# Learning Essential Task Features By Visually Analyzing Non-Expert Human Demonstrations*

Andrei Costinescu, Riddhiman Laha[1]

***Abstract*— We present a learning system using a human-in-the-loop method for representing a task of daily living from a human visual demonstration. An autonomous decision-making algorithm, embedded with a knowledge base of action definitions, proposes targeted modifications of the task's parameters that are either approved or not by a human oracle with knowledge about the task. The system's proposed modifications enable fast exploration of possible variations in the execution of the observed task. We define task descriptors that model the observed actions and show multiple examples of how these results simplify the deployment of the observed task on the robot through identified variations and flexibilities in the execution of task segments.**

## I. Introduction

With the projected increase in elderly population [1], elderly care facilities or personnel may not handle this increase or properly accord the needed care for each of their large number of patients. To address this issue, elderly care robots have been proposed. However, the implementation of everyday tasks on a manipulator is a tedious task that requires expert knowledge in robotics or *point-wise teaching* of trajectory points that are difficult to transfer to different environments. In an ideal case, this programming can be replaced by observation of the human demonstration of the task, which would allow elderly care robots to learn tasks directly from their patients. However the workspace of the human is often larger than that of the manipulator, and many of the observed motion segments result from convenience and are not true constraints on the task. It is desirable to learn only the necessary (essential) parts of the task execution to give the manipulator's planner some space for adaptations. This simplifies the execution on the manipulator, allowing a choice of alternative trajectories, and it simplifies the transfer to different environments. Transfer to different environments is a desirable feature for "new generations" of elderly care robots to use the already accumulated task knowledge of the previous robot generation.

Our goal is to explore feasible deviations from the pure imitation of the observed task. One way to do this would be to identify possible variations from alternative executions presented by humans, but this would likely require the observation of multiple different humans in the same environment. An alternative approach that we want to follow here is that the system generates possible variations from the demonstration and asks the human (the oracle) if such a change is acceptable. The goal is to limit the number and type of questions to a minimum that will allow finding necessary constraints without burdening *the oracle* too much. In the context of elderly care, this approach has the additional benefit of already familiarizing the patients with the robot's execution preference. And, since the human is the oracle dictating whether a trajectory modification is allowed or not, certain task parameter values may be pruned or restricted by the person if deemed to be an unnatural execution of a task.

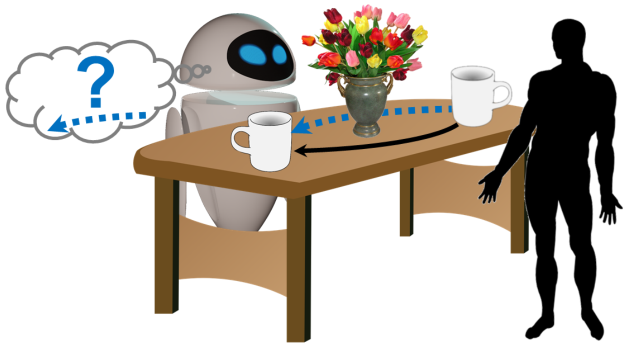

Fig. 1. In the context of a transportation task, traversed paths are sometimes unimportant. This feature must be verified by the system. If irrelevant, this relaxation of the demonstrated path is remembered to allow a robot with a different kinematic structure to complete the task in a different manner.

In this work, we focus on creating a system to visually capture the essence of a household task in a generalizable representation from one user demonstration of the task, using semantic knowledge about objects and by actively investigating allowed task freedoms not observed in the demonstration (Fig. 1).

### A. Related Work

This work is closely related to the learning by observation paradigm. Kuniyoshi et al. [2] introduced a system to represent a task plan and generalized it to a new environment. Similarly, Xiong et al. [3] designed a system based on a single video demonstration capable of performing action-imitation generalizable to different kinematic structures. Their goal was to learn to imitate actions, as opposed to learning their parametrization in particular instances of a task.

