Autonomous Narration of Humanoid Robot Kitchen Task Experience

Qingxiaoyang Zhu[1], Vittorio Perera[2], Mirko Wächter[1], Tamim Asfour[1] and Manuela Veloso[2]

***Abstract*— The progress in humanoid robotics research has led to robots that are able to perform complex tasks with a certain level of autonomy by integrating perception, action, planning, and learning capabilities. However, robot capabilities are still limited in regard to how they externalize their internal state and world state, i.e. their sensorimotor experience, and how they explain which tasks they performed and how they performed these tasks. In other words, their capability in conveying information to the user in a way similar to what humans do is limited. To this end, we present a verbalization system that generates natural language explanations of the robot's past navigation and manipulation experience. We propose a three-layered model to represent robot experience which doubles as a retrievable episodic memory. Through the memory system, the robot can select a matching experience given a user query. In order to generate flexible narrations, we use verbalization parameters to capture user preferences. We show that our verbalization algorithm is capable of producing appropriate results based on these verbalization parameters. The proposed verbalization system is able to generate explanations for navigation as well as grasping and manipulation tasks. The resulting system is evaluated in a pick-and-place kitchen scenario.**

I. Introduction

Humanoid household robots are able to execute tasks in human-centered environments with a certain degree of autonomy [2][3]. To automatically accomplish tasks specified by users such as preparing salad and putting dishes in a dishwasher, sophisticated robots employ symbolic planning and manipulation skills which coordinate the robot's motor and perception capabilities. Progress in mechatronics, learning, interaction and system integration has led to robots with advanced capabilities such as accomplishing complex assignments and understanding human beings [4]. However, robot capabilities regarding the externalization of their experience are still very limited. Robots are still unable to explain which tasks they performed and how they performed these tasks and convey such information to the user. Sometimes things do not go as expected, e.g. the task was not completed in a timely manner or mistakes were made, and people would wonder, what happened? To answer this question, a user friendly interface that allows end users to comprehend the robot's experience needs to be provided. Therefore, we present a verbalization system, which lets the robot narrate its past experience in natural language. Fig. 1 shows the underlying idea of having robots able to recall previous experience and generate natural language description of the executed task. The work is inspired and is based on the *verbalization* concept of autonomous robots presented in [1], where verbalization of navigation tasks was investigated. In this work, we build on this concept and explore the possibilities of verbalization of manipulation tasks and demonstrate the use of verbalization system in the context of household tasks.

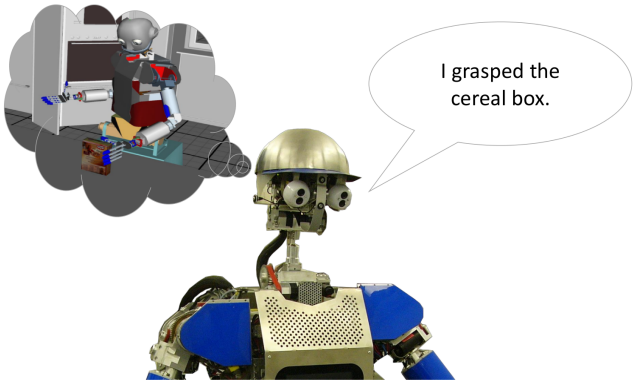

Fig. 1. The verbalization system generates a natural language description based on recorded experience during task execution of the robot.

Normally, humanoid household robots work in a human centered environment. Therefore, we consider natural language between human users and robots to be an efficient interaction method to provide more transparency in the robot's behavior [4]. Additionally, it also increases users' trust in intelligent systems [5]. In order to map humanoid household robots' past experience into natural language narrations that satisfy user preferences, we developed a modular verbalization system. The developed pipeline of the verbalization system is shown in Fig. 2 and consists of six major components. During the robots' autonomous operation, a *Log File Generator* runs alongside it. The *Log Files Generator* records significant events that happened during the task execution. Later when users ask what the robot did, the *Verbalization Parameter Determination* module will capture the users' interest and convert them into verbalization parameters: *specificity, abstraction,* and *semantic*. The robot will then select an experience from the episodic memory according to users' interest using *Log File Filter*. Because the selected experience is still in the form of raw data, the *Information Fusion* module fuses the selected experience with the environmental information from the *Memory System*. Finally, the system generates an appropriate interpretation of this information through the *Narration Generation* module. The resulting narration can vary in accordance to the levels in

The research leading to these results has received funding from the European Union's Horizon 2020 Research and Innovation programme under grant agreement No 643950 (SecondHands).

