2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

MAkEable: Memory-centered and Affordance-based Task Execution Framework for Transferable Mobile Manipulation Skills

Christoph Pohl*, Fabian Reister*, Fabian Peller-Konrad and Tamim Asfour

***Abstract*— To perform versatile mobile manipulation tasks in human-centered environments, the ability to efficiently transfer learned skills, knowledge, and experiences from one robot to another or across different environments is critical. In this paper, we present *MAkEable*, a versatile uni- and multi-manual mobile manipulation framework that facilitates the transfer of capabilities and knowledge across different tasks, environments, and robots. Our framework integrates an affordance-based task description into the memory-centric cognitive architecture of the ARMAR humanoid robot family, which supports the sharing of experiences and demonstrations for transferring mobile manipulation skills. By representing mobile manipulation actions through affordances, i. e., interaction possibilities of the robot with its environment, we provide a unifying framework for the autonomous uni- and multi-manual manipulation of known and unknown objects in various environments. We demonstrate *MAkEable*'s applicability in real-world experiments for multiple robots, tasks, and environments. This includes grasping known and unknown objects, object placing, bimanual object grasping, memory-enabled skill transfer in a drawer opening scenario across two different humanoid robots, and a pouring task learned from human demonstration. *Code is available through our project page*[1].**

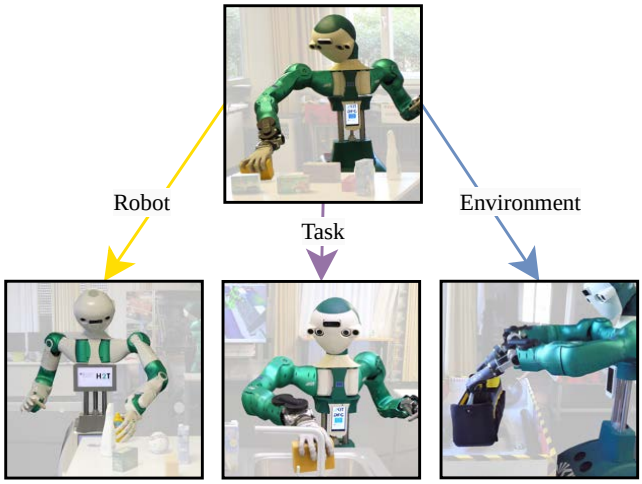

Fig. 1: Our framework facilitates triple-mode transfer, i. e., across tasks (e. g., *grasp* and *place*), robots (e. g., ARMAR-6 and ARMAR-DE) and environments (e. g., household and maintenance).

I. Introduction

In the rapidly evolving landscape of robotics, the ability to efficiently transfer skills, knowledge, and experiences from one robot to another or across diverse environments is pivotal (see Fig. 1). This could not only accelerate the deployment of robots into new settings but also significantly enhance their adaptability and functionality [1]. For example, when deploying humanoid robots to household scenarios, this translates to a seamless transition of a robot from one home to another, adapting to different layouts and task requirements without the need for extensive reprogramming. Additionally, having robots that share their experiences e. g., via a central knowledge base or a memory system, can bootstrap the transfer of learned skills and capabilities to other robots, tasks, and environments.

To date, in most cases, manipulation skills are designed with specific scenarios and contexts in mind, i. e., reusing skills across different robots or environments is often not feasible (e. g., [2] and [3]). To promote this kind of transferability at different levels of abstraction in dynamic, unstructured, and cumulative scenarios, a universal task description is required [1]. On the one hand, combining this description with a memory system facilitates the accumulation of a rich repository of mobile manipulation experiences that can be applied to solve novel problems. Therefore, a memory-centric task description fosters a collaborative learning environment among robots, enabling them to leverage knowledge about the success and failure of other robots. On the other hand, this task description can be decoupled from robot-specific information using the concept of affordances (i. e., interaction possibilities of the robot with its environment [4]) from cognitive psychology. By assigning action possibilities as properties to relevant objects and locations, affordances provide a way to reason about the environment in terms of what can be done with objects rather than which specific robot does it. Consequently, affordance-based representations are transferable between agents and environments (as seen in e. g., [5] and [6]).

To address the three modes of transferring knowledge and experience, i. e., transfer between tasks, robots, and environments, according to the taxonomy of [1], there is a need for such a universal framework that facilitates the flexible design and implementation of mobile manipulation tasks involving known and unknown objects in unstructured environments.

*The authors contributed equally (listed in alphabetical order).

The research leading to these results has received funding from the German Federal Ministry of Education and Research (BMBF) under the competence center ROBDEKON (13N14678), the Carl Zeiss Foundation through the JuBot project, and European Union's Horizon Europe Framework Programme under grant agreement No 101070596 (euROBIN).

The authors are with the High Performance Humanoid Technologies Lab, Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology (KIT), Germany {pohl, asfour}@kit.edu

[1]https://sw.pages.h2t.iar.kit.edu/makeable/project_page/

In this paper, we propose *MAkEable*, a memory-centered, affordance-based framework for mobile manipulation that unifies the description and execution of mobile manipulation actions (e. g., pick-and-place tasks, opening, pouring, etc.) across different tasks, environments, and robots. To the best of our knowledge, our approach is the first to facilitate the transfer of mobile manipulation skills and experiences across all three modes. It allows for the autonomous and semi-autonomous generation and execution of uni- and multi-manual manipulation actions while being flexible enough to support customization of individual steps to the user's needs and various scenarios. We provide several use cases and show the transfer between tasks and environments through uni- and bimanual grasping, placing of known and unknown objects, and transfer of a drawer opening skill to another robot using the humanoid robots ARMAR-6 [7] and ARMAR-DE. We also show that our framework is versatile enough to accommodate different robots by executing a pouring task in simulation.

II. Related Work

Similar to [1], we distinguish between three levels of task descriptions for robotic mobile manipulation that differ in their respective degree of abstraction. The semantically-lowest level encompasses only descriptions of single tasks using the highest information density (e. g., state-machines [8], [9] or behavior trees [10]) where transfer is difficult as they are tailored to specific robots and tasks. The highest level is that of a natural language description of tasks, like in many recent works that use *Large Language Models* (LLMs) for task planning (e. g., [11] or [12]), where only a goal state is provided in natural terms. However, no concrete information about the task (e. g., how an object needs to be grasped) is given, and only a high-level description is available. As mentioned in [1], it is easy to transfer to other robots but has no direct connection to the actual execution on a robot, necessitating an already implemented stack of low- to mid-level capabilities for each individual robot. Finally, at the intermediate level, actions and executions are abstracted into skills with specific goals in mind. In this level, enough information for execution exists about the task, but also enough abstraction to transfer these descriptions to other robots, environments, and tasks (e. g., [5], [6]). Therefore, this level can be naturally used to connect task descriptions of the highest level with the low-level execution. However, to be able to execute high-level tasks across different agents and environments, the medium abstraction level is where the actual transfer has to happen. For example, in [13], we introduced an approach to execute mobile pick-and-place tasks using high-level plans generated by an LLM. To be able to execute them on different robots, it was necessary to have a framework that could ground the steps of the plan to robot-specific instructions for the execution of the grasping and placing tasks.

In this context, the paper develops a task description and execution framework that facilitates the transfer of mobile manipulation skills across different robots, environments, and tasks. Therefore, in the following, we focus only on the medium abstraction level and categorize works depending on the number of transfer modes as described in [1].

A. Single-mode Transfer

The prevalent part of research so far has focused on the methods that are transferable across a single mode (either robot, task, or environment). For example, early works like that of [2], [3] or [14] tried to create systems that facilitate the transfer across environments by creating integrated hardware and software solutions for mobile manipulation tasks. Rovida and Kruger [15] present a modular framework for programming robots based on tasks and skills, which is organized into abstraction levels. Their lowest abstraction level, the *Device Layer*, allows transfer across robots in industrial setups. In [16], Keleştemu et al. present a mobile manipulation system for domestic environments. This system can execute grasping tasks in different household environments on the HSR robot. Some approaches, like [17] or [18], specifically focus on reactive grasping and provide special control architectures for robust transfer of grasping tasks across environments.

B. Dual-mode Transfer

A major research trend in recent years has been increasing the flexibility of mobile manipulation systems, therefore transitioning from a single-mode transfer to a dual-mode transfer. A modular, general-purpose software framework for mobile manipulation in household environments is introduced in [19]. It covers navigation, visual perception, manipulation, human-robot interaction, and high-level autonomy. Its versatility was demonstrated on the robots HSR and MSR-1 across various environments. With the increased performance and availability of *Foundation Models* for robotic applications in recent years, recent works like [20] and [21] have investigated *Open Vocabulary Mobile Manipulation* (OVMM). The aim of OVMM is "picking any object in any unseen environment, and placing it in a commanded location" [20], promoting a transfer across tasks and environments. However, these works have so far only been implemented for single robots. The *Affordance Templates Task Description Language* ([5], [6]) is particularly relevant to our approach. It also employs affordances to describe manipulation tasks in a robot-agnostic manner, which we took inspiration from. A key difference to our approach is that an *Affordance Template* has to be created for each task separately, hindering a transfer of capabilities and knowledge across tasks.

In contrast to the existing literature, our focus lies in creating a task description and execution framework that facilitates the transfer of mobile manipulation tasks and experiences to different robots and environments. We employ a memory-centered architecture that promotes interpretability in every step of our approach and the explainability of the results. Our affordance-based representation is inherently transferable between robots, tasks, and environments and covers actions like grasping, placing, opening, and pouring

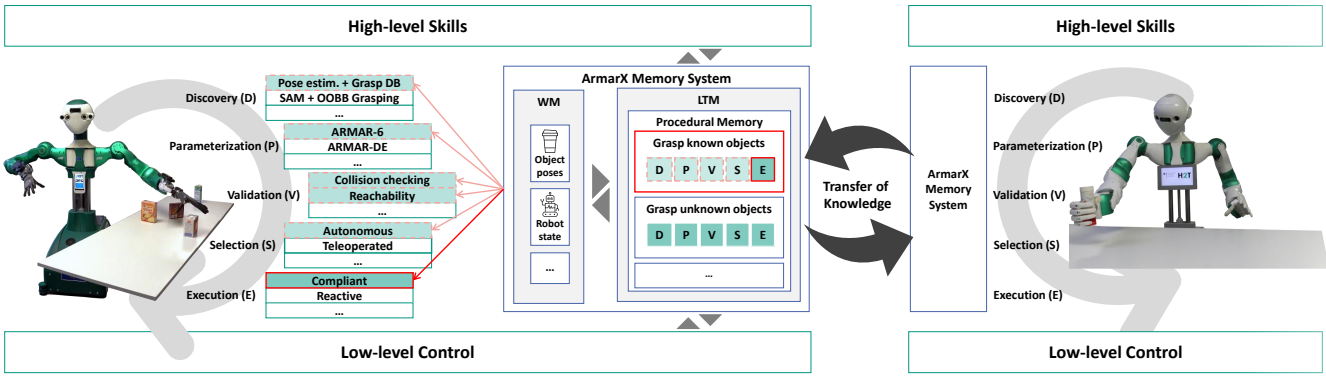

Fig. 2: Embedding of *MAkEable* into the memory-centric cognitive architecture [22] implemented in ArmarX. Several strategies that implement the five steps of the architecture (see Section III-C) are connected to the robot's memory.

in complex, unstructured scenarios. Therefore, we facilitate a triple-mode transfer of mobile manipulation skills.

III. The Framework

In this section, we present *MAkEable* and its proposed description for mobile manipulation tasks in unstructured environments for the autonomous execution of uni- and multi-manual actions across different robotic platforms. Based on the fundamental requirements from Section I, we present our system architecture and its leading design principles. As shown in Fig. 2, our framework is embedded into the memory of the cognitive architecture [22] of ArmarX [23] and makes use of its provided interpretable data format (*IDF*), facilitating the transfer across all three modes.

A. Design Principles

We identify various requirements for a software framework that provides skills that are transferable across tasks, robots, and environments. Such a framework should be **modular** to support various skill types, **extensible** regarding tasks and robots, and support "transferable and universal representations" [1], thus being **interpretable** and **explainable**. In developing our framework, we address these requirements through the following design principles: (i) An ***affordance-based*** design allows the unification of different action types for known and unknown objects and supports transfer across actions and tasks; (ii) a ***memory-centered*** design utilizing a memory system that supports explainability through introspection, learning from experience, and transfer of experience and knowledge. In addition, this memory system is modular and extensible; (iii) a ***robot-agnostic*** implementation of skills and data types supports the extensibility of the system and transfer to different robots; (iv) a ***strategy-based*** software architecture permits enough flexibility and precision to adapt to the specific environment, robot, and task, and finally (v) an ***end-effector-based*** representation of skills that allows defining unimanual and multi-manual actions, thus supporting a modular design. Next, we will explain the task description, which is derived from these design principles.

B. Task Description using IDF

The description of a manipulation task is – similar to [5] – based on the concept of affordances and is defined in terms of *IDF* objects, as shown in Fig. 3. As affordances are, by definition, agent-specific, we define the robot-agnostic counterpart to an `Affordance` to be an `ActionHypothesis`. An action hypothesis is, therefore, an end-effector pose in an abstract frame connected to an `ActionType`, like *Grasp*, *Place*, or *Push*. This facilitates the extraction of action candidates from visual perception independent of the robot.

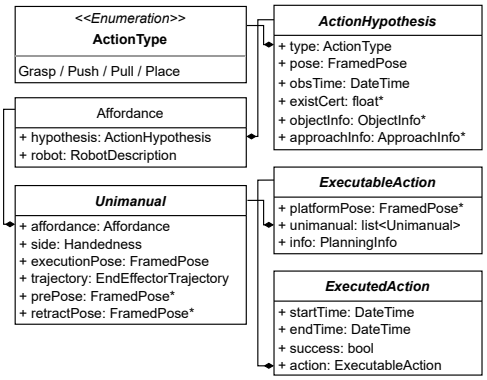

Fig. 3: Simplified class diagram of the *IDF* task description. Types marked with a "*" are optional.

The main object relevant to the execution is the `ExecutableAction`, which contains all relevant and necessary information for a specific robot. This object can contain up to n `Unimanual` actions, which consist of relevant information for a single end-effector. By generating an `ExecutableAction` with a `Unimanual` for each end-effector, we facilitate the execution of multi-manual manipulation actions[2]. An affordance-based manipulation action is defined to be an `EndEffectorTrajectory` (i. e., a framed trajectory of the end-effector with optional finger-joint values or hand-shape names for each keypoint) that is executed at the `executionPose` (i. e., an end-effector pose). Additionally, a pre- and retract pose can be defined, which will be approached before and after the execution of the action trajectory, respectively. After an `ExecutableAction`

[2]Note: Bimanual manipulation is generally much more complex than just executing independent trajectories for each arm, and would therefore require additional coordination between the arms based on high-level task information. For a more detailed discussion of the topic, see e. g., [24].

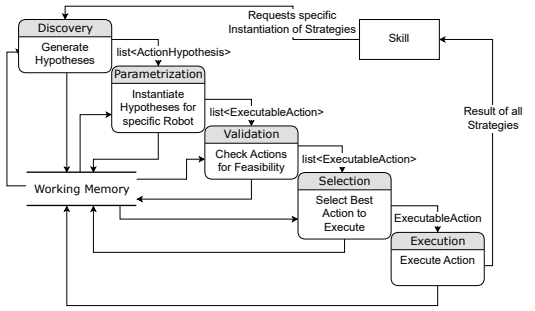

Fig. 4: Data flow in our framework visualizing the interaction of the *IDF* task description with the system architecture.

has been executed, its result and all relevant execution information are saved in an `ExecutedAction` for storage, introspection in the memory, and continual learning. Our task descriptions are embedded in our overall system architecture, as explained next.

C. System Architecture

Our integrated architecture is designed to facilitate the autonomous discovery and execution of uni- and multi-manual mobile manipulation actions on unknown objects in unstructured scenes while being flexible enough to adapt to different tasks, environments, and robots. To this end, we split the overall task of generating and executing actions into five distinct steps:

1) ***Discovery*** of actions (`ActionHypothesis`) from e. g., visual perception or prior scene knowledge.
2) ***Parameterization***. An `ExecutableAction` is derived from a `ActionHypothesis` for a specific robot, which contains all necessary information for execution (i. e., robot base poses, `EndEffectorTrajectory`, etc.).
3) ***Validation*** of all `ActionHypothesis` by checking for feasibility e. g., reachability, correct handedness, approach direction, collision checking.
4) ***Selection*** of the best `ExecutableAction` based on multiple criteria (e. g., execution height, execution side, platform movement, etc.).
5) ***Execution*** of the `ExecutableAction` on the targeted robot. In this paper, we use an approach similar to [25] combined with the navigation of [26].

Figure 4 shows an overview of the data flow of our architecture. All steps are implemented using the *Strategy* pattern to be easily exchangeable and customizable by the user. The basic workflow is controlled by a state machine that can decide during runtime which strategy to call based on the type information of *IDF*. Each step also has a specific interface for the implementation of external strategies for additional flexibility in case of special use cases. In the case of more general scenarios, it is possible to combine different strategies of one step, e. g., detecting `ActionHypotheses` for known and unknown objects at the same time. For ease of use, users can request certain combinations of strategies through high-level skills, as explained in Section III-D next.

D. Embedding into Memory Architecture

We fully integrate the proposed framework into the memory-centric cognitive architecture [22] implemented in ArmarX [23] where the memory acts as a mediator between the high-level skills of the robotic system and the low-level control components. Thus, all communication from high- to low-level passes through the robot's memory, which requires the data to have a specific format that is understandable by the memory (i. e., *IDF*). Through introspection of knowledge, our memory can adapt its behavior based on its content. Instead of being a simple static data storage, we believe that the robot's memory should play an active role so that it can adapt to incoming multi-modal and possibly associative streams of information. As part of the robot's long-term memory, executable skills are stored in the robot's procedural memory in the form of references to executable code, which can be parameterized based on the content of the robot's working memory. In the case of grasping and manipulation, a skill (e. g., `GraspingKnownObject` or `GraspingUnknownObject`) can be parameterized by one or more strategies per step of our proposed framework as depicted in Fig. 2. Moreover, the behavior of each step can be adapted based on the content of the robot's memory, e. g., object poses or common knowledge such as typical fetching and placing positions. The intermediate and final results of each step may be stored back in the memory. This is especially useful in the case of skill transfer. Since the knowledge inside the memory is already generalized and all our robots have the same memory structure (i. e., distributed working and long-term memory in the form of memory servers and segments as described in [22]), execution knowledge can directly be transferred from one memory to another.

IV. Use Cases

This section presents different use cases on which we evaluate our mobile manipulation framework in Section V. This includes grasping of known and unknown objects, object placing, and semi-autonomous bimanual grasping of unknown objects. Additionally, we demonstrate the opportunities our approach provides to learning skills from demonstration through example scenarios in which kinesthetic teaching and recorded human trajectories are used to derive new abilities for a humanoid and industrial robot, respectively.

A. Grasping of Known Objects

For the grasping of known objects, the *Discovery* and *Parameterization* steps can be combined. Grasp affordances are continuously discovered based on 6D object pose estimation and manually defined grasps stored in a grasp database as part of the robot's prior knowledge. Based on those instantiated grasp hypotheses, suitable robot placements are generated in a two-step approach: First, based on our previous work [26], initial collision-free robot placements are generated. Second, a local refinement is performed by solving a non-linear optimization problem similar to [27], which considers the end-effector target pose, joint-limits avoidance,

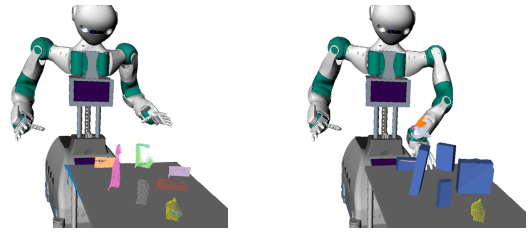

Fig. 5: Action hypothesis extraction for unknown objects. Point cloud segmentation using Segment Anything [32] (left) and object-oriented bounding boxes fitting [33] (right).

environmental collision, human-joint limits [28], [29] and maximizes the manipulability of the end-effector [30], [31] while also orienting the robot towards the object. Only if the aforementioned criteria are fulfilled to a certain extent, the action hypothesis is further considered. The *Execution* step makes use of the referenced object pose information in the `ExecutableAction` to refine the execution pose through re-localization of the object to account for inaccuracies in previous object pose estimation, self-localization, and eventual unforeseen movement of the object itself. The action execution and movement of the end-effectors are performed as described in our previous work [25]. There, the *tool center point* (TCP) is moved from the pre-pose to the execution pose until a force threshold is reached. At this point, the `EndEffectorTrajectory` – a coordinated hand and finger motion – is executed. Afterward, the TCP is moved to a secure retract pose.

B. Grasping of Unknown Objects

In order to discover grasp affordances and to generate action hypotheses for unknown objects, we use an RGB-D camera. As shown in Fig. 5, we first segment the color image using Segment Anything [32]. The action hypotheses are generated in the *Discovery* step according to [33]: object-oriented bounding boxes (OOBB) are fit to each segment in the point cloud. If the OOBB are within certain margins that conform to the robot's end-effector, grasp hypotheses are generated along the sides of the box for left- and right-handed grasps.

C. Object Placement at Common Places

Contextual knowledge about known objects can be added via the robot's memory to solve e. g., a pick-and-place task. This includes *common places*, i. e., grounded spatial symbols that indicate where to search or where to place an object. This symbolic representation of common sense knowledge is grounded in the continuous real world in order to be useful for execution. A common place is a volumetric space defined as either absolute or relative to an object class or instance. Each object can have multiple prioritized common places which the robot can choose from in the given scene. Fig. 6 shows several common locations used in our experiments, including their symbolic labels and sub-symbolic real positions. Here, knowledge about common places is provided through *Prior Knowledge*, i. e., knowledge given by the programmer to the robot and available from

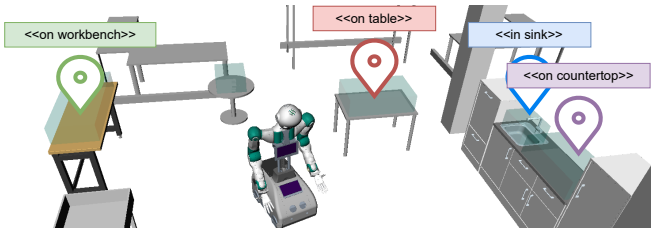

Fig. 6: Common places used in our experiments. The colored labels indicate symbolic names for each common place and the light-green boxes depict their respective position and extents in the global frame.

startup, but can also be learned from experience through the episodic memory.

D. Bimanual Grasping of Unknown Objects

For the discovery of bimanual grasp affordances, an approach combining the teleoperation from [34] with the hypothesis generation of [35] can be used. The grasp candidates were generated by a human operator by clicking on a specific point in an interactive visualization of the scene during the *Discovery* step. The pose of the `ActionHypothesis` was then generated using the averaged local surface information of the point cloud by calculating the *Local Curvature Frame* at that point and defining the pose relative to that frame [35]. The pose can be adapted by the operator after the initial generation of the `ActionHypothesis`. The *Parameterization* step then only generates a `Unimanual` action for each `ActionHypothesis` and combines them into one `ExecutableAction` and computes a platform placement centered in the middle between both hands. During the *Execution*, a bimanual grasp candidate is treated equivalently to two independent grasp candidates by concurrently executing one grasp candidate with each arm. After each phase, the arms stop until both arms have finished the phase to synchronize the grasping process between both arms. As we do not use any form of coordination between the arms beside the four synchronization points (i. e., `prePose`, `executionPose`, `trajectory` and `retractPose`), the compliant behavior of the controller helps to compensate for small misalignments of the tool center points with respect to each other when lifting or carrying an object bimanually.

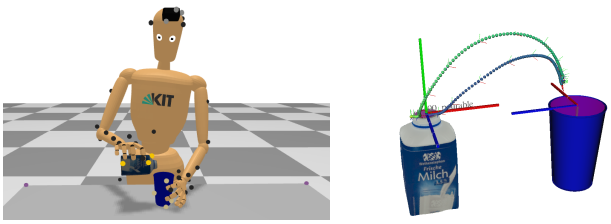

Fig. 7: Exemplary pouring motion selected from the *KIT Bimanual Manipulation Dataset* [36] (left). Extracted reference motion of the bottle's *pour* affordance frame relative to the cup's *fill* affordance frame (right).

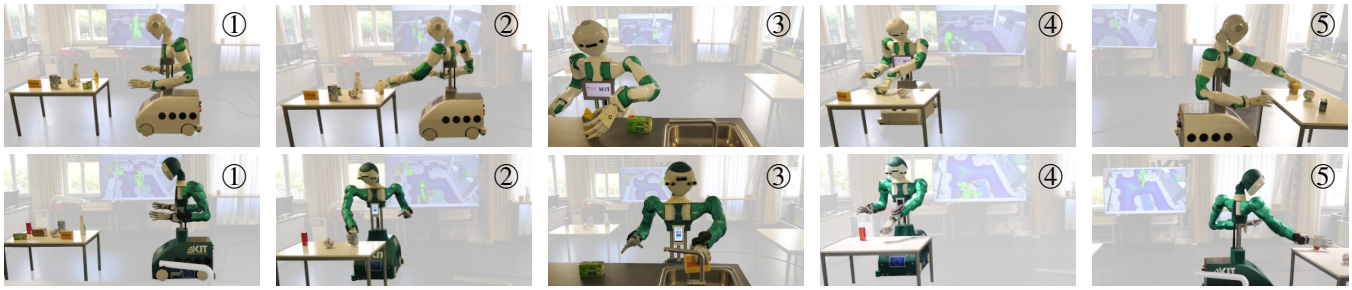

Fig. 8: Table-clearing of known and unknown objects with ARMAR-DE and ARMAR-6. ① Initial setup, ② grasping of known objects, ③ placing of known objects, ④ grasping of unknown objects, and ⑤ placing of unknown objects.

E. Learning to Interact with Articulated Objects using Kinesthetic Teaching

We show the applicability of our framework in transferring learned skills between robots using a drawer-opening task. More specifically, we show that our framework can be used for transferring knowledge and skills on the lowest abstraction level (i. e., *trajectory demonstrations*, as defined in [1]). In this use case, we can rely on the abstractions that the `EndEffectorTrajectory` provides. As each key-point can have an optional shape name associated, we can kinesthetically teach-in the trajectory and only tell the robot when to "*open*" and "*close*" its end-effector, or when to use preshapes, e. g., for a "*hook*" grasp. Based on the demonstration, the robot learns a representation of the trajectory in the abstract end-effector frame relative to a local affordance frame, e. g., the drawer's handle. In this format, the learned trajectory can directly be loaded into the memory of other robots so that they may execute the same action but at a new location of the affordance with their own definition of the shapes "*open*" and "*close*".

F. Learning and Transferring Motion of Affordance Frames from Human Demonstration

To demonstrate the flexibility of our framework in incorporating new knowledge, e. g., through observational learning, we extend the above approach of representing an end-effector trajectory in a local frame: Instead of focusing on the robot's gripper motion, we focus here on the motion of object affordance frames relative to each other – inspired by [37]. This is particularly relevant for tasks where the consideration of two affordances is necessary, possibly through tool use. Examples are cutting with a knife or pouring water from a bottle into a glass. In the latter example, the motion of a local *pour* frame of a bottle can be expressed in a *fill* affordance frame of a glass, as exemplified in Fig. 7. Such a representation can easily be transferred to new pouring tasks as long as the affordance frames are present – which we assume to be given as part of the object description. When confronted with a new situation, we learn *Via-point Movement Primitives* [38] from the reference motion and adapt the start pose accordingly. Via-points can be specified to preserve additional characteristics of the motion.

V. Experiments

We performed several real-world experiments on the ARMAR humanoid robots to demonstrate the applicability of our framework to transfer across all modes (tasks, robots, and environments) in realistic environments. The experiments consist of a table-clearing, a box-picking and a drawer-opening scenario. An additional experiment in simulation shows the transfer across agents and proves the applicability to non-humanoid robots. The videos on our project page show the execution of all experiments.

A. Clearing a Table with Known and Unknown Objects

We show the generalization and transfer of manipulation tasks across different robots using the two humanoid robots ARMAR-6 [7] and ARMAR-DE in a table-clearing setup (see Fig. 8). This requires the robots to grasp known and unknown objects and place them at common places, covering the corresponding use cases Section IV-A-IV-C. Both robots are equipped with two anthropomorphic 8 degrees of freedom (DoF) arms and two underactuated five-finger hands with 2 DoF (ARMAR-6) and 4 DoF (ARMAR-DE). For 6D object pose estimation, we use an RGBD-based pose estimation on both platforms and additionally a stereo-based pose estimation on ARMAR-6. Each table-clearing experiment consists of 7 different rigid and deformable household objects, which are placed arbitrarily on a table in structured clutter. Each of the known objects is associated with a common place (*sink*, *kitchen countertop*, or *workbench*). As long as the robot recognizes known objects, it will prioritize manipulating them before unknown objects. Due to the aforementioned differences in 6D object pose estimation, the robots treat different objects as known and unknown. ARMAR-DE is able to recognize the *mustard*, the *bio-milk*, the *apple-tea* and the *spraybottle*. The first three objects should be placed on the *countertop* while the latter should be placed on the *workbench*. In addition, ARMAR-6 is able to recognize the *screwbox*, which should be placed on the *workbench*, and the *sponge*, which should be placed in the *sink*. All unknown objects or objects that the robot cannot recognize should be placed on the free table next to the kitchen as shown in Fig. 6. Both robots were able to clear the table, as shown in the accompanying video.

B. Box Picking through Bimanual Grasping of Unknown Objects

To showcase the ability of our framework to handle more than unimanual actions and incorporate user feedback through teleoperation, we performed a number of semi-autonomous, bimanual pick-and-place executions of larger

Fig. 9: The humanoid robot ARMAR-6 grasping and carrying multiple objects (exhaust, pan, and pipe) bimanually.

objects on ARMAR-6. The bimanual grasping approach explained in Section IV-D was used for all executions, demonstrating the successful transfer between tasks and environments. After an object was successfully lifted, ARMAR-6 navigated autonomously to a second box and placed the object there. For the bimanual *placing* of objects, a similar strategy to the unimanual case was used: The object was lowered with both arms until a force threshold was surpassed and the hands were opened. The only difference to the unimanual *placing* is that the `EndEffectorTrajectory` only has a single waypoint where the hand is opened. Fig. 9 shows exemplary successful bimanual grasp executions of different objects.

C. Memory-enabled Transfer of a Drawer-Opening Skill

The experimental setup for the drawer opening task explained in Section IV-E, and its results can be seen in Fig. 10. ARMAR-6 is given the task of opening a drawer. Although the robot has an understanding of the "*Open*" affordance, it does not know how to interact with the drawer to open it (i. e., no suitable `EndEffectorTrajectory` is known to the robot). A human is asked by the robot to demonstrate the motion through kinesthetic teaching from a suitable `executionPose`, which is derived from the known position of the drawer's handle in the prior knowledge. This experience is preserved and stored in the robot's procedural memory. Due to the universal description, the learned motion can easily be transferred to other platforms, e. g., ARMAR-DE. To show this, ARMAR-DE is tasked to open another drawer by leveraging the gained knowledge through ARMAR-6. As shown in Fig. 10, it can successfully instantiate the drawer opening skill for its left arm (even though the demonstration was for the right arm of ARMAR-6) and execute it at the position of the new drawer.

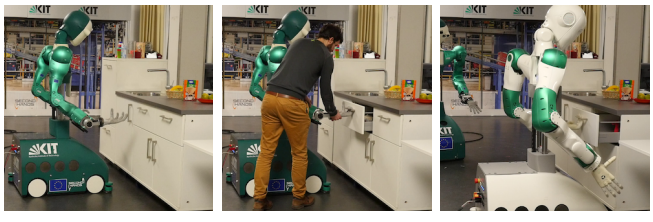

Fig. 10: Transfer of drawer opening skill learned through kinesthetic learning from ARMAR-6 to ARMAR-DE.

D. Grasping and Pouring with a Non-Humanoid Robot

To demonstrate the versatility and extensibility of our framework, we exemplary instantiate the grasping and pouring skill for the Omni-Frankie [31] featuring a 7 DoF

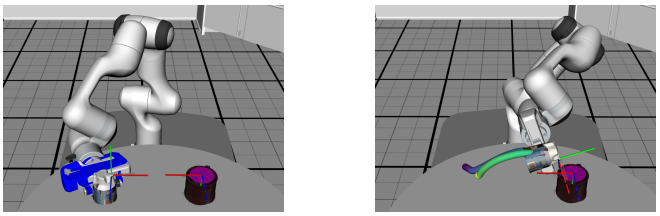

Fig. 11: Omni-Frankie grasping a milk jug (left) and pouring milk into a mug using an end-effector trajectory learned and adapted from human demonstration (right).

Franka-Emika Panda manipulator on a Ridgeback mobile base in simulation. In contrast to the ARMAR humanoid robots, this robot has a parallel gripper. Yet, due to the universal task and robot description, our framework can generate grasp hypotheses for the robot and gripper to grasp and lift the object (Fig. 11). In addition, we transfer the reference pour motion described in Section IV-F to filling milk from a milk jug into a coffee mug as shown in Fig. 11. This also showcases a successful transfer between different embodiments and environments.

VI. Conclusion

This paper presented *MAkEable*, a modular, memory-centered, and affordance-based grasping and mobile manipulation framework, unifying the autonomous manipulation of known and unknown objects in various environments. In multiple complex real-world and simulated experiments, we showed that our framework (i) can be used for different task definitions, such as grasping known and/or unknown objects, (ii) can be applied to different robots with different kinematics (i. e., end-effectors), (iii) supports multiple autonomy levels (full autonomy, semi-autonomy, and teleoperation), and (iv) can be used for the execution of uni- and bimanual actions. Thereby, we demonstrated the capability of our approach to transfer knowledge and experience across tasks, environments, and robots. Additionally, we showed that the link to a memory system, as part of a cognitive architecture, offers contextual awareness, supporting the utilization of common knowledge in the context of manipulation tasks and also facilitating learning from both success and failure. By creating a task description of the medium abstraction level, we facilitate the execution of manipulation actions while being independent of robots and environments. This way, we provide a bridge between high-level, natural language instructions and the corresponding low-level, robot-specific execution of the required skills (as e. g., in [13]).

Future work will consist of incorporating additional feedback into our system to enable a more closed-loop approach to mobile manipulation. To account for failures, especially during grasping, we will combine our framework with more reactive mobile manipulation approaches. Additionally, integrating an online failure detection (e. g., [39]) would increase the robustness of our framework. In order to perform a reproducible quantitative evaluation of the overall framework, we plan to standardize an evaluation scenario that focuses on grasping known and unknown objects in highly cluttered scenes.

ACKNOWLEDGEMENTS

We would like to thank Rainer Kartmann, Patrick Hegemann, Noémie Jaquier, Abdelrahman Younes, and Andre Meixner for their contributions, support, and assistance during the development of this framework.

REFERENCES

[1] N. Jaquier, M. C. Welle, A. Gams, K. Yao, B. Fichera, A. Billard, A. Ude, A. Tamim, and D. Kragic, "Transfer Learning in Robotics: An Upcoming Breakthrough? A Review of Promises and Challenges," *International Journal of Robotics Research (IJRR)*, 2024.

[2] P. Nebot and E. Cervera, "An integrated agent-based software architecture for mobile and manipulator systems," *Robotica*, vol. 25, pp. 213–220, 2007.

[3] A. Hermann, Z. Xue, S. W. Rühl, and R. Dillmann, "Hardware and Software Architecture of a Bimanual Mobile Manipulator for Industrial Application," in *IEEE International Conference on Robotics and Biomimetics*, 2011, pp. 2282–2288.

[4] J. J. Gibson, "The theory of affordances," in *The Ecological Approach to Visual Perception*. Houghton Mifflin, 1979, ch. 8, pp. 119–137.

[5] S. Hart, P. Dinh, and K. Hambuchen, "The Affordance Template ROS Package for Robot Task Programming," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 6227–6234.

[6] S. Hart, A. H. Quispe, M. W. Lanighan, and S. Gee, "Generalized Affordance Templates for Mobile Manipulation," in *IEEE International Conference on Robotics and Automation*, 2022, pp. 6240–6246.

[7] T. Asfour, M. Wächter, L. Kaul, S. Rader, P. Weiner, S. Ottenhaus, *et al.*, "ARMAR-6: A High-Performance Humanoid for Human-Robot Collaboration in Real World Scenarios," *IEEE Robotics & Automation Magazine*, vol. 26, no. 4, pp. 108–121, 2019.

[8] M. Wächter, S. Ottenhaus, M. Kröhnert, N. Vahrenkamp, and T. Asfour, "The ArmarX Statechart Concept: Graphical Programming of Robot Behaviour," *Frontiers in Robotics & AI*, vol. 3, p. 33, 2016.

[9] J. Bohren and S. Cousins, "The SMACH High-Level Executive," *IEEE Robotics & Automation Magazine*, vol. 17, pp. 18–20, 2010.

[10] M. Iovino, J. Förster, P. Falco, J. J. Chung, R. Siegwart, and C. Smith, "On the programming effort required to generate Behavior Trees and Finite State Machines for robotic applications," in *IEEE International Conference on Robotics and Automation*, 2023, pp. 5807–5813.

[11] B. Liu, Y. Jiang, X. Zhang, Q. Liu, S. Zhang, J. Biswas, and P. Stone, "LLM+P: Empowering Large Language Models with Optimal Planning Proficiency," *arXiv preprint arXiv:2304.11477*, 2023.

[12] A. Brohan, Y. Chebotar, C. Finn, K. Hausman, A. Herzog, D. Ho, J. Ibarz, A. Irpan, E. Jang, R. Julian, *et al.*, "Do As I Can, Not As I Say: Grounding Language in Robotic Affordances," in *Conference on Robot Learning (CoRL)*. PMLR, 2023, pp. 287–318.

[13] T. Birr, C. Pohl, A. Younes, and T. Asfour, "AutoGPT+P: Affordance-based Task Planning with Large Language Models," in *Robotics Science and Systems (RSS)*, 2024.

[14] J. A. Bagnell, F. Cavalcanti, L. Cui, T. Galluzzo, M. Hebert, M. Kazemi, M. Klingensmith, J. Libby, T. Y. Liu, N. Pollard, M. Pivtoraiko, J.-S. Valois, and R. Zhu, "An Integrated System for Autonomous Robotics Manipulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, p. 2955–2962.

[15] F. Rovida and V. Kruger, "Design and development of a software architecture for autonomous mobile manipulators in industrial environments," in *IEEE International Conference on Industrial Technology (ICIT)*, 2015, pp. 3288–3295.

[16] T. Keleştemur, N. Yokoyama, J. Truong, A. A. Allaban, and T. Padir, "System Architecture for Autonomous Mobile Manipulation of Everyday Objects in Domestic Environments," in *International Conference on Pervasive Technologies Related to Assistive Environments*. ACM, 2019, pp. 264–269.

[17] M. Logothetis, G. Karras, S. Heshmati-Alamdari, P. Vlantis, and K. Kyriakopoulos, "A Model Predictive Control Approach for Vision-Based Object Grasping via Mobile Manipulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 1–6.

[18] B. Burgess-Limerick, C. Lehnert, J. Leitner, and P. Corke, "An Architecture for Reactive Mobile Manipulation On-The-Move," in *IEEE International Conference on Robotics and Automation*. IEEE, 2023, pp. 1623–1629.

[19] J.-B. Yi, T. Kang, D. Song, and S.-J. Yi, "Unified Software Platform for Intelligent Home Service Robots," *Applied Sciences*, vol. 10, no. 17, p. 5874, 2020.

[20] S. Yenamandra, A. Ramachandran, K. Yadav, A. Wang, M. Khanna, T. Gervet, T.-Y. Yang, V. Jain, A. W. Clegg, J. Turner, Z. Kira, M. Savva, A. Chang, D. S. Chaplot, D. Batra, R. Mottaghi, Y. Bisk, and C. Paxton, "HomeRobot: Open-Vocabulary Mobile Manipulation," *arXiv preprint arXiv:2306.11565*, no. arXiv:2306.11565, 2024, arXiv:2306.11565 [cs].

[21] P. Liu, Y. Orru, C. Paxton, N. M. M. Shafiullah, and L. Pinto, "OK-Robot: What Really Matters in Integrating Open-Knowledge Models for Robotics," *arXiv preprint arXiv:2401.12202*, no. arXiv:2401.12202, 2024, arXiv:2401.12202 [cs].

[22] F. Peller-Konrad, R. Kartmann, C. R. G. Dreher, A. Meixner, F. Reister, M. Grotz, and T. Asfour, "A Memory System of a Robot Cognitive Architecture and its Implementation in ArmarX," *Robotics & Autonomous Systems*, vol. 164, pp. 1–20, 2023.

[23] N. Vahrenkamp, M. Wächter, M. Kröhnert, K. Welke, and T. Asfour, "The robot software framework ArmarX," *it - Information Technology*, vol. 57, 2015.

[24] F. Krebs and T. Asfour, "A Bimanual Manipulation Taxonomy," *IEEE Robotics & Automation Letters*, vol. 7, no. 4, pp. 11 031–11 038, 2022.

[25] C. Pohl, P. Hegemann, B. An, M. Grotz, and T. Asfour, "Humanoid Robotic System for Grasping and Manipulation in Decontamination Tasks," *at - Automatisierungstechnik*, vol. 70, pp. 850–858, 2022.

[26] F. Reister, M. Grotz, and T. Asfour, "Combining navigation and manipulation costs for time-efficient robot placement in mobile manipulation tasks," *IEEE Robotics & Automation Letters*, vol. 7, no. 4, pp. 9913–9920, 2022.

[27] D. Rakita, H. Shi, B. Mutlu, and M. Gleicher, "CollisionIK: A Per-Instant Pose Optimization Method for Generating Robot Motions with Environment Collision Avoidance," in *IEEE International Conference on Robotics and Automation*, 2021, pp. 9995–10 001.

[28] K. Luttgens, H. Deutsch, and N. Hamilton, *Kinesiology: Scientific Basis of Human Motion*. Brown & Benchmark, 1992.

[29] C. Gäbert, S. Kaden, and U. Thomas, "Generation of human-like arm motions using sampling-based motion planning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021, pp. 2534–2541.

[30] T. Yoshikawa, "Manipulability of Robotic Mechanisms," *The international journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.

[31] J. Haviland, N. Sünderhauf, and P. Corke, "A Holistic Approach to Reactive Mobile Manipulation," *IEEE Robotics & Automation Letters*, vol. 7, no. 2, pp. 3122–3129, 2022.

[32] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, *et al.*, "Segment Anything," in *IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4015–4026.

[33] R. Grimm, M. Grotz, S. Ottenhaus, and T. Asfour, "Vision-Based Robotic Pushing and Grasping for Stone Sample Collection under Computing Resource Constraints," in *IEEE International Conference on Robotics and Automation*, 2021, pp. 6498–6504.

[34] P. Kaiser, D. Kanoulas, M. Grotz, L. Muratore, A. Rocchi, E. M. Hoffman, N. G. Tsagarakis, and T. Asfour, "An Affordance-based Pilot Interface for High-Level Control of Humanoid Robots in Supervised Autonomy," in *IEEE-RAS International Conference on Humanoid Robots*, 2016, pp. 621–628.

[35] C. Pohl and T. Asfour, "Probabilistic Spatio-Temporal Fusion of Affordances for Grasping and Manipulation," *IEEE Robotics & Automation Letters*, vol. 7, no. 2, pp. 3226–3233, 2022.

[36] F. Krebs, A. Meixner, I. Patzer, and T. Asfour, "The KIT Bimanual Manipulation Dataset," in *IEEE-RAS International Conference on Humanoid Robots*, 2021, pp. 499–506.

[37] M. Muhlig, M. Gienger, S. Hellbach, J. J. Steil, and C. Goerick, "Task-level Imitation Learning using Variance-based Movement Optimization," in *IEEE International Conference on Robotics and Automation*, 2009, pp. 1177–1184.

[38] Y. Zhou, J. Gao, and T. Asfour, "Learning Via-Point Movement Primitives with Inter- and Extrapolation Capabilities," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 4301–4308.

[39] P. Hegemann, T. Zechmeister, M. Grotz, K. Hitzler, and T. Asfour, "Learning Symbolic Failure Detection for Grasping and Mobile Manipulation Tasks," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022, pp. 4302–4309.