In Zoliner et al. [4], a programming by demonstration paradigm was deployed to represent tasks as sequences of action blocks, e.g., [grasp, move, ..., place]. The usage of such macro operators yields a similar task instance model to ours – however, without the exploration and exploitation of action freedoms of the represented task proposed herein.

In [5], [6], a task is represented as a finite state machine and is demonstrated using kinesthetic teaching. The goal

*This work was supported by the Lighthouse Initiative Geriatronics by StMWi Bayern (Project X, grant no. 5140951).

[1]All authors are with the School Of Computation, Information and Technology at the Technical University of Munich. {andrei.costinescu, riddhiman.laha}@tum.de

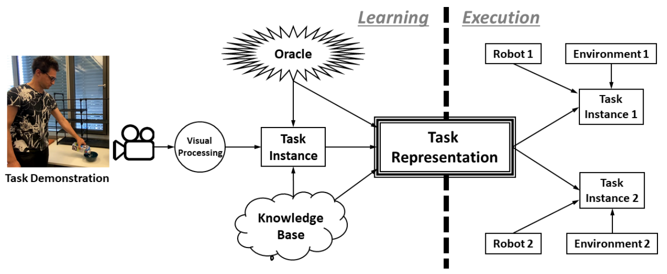

Fig. 2. The two-phase system design: one observation of a task instance is transformed in a task representation (learning), it is used to generate multiple task instances for different robots and in different environments (execution).

being, similar to ours, to enable non-expert users to teach a robot new tasks. Generalization of object concepts and path adaptation to new environments is achieved through multiple demonstrations of the same task, contrary to our approach which uses one visual demonstration and the information from the knowledge base to actively ask an oracle about the constraints and freedoms of the task.

In addition to the aforementioned literature, the works in [7]–[9] focus on geometric contact relations between objects to segment the task. Furthermore, the results in [10]–[13] aim to embed a knowledge base into their system for object representation and semantic information, such as affordances. Finally, also connected to our work, the results in [14]–[16] explored using a knowledge base for accessing action templates during deployment.

We aim to extend the previous frameworks by the ability to generalize the task and to make it transferable to new environments. We split the task in actions with defined phases, where the system identifies start and end constraints on the action and explores the possible variance in the transport motions. Our contribution is a system designed to extract a task representation from one visual demonstration of a task. The extracted representation is not grounded in an environment or specific to a human or robot actor. Our goal is to represent the allowed variation of a task so that it can be easily transferred to a new environment. Capturing the necessary constraints of the task and specifying its possible relaxations should enable anyone, including a robot, execute the task in a new environment or determine if and why the task can not be executed.

## II. Approach to Task Segmentation and Representation

Fig. 2 displays the process of creation and usage of a task representation. Learning a task representation consists of creating the model with all its parameters from a visual task demonstration. The model is then used to create plans for agents to execute the task in their deployment environments.

In our system, a task is a graph, where nodes are the actions to be done in the task and the edges are time-/ordering-dependencies between the task's actions. An action is an elementary change in the environment and a task causes multiple changes in the environment. An action database represents actions as a 3-tuple $a_d = (D, F, C)$, where $D = (V_1 \times V_2 \times \ldots \times V_n)$ is the product of the value domain $V_i$ of

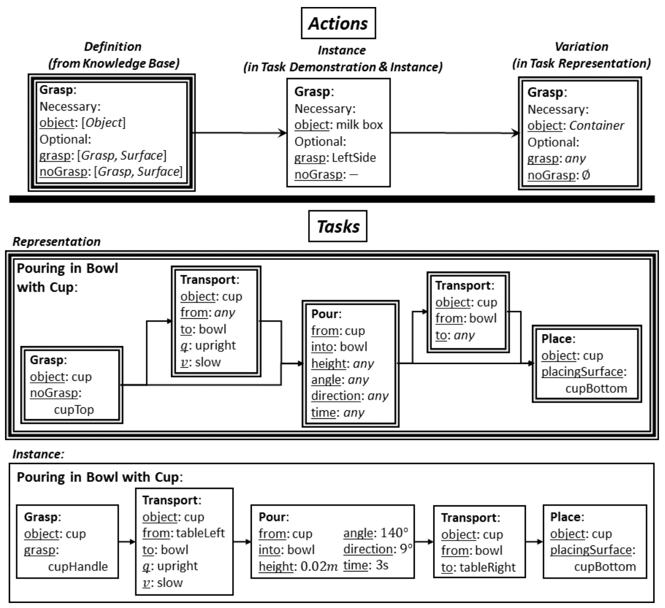
Actions
Definition (from Knowledge Base)
Grasp: Necessary: object: [Object] Optional: grasp: [Grasp, Surface] noGrasp: [Grasp, Surface]
Instance (in Task Demonstration & Instance)
Grasp: Necessary: object: milk box Optional: grasp: LeftSide noGrasp: –
Variation (in Task Representation)
Grasp: Necessary: object: Container Optional: grasp: any noGrasp: Ø
Tasks
Representation
Pouring in Bowl with Cup:
Grasp: object: cup noGrasp: cupTop
Transport: object: cup from: any to: bowl g: upright v: slow
Pour: from: cup into: bowl height: any angle: any direction: any time: any
Transport: object: cup from: bowl to: any
Place: object: cup placingSurface: cupBottom
Instance:
Pouring in Bowl with Cup:
Grasp: object: cup grasp: cupHandle
Transport: object: cup from: tableLeft to: bowl g: upright v: slow
Pour: from: cup into: bowl height: 0.02m angle: 140° direction: 9° time: 3s
Transport: object: cup from: bowl to: tableRight
Place: object: cup placingSurface: cupBottom

Fig. 3. Action instances are created from their definitions in the knowledge base based on the observation in a demonstration, and the allowed parameter variation is the context of a task is represented in the action variation. A task representation is a DAG of action variations. It is used to generate task instances. Note that *any* values represent the parameter's full value domain.

each parameter of the action. The top row of Fig. 3 exemplifies the parameters of a grasp action.

$F$ is a set of functions defining semantic information about the involved objects in the action. For pouring, one function $f = \{s_{active} | s_{active} \in ObjectSurfaces(o_i), o_i \in V_i\}$ defines the active surfaces of the objects involved in the action: a cup's top surface is the active surface in pouring action containing the cup. Information about the active surface is used by the task representation to speed-up determining which grasps are not allowed to execute on the object. Another semantic function defines object affordances before and after the action. The cup must be a container before and is no longer after the action has been executed, assuming a full pouring action.

$C$ is the set of (visual) check-functions that have to be fulfilled to (visually) recognize the action. For example, one check $c \in C_{pouring}$ for a pouring task is that the from parameter is a subconcept of *Container* and the into parameter is a subconcept of *Recipient*. Another check is that the angle of the pourer and poured parameters, i.e the respective surface normal vectors of the container and recipient involved in pouring, form an angle greater than 90°.

An **action instance** $a_i$ is the tuple of parameter values $p \in D, p = (p_1 \times p_2 \times \ldots \times p_n)$ that define the instance. The parameters $p_i$ are fixed, allowing no variation, because the parameters are bound to one action instantiation. Task instances contain only action instances.

We call an **action variation** $a_v = (v_1 \times v_2 \times \ldots \times v_n)$ a tuple defining for each parameter the allowed variation in its value. Formally, $\forall i : v_i \in 2^{V_i}$. Thus, $v_i$ is the set of allowed values for the action parameter in the context of a task representation. Task representations contain only action variations.

Fig. 4 shows the processing pipeline from a task demonstration to a task representation. In a first step, the observed

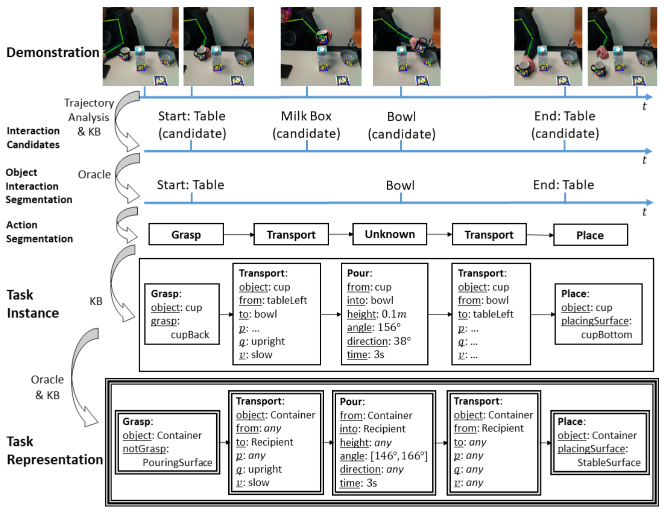
Demonstration
Trajectory Analysis & KB
Interaction Candidates
Start: Table (candidate)
Milk Box (candidate)
Bowl (candidate)
End: Table (candidate)
t
Object Interaction Segmentation
Oracle
Start: Table
Bowl
End: Table
Action Segmentation
Grasp
Transport
Unknown
Transport
Place
Task Instance
KB
Grasp: object: cup grasp: cupBack
Transport: object: cup from: tableLeft to: bowl p: ... q: upright v: slow
Pour: from: cup into: bowl height: 0.1m angle: 156° direction: 38° time: 3s
Transport: object: cup from: bowl to: tableLeft p: ... q: ... v: ...
Place: object: cup placingSurface: cupBottom
Oracle & KB
Task Representation
Grasp: object: Container notGrasp: PouringSurface
Transport: object: Container from: any to: Recipient p: any q: upright v: slow
Pour: from: Container into: Recipient height: any angle: [146°, 166°] direction: any time: 3s
Transport: object: Container from: Recipient to: any p: any q: any v: any
Place: object: Container placingSurface: StableSurface

Fig. 4. The processing pipeline from a demonstration to a task representation.

task, captured as a stream of RGB-D images, needs to be segmented into elementary actions that represent changes in the environment. We give the system the prior information that elementary actions can be identified by changes in the contact relations between the manipulator and structures in the environment. This leads to a basic, sometimes incomplete segmentation of the demonstration, which is further refined by analysing the interactions between the manipulated object and other objects along the manipulated path. With the help of the oracle, the task is further split into action segments and, with the knowledge base information, the parameters of the identified actions of the task are filled. Fig. 3 shows at the bottom a task instance created with the above segmentation approach. In the next step of the pipeline, the allowed variation of the task's action parameters, such that the essence of the demonstrated task is not changed, must be captured.

The allowed variation of an action's parameters in the context of a task is not determined through multiple demonstrations, but by letting the system generate alternate task instances by changing the values of the task's action parameters and checking with an *oracle* whether the different instance is still a valid task execution.

We aim to use non-expert oracles, who just know the constraints of the task without any knowledge in robotics. For having a normal, non-expert user interact with the system and to reduce the number of questions necessary to learn the task representation, we embed a knowledge base into the system. Several data from the knowledge base, for example object semantics and concepts, their surfaces and affordances, coarse action definitions, visual action confirmations, and which object concepts are participating in the action, are used to understand faster what was demonstrated and are thus necessary to reduce the number of questions for the oracle.

Furthermore, for targeted questions about the allowed variations and freedoms in the task, the knowledge base represents an action as having parameters which may vary in the instantiation of the action. For grasping, for example, the grasp points are an action parameter as well as the object being grasped. For transporting, the position and orientation

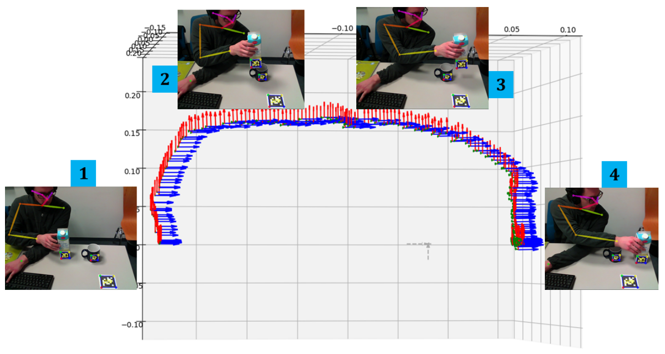

Fig. 5. Transportation task instance containing an irrelevant cup interaction.

trajectory, the velocity profile, the location of the start and goal are parameters of the action. In an action instantiation, these variable parameters are fixed to a specific value. However, in the action definition, i.e. what is stored in the knowledge base, the stored information is that there are variables that are allowed to differ in all possible instantiations of the action.

*Thus, what the system does in the learning phase, is retrieving the task representation that the person intended, from their demonstrated task instantiation, using the additional knowledge of action definitions from the knowledge base, to speed up the learning process.*

In the execution phase of Fig. 2 our proposed representation enables any robot with any kinematic structure to perform the task in any environment or to actively determine that the task can not be executed by the robot and which task constraint can not be fulfilled. Furthermore, the represented freedoms or constraint relaxations reduce the computational burden of the planner creating the path for a robot to follow. Knowing, for example, if a robot must exactly follow the demonstrated positions and orientations of the manipulated object or if only reaching the goal of the transportation action is important greatly simplifies the planner's work. In the case where the position and orientation of the manipulated object are not relevant, the planner can choose to create a simple interpolated trajectory with collision avoidance [17], or even the shortest or fastest path between the start and goal [18].

*Knowing that there is a flexibility in the task allows the planner to exploit it and create efficient execution plans to complete the task's goal.*

## III. Experiments

Our experiments set out to prove that the system can correctly segment one user demonstration into a task instance and then extract a task representation and that a planner can exploit the freedoms in a task representation's constraints to create a plan that suited for the kinematic structure of the robot and for the deployment environment of the task.

### A. Identification of Demonstrated Task Features

In one demonstration, see Fig. 5, a user intended to show an object movement task, with no constraints on the moved object-concept, no motion constraints, no grasping or placement constraints. During the demonstration however, the manipulated object entered the interaction volume of a cup and the system registered it as a potential segmentation

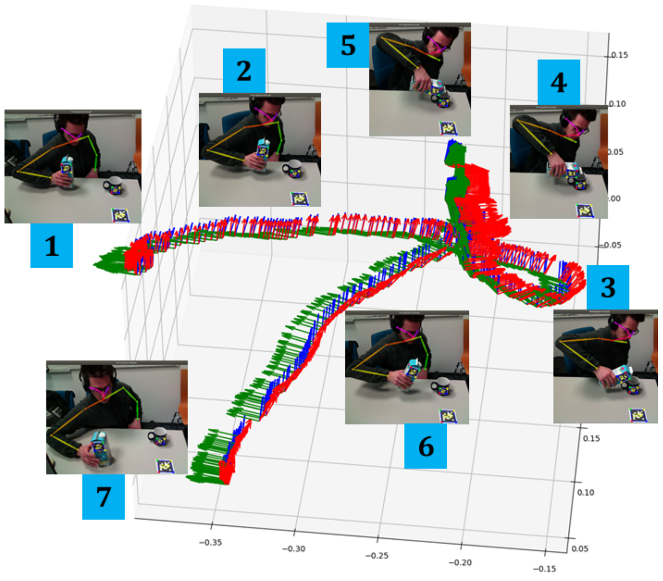

Fig. 6. Pouring task instance. Snapshots are ordered by time.

point. Once asked if the cup was relevant to the task, the user responded with 'no', marking the cup a particularity of the instance environment, not a feature of the task. Hence, the system segmented the demonstration into three action instances: grasping a milk box from the left surface, transporting it from the table position $A$ to the table position $B$ and placing the milk box on the bottom surface. Upon checking all action parameters, the system correctly determined the task representation as grasping any object from anywhere on the object, transport the object anyhow from anywhere to anywhere, and place the object on any surface.

In another demonstration, see Fig. 6, a user demonstrated a pouring task from a milk box into a cup. The cup was correctly identified by the system, and confirmed by the user, to interact with the manipulated object, the milk box. Looking into the knowledge base for the affordances of the cup and the milk box, the system sees the box is a *Container* and the cup is a *Recipient*, suggesting the possibility of a pouring action being executed in the interaction segment. This is confirmed by the executed motion in which the cup's top surface, having the concept of a *PourableSurface*, is facing the milk box's pourer surface.