[1]The authors are with the Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology, Karlsruhe, Germany asfour@kit.edu

[2]The authors are with the Carnegie Mellon University (CMU), School of Computer Science, Pittsburgh, USA mmv@cs.cmu.edu

specificity, abstraction, and semantic.

The paper is organized as follows. Section II gives an overview of related work in the area of robot and cognitive system narration. In Section III, we describe how robot experiences are recorded and represented as the episodic memory. In Section IV, we explain how robots narrate their past experience in natural language according to user interests through a flexible verbalization algorithm. We demonstrate our concept by implementing the verbalization system on our kitchen robot ARMAR-III [3], and evaluate the narration of the robot's past experience in the context of a pick-and-place scenario in Section V. In Section VI, we conclude our work.

II. Related Work

There exist several works in the field on automatically generating a natural language narration of robot experience, which consists of planned and executed behaviors ([1], [6]), on interpreting perceived robot behaviors through natural language ([7], [8], [9]), as well as on producing instructions for people based on planned actions ([10], [11], [12]).

The work in [1] and [6] are most related to our work. In [1], the verbalization concept is proposed to generate description of navigation experiences, which converts traversed route into natural language. Moreover, route narratives are variable in representation according to user preferences which are expressed by the verbalization space. In the verbalization space, three parameters namely *Locality, abstraction*, and *specificity* are used to model users preferences. The method in [6] allows robots to dynamically determine values of the verbalization parameters through learning the mapping from dialogue to verbalization space. However, both works mainly focus on annotating navigation experience of autonomous robot with natural language.

Related to the topic of explaining robot behaviors, robot soccer commentator systems are capable of understanding time-varying scenes and generating summaries of robot actions in the robot soccer game (see [7], [8]). The commentator system accepts live feeds of RoboCup games [8] or simulated RoboCup games [7] as input to perceive robot behaviors from the outside. Afterwards, the system generates live reports on the game after recognizing significant events.

Another related work discusses the generation of understandable guidance of planned navigation route for users. In [10], a real time navigation system that understands drivers' queries in natural language and generates appropriate answer is implemented. The work in [11] proposes that navigation systems should use landmarks and other visible features to describe routes rather than only use street names and distance if a natural language description is required. Furthermore, Bohus et al. indicate in [12] that navigation directions should take locations into account. Besides, directions should be generated with different precision for different places.

Although natural language is widely used in human-robot interaction, we realize that there are still no works on the narration of humanoid robot experience with natural language. Our research focuses on generating narration of humanoid kitchen robots' past experience, which provides a possibility for the user to check what happened when the robot worked alone. We propose an approach to generate episodic memory for humanoid robots so that users can query the robots' past experience. In addition, by making use of the episodic memory component, our verbalization system is able to generate narrations of both high-level symbolic plans and low-level sensor-motor events.

III. Episodic Memory Generation

In order to narrate past experience, robots should be capable of memorizing and recalling significant past behaviors. For the purpose of memorization, we create a model for robot behaviors which uses the execution plan as well as the sensorimotor perception data as input. Additionally, we also develop a module to encode and generate an episodic memory for platform and end-effector related events considering the type of robot tasks.

The episodic memory is a memory system that stores information on autobiographical episodes or events, as well as related time-indexes and spatial relations [13]. Such a memory implemented in the robot system supports recording or retrieving personal experience in a specific place and a period of time. In [14] and [15], the episodic memory is applied to a multi-agent cognitive robot system and human-like robots. In order to achieve our ultimate goal, which is to enable a humanoid robot to narrate its past experience, we firstly have the robot generate episodic memory through a *Log Files Generator*. In the next paragraphs, we discuss how to encode episodic memory items and when to generate a memory record.

A. Modeling of the Robot Experience

The episodic memory records the history of events and tasks of the robot. Therefore, the encoding of memory items is based on the experience of the robots that can be demonstrated in various semantic granularity. For instance, when the robot describes what happened over a period of time in the past, it can either roughly tell people the activity, such as "prepare salad", or explain a sequence of actions in detail, such as "taking out salad from the fridge, put salad on the plate...". We propose a layer-structured model to represent robots' experiences which consