungsbereich) 1320 "EASE - Everyday Activity Science and Engineering", University of Bremen (<http://www.ease-crc.org/>).

REFERENCES

- [1] Aliakbar Akbari, Fabien Lagriffoul, and Jan Rosell. Combined heuristic task and motion planning for bi-manual robots. *Autonomous robots*, 43(6):1575–1590, 2019.
- [2] Dare A. Baldwin, Jodie A. Baird, Megan M. Saylor, and M. Angela Clark. Infants parse dynamic action. *Child development*, 72(3):708–717, 2001.
- [3] Michael Beetz, Daniel Beßler, Andrei Haidu, Mihai Pomarlan, Asil Kaan Bozcuoglu, and Georg Bartels. Knowrob 2.0 – a 2nd generation knowledge processing framework for cognition-enabled robotic agents. In *Int. Conf. on Robotics and Automation (ICRA)*, Brisbane, Australia, 2018.
- [4] Michael Beetz, Dominik Jain, Lorenz Mosenlechner, Moritz Tenorth, Lars Kunze, Nico Blodow, and Dejan Pangercic. Cognition-enabled autonomous robot control for the realization of home chore task intelligence. *Proc. of the IEEE*, 100(8):2454–2471, 2012.
- [5] Michael Beetz, Ulrich Klank, Ingo Kresse, Alexis Maldonado, Lorenz Mosenlechner, Dejan Pangercic, Thomas Rühr, and Moritz Tenorth. Robotic roommates making pancakes. In *11th IEEE-RAS Int. Conf. on Humanoid Robots*, pages 529–536. IEEE, 2011.
- [6] Michael Beetz, Lorenz Mosenlechner, and Moritz Tenorth. CRAM—a cognitive robot abstract machine for everyday manipulation in human environments. In *2010 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1012–1017. IEEE, 2010.
- [7] Daniel Beßler, Robert Porzel, Mihai Pomarlan, Abhijit Vyas, Sebastian Höffner, Michael Beetz, Rainer Malaka, and John A. Bateman. Foundations of the Socio-physical Model of Activities (SOMA) for Autonomous Robotic Agents. In *Proc. of the 12th Int. Conf. Formal Ontology and Information Systems FOIS 2021, Bozen-Bolzano, Italy, September 13–16, 2021*, Frontiers in Artificial Intelligence and Applications. IOS Press, 2021.
- [8] Mohammed Diab, Mihai Pomarlan, Daniel Beßler, Aliakbar Akbari, Jan Rosell, John A. Bateman, and Michael Beetz. An ontology for failure interpretation in automated planning and execution. In *Iberian Robotics conference*, pages 381–390. Springer, 2019.
- [9] Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning: theory and practice*. Elsevier, 2004.
- [10] James J. Gibson. The theory of affordances. In Robert Shaw and John Bransford, editors, *Perceiving, Acting, and Knowing: Toward an Ecological Psychology*, pages 67–82. NJ: Lawrence Erlbaum, Hillsdale, 1977.
- [11] Andrei Haidu and Michael Beetz. Automated acquisition of structured, semantic models of manipulation activities from human VR demonstration. *CoRR*, abs/2011.13689, 2020.
- [12] Maria M. Hedblom. *Image Schemas and Concept Invention: Cognitive, Logical, and Linguistic Investigations*. Cognitive Technologies. Springer Computer Science, 2020.
- [13] Maria M. Hedblom, Oliver Kutz, Rafael Peñaloza, and Giancarlo Guizzardi. Image schema combinations and complex events. *KI-Künstliche Intelligenz*, 33(3):279–291, 2019.
- [14] Maria M. Hedblom, Mihai Pomarlan, Robert Porzel, Rainer Malaka, and Michael Beetz. Dynamic Action Selection Using Image Schema-based Reasoning for Robots. In *Proc. of the Joint Ontology Workshops 2021*, CEUR-WS, Bolzano, Italy, 2021.
- [15] AT Ismail, Alaa Sheta, and Mohammed Al-Weshah. A mobile robot path planning using genetic algorithm in static environment. *Journal of Computer Science*, 4(4):341–344, 2008.
- [16] Mark Johnson. *The Body in the Mind: The Bodily Basis of Meaning, Imagination, and Reason*. The University of Chicago Press, Chicago and London, 1987.
- [17] Gayane Kazhoyan and Michael Beetz. Executing underspecified actions in real world based on online projection. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2019.
- [18] Gayane Kazhoyan, Simon Stelter, Franklin Kenghagh Kenfack, Sebastian Koralewski, and Michael Beetz. The robot household marathon experiment. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2021.
- [19] Z. Kootbally, C. Schlenoff, C. Lawler, T. Kramer, and S.K. Gupta. Towards robust assembly with knowledge representation for the planning domain definition language (PDDL). *Robot. Comput.-Integr. Manuf.*, 33(C):42–55, June 2015.
- [20] Werner Kuhn. An image-schematic account of spatial categories. In *Int. Conf. on Spatial Information Theory*, pages 152–168. Springer, 2007.
- [21] Lars Kunze, Moritz Tenorth, and Michael Beetz. Putting people's common sense into knowledge bases of household robots. In *Annual Conf. on Artificial Intelligence*, pages 151–159. Springer, 2010.
- [22] Jean-Claude Latombe. *Robot motion planning*, volume 124. Springer Science & Business Media, 2012.
- [23] Steven James Levine and Brian Charles Williams. Concurrent plan recognition and execution for human-robot teams. In *24th Int. Conf. on Automated Planning and Scheduling*, 2014.
- [24] Claudio Masolo, Stefano Borgo, Aldo Gangemi, Nicola Guarino, and Alessandro Ultramari. Wonderweb deliverable d18-ontology library (final report). *National Research Council-Institute of Cognitive Science and Technology*, 2003.
- [25] Deborah L. McGuinness, Frank Van Harmelen, et al. OWL web ontology language overview. *W3C recommendation*, 10(10):2004, 2004.
- [26] Huaqing Min, Ronghua Luo, Jinhui Zhu, Sheng Bi, et al. Affordance research in developmental robotics: A survey. *IEEE Transactions on Cognitive and Developmental Systems*, 8(4):237–255, 2016.
- [27] Michael Neumann, Sebastian Koralewski, and Michael Beetz. Urobosim - an episodic simulation framework for prospective reasoning in robotic agents. *CoRR*, abs/2012.04442, 2020.
- [28] Matteo Parigi Polverini, Arturo Laurenzi, Enrico Mingo Hoffman, Francesco Ruscelli, and Nikos G Tsagarakis. Multi-contact heavy object pushing with a centaur-type humanoid robot: Planning and control for a real demonstrator. *IEEE Robotics and Automation Letters*, 5(2):859–866, 2020.
- [29] Mihai Pomarlan and John A. Bateman. Embodied functional relations: A formal account combining abstract logical theory with grounding in simulation. In *FOIS*, pages 155–168, 2020.
- [30] Minija Tamosiunaite, Mohamad Javad Aein, Jan Matthias Braun, Tomas Kulvicius, Irena Markievicz, Jurgita Kapociute-Dzikiene, Rita Valteryte, Andrei Haidu, Dimitrios Chrysostomou, Barry Ridge, Tomas Krilavicius, Daiva Vitkute-Adzgauskiene, Michael Beetz, Ole Madsen, Ales Ude, Norbert Krüger, and Florentin Wörgötter. Cut & recombine: reuse of robot action components based on simple language instructions. *The Int. Journal of Robotics Research*, 2019.
- [31] M. T. Turvey. Ecological foundations of cognition: Invariants of perception and action. *American Psychological Association*, 1992.
- [32] Natsuki Yamanobe, Weiwei Wan, Ixchel G Ramirez-Alpizar, Damien Petit, Tokuo Tsuji, Shuichi Akizuki, Manabu Hashimoto, Kazuyuki Nagata, and Kensuke Harada. A brief review of affordance in robotic manipulation research. *Advanced Robotics*, 31(19-20):1086–1101, 2017.
- [33] Kimitoshi Yamazaki, Ryohei Ueda, Shunichi Nozawa, Yuto Mori, Toshiaki Maki, Naotaka Hatao, Kei Okada, and Masayuki Inaba. System integration of a daily assistive robot and its application to tidying and cleaning rooms. In *2010 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1365–1371. IEEE, 2010.
- [34] Jeffrey M. Zacks and Khena M. Swallow. Event segmentation. *Current directions in psychological science*, 16(2):80–84, 2007.