Thus, the initial segmentation of the demonstration is grasping the milk box from the back surface, transportation from table position $C$ to the cup's interaction volume position $D$, pouring from the milk box's pouring surface into the cup's top surface using a maximal pouring height of 5.36cm, a maximal pouring angle of 133.45°, an approach direction of $-41.06°$ relative to the box's coordinate frame and a pouring time of 13.6s, followed by another transportation from the box's interaction volume point $E$ to the table position $F$ and finally, placing the milk box on its bottom surface.

## B. *Executing Pouring Task*

We have used CoppeliaSim [19] to simulate the execution of the represented pouring task with a Franka robot and have used object meshes of the milk box and of a bowl as the objects on which to execute the pouring task. The robot

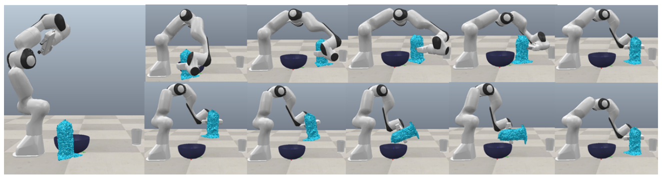

Fig. 7. Left large image shows the initial environment configuration (different than the demonstrated environment) with the opening of the milk box pointing to the up-right. The scanned mesh of the milk box is shown in light blue, the bowl in dark purple, and the cup in light grey. From left to right, first to second row, the robot executes a new pouring task instance stemming from our task representation.

used the freedoms of the task representation to create a new instance of the task representation which still fulfilled its goal of pouring. A chosen freedom was the poured object concept; it being any container instance, the robot selected the bowl instead of the cup, which was too far away from the static manipulator, to execute the pouring action. The next freedom was choosing the approach direction of the pouring action by sampling a direction angle between $\left[\frac{-\pi}{2}, \frac{\pi}{2}\right]$. The elimination of the interval $\left[-\pi, \frac{-\pi}{2}\right) \cup \left(\frac{\pi}{2}, \pi\right]$ is due to the robot's workspace preventing the robot from performing the pouring action. The approach direction determined the goal position of the transportation action between the milk box and the bowl and the path was linearly interpolated between the start and goal whilst keeping the orientation constant according to the specified constraint of the transportation action. Finally, another freedom was choosing the placing position of the milk carton to be just next to the bowl, eliminating the need for a more complex path. Fig. 7 shows snapshots of the robot's task execution in simulation.

## IV. CONCLUSION

This paper presented a method to represent a task from one visual demonstration using a non-expert oracle to explore possible task variations together with an autonomous knowledge-based decision-making algorithm to reduce the number of questions for the oracle – as to not burden or bore them. To this aim, a segmentation approach creates a task instance as a sequence of action instances – with their action parameters filled from the knowledge base action definitions. Full task representations are then extracted from such task instances. Performed experiments validated our segmentation and representation approach. Furthermore, the represented action variation parameters enable during deployment the creation of easier plans for different manipulators executing the task in different environments.

Our approach can be used in the Geriatronics context to enable elderly persons to teach their caring robot new tasks without them having robotics or programming knowledge. It requires a good knowledge base describing the world and a robot planner able to take advantage of the freedoms embedded in the task representation. Creating the knowledge base is a challenging process, which can in the future be eased up by learning from experience or using large language models. Regardless, the knowledge base is a powerful tool for speeding up the extraction of task representations. In future work, we also plan to extend our system to address multimodalities and to determine the representation of non-sequential tasks, e.g. eating or drinking.

## References

[1] Statistisches-Bundesamt, “Press release no. 511 of 2 december 2022,” 12 2022. [Online]. Available: https://www.destatis.de/EN/Press/2022/12/PE22_511_124.html

[2] Y. Kuniyoshi, M. Inaba, and H. Inoue, “Learning by watching: extracting reusable task knowledge from visual observation of human performance,” *IEEE Transactions on Robotics and Automation*, vol. 10, no. 6, pp. 799–822, 1994.

[3] H. Xiong, Q. Li, Y.-C. Chen, H. Bharadhwaj, S. Sinha, and A. Garg, “Learning by watching: Physical imitation of manipulation skills from human videos,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 7827–7834.

[4] R. Zoliner, M. Pardowitz, S. Knoop, and R. Dillmann, “Towards cognitive robots: Building hierarchical task representations of manipulations from human demonstration,” in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005, pp. 1535–1540.

[5] E. M. Orendt and D. Henrich, “Control flow for robust one-shot robot programming using entity-based resources,” in *2017 18th International Conference on Advanced Robotics (ICAR)*, 2017, pp. 68–74.

[6] ——, “Task generalization for robust one-shot robot programming using entity-based resources,” in *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2018, pp. 428–434.

[7] E. E. Aksoy, A. Abramov, J. Dörr, K. Ning, B. Dellen, and F. Wörgötter, “Learning the semantics of object–action relations by observation,” *The International Journal of Robotics Research*, vol. 30, no. 10, pp. 1229–1249, 2011. [Online]. Available: https://doi.org/10.1177/0278364911410459

[8] E. E. Aksoy, A. Abramov, F. Wörgötter, and B. Dellen, “Categorizing object-action relations from semantic scene graphs,” in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 398–405.

[9] M. Wächter and T. Asfour, “Hierarchical segmentation of manipulation actions based on object relations and motion characteristics,” in *2015 International Conference on Advanced Robotics (ICAR)*, 2015, pp. 549–556.

[10] M. Wächter, S. Schulz, T. Asfour, E. Aksoy, F. Wörgötter, and R. Dillmann, “Action sequence reproduction based on automatic segmentation and object-action complexes,” in *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2013, pp. 189–195.

[11] P. Gajewski and B. Indurkhya, “An approach to task representation based on object features and affordances,” *Sensors*, vol. 22, no. 16, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/16/6156

[12] A. Mitrevsk, P. G. Plöger, and G. Lakemeyer, “Ontology-assisted generalisation of robot action execution knowledge,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 6763–6770.

[13] M. Beetz, D. Beßler, A. Haidu, M. Pomarlan, A. K. Bozcuoğlu, and G. Bartels, “Know rob 2.0 — a 2nd generation knowledge processing framework for cognition-enabled robotic agents,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 512–519.

[14] N. Krüger, C. Geib, J. Piater, R. Petrick, M. Steedman, F. Wörgötter, A. Ude, T. Asfour, D. Kraft, D. Omrčen, A. Agostini, and R. Dillmann, “Object-action complexes: Grounded abstractions of sensory-motor processes,” *Robotics and Autonomous Systems*, vol. 59, no. 10, pp. 740–757, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889011000935

[15] K. Ikeuchi and T. Suchiro, “Towards an assembly plan from observation. i. assembly task recognition using face-contact relations (polyhedral objects),” in *Proceedings 1992 IEEE International Conference on Robotics and Automation*. Los Alamitos, CA, USA: IEEE Computer Society, may 1992, pp. 2171–2177. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/ROBOT.1992.219935

[16] G. Kazhoyan and M. Beetz, “Programming robotic agents with action descriptions,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 103–108.

[17] R. Laha, R. Sun, W. Wu, D. Mahalingam, N. Chakraborty, L. F. Figueredo, and S. Haddadin, “Coordinate invariant user-guided constrained path planning with reactive rapidly expanding plane-oriented escaping trees,” in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 977–984.

[18] R. Laha, W. Wu, R. Sun, N. Mansfeld, L. F. Figueredo, and S. Haddadin, “S*: On safe and time efficient robot motion planning,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[19] E. Rohmer, S. P. N. Singh, and M. Freese, “Coppeliasim (formerly v-rep): a versatile and scalable robot simulation framework,” in *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*, 2013, www.coppeliarobotics.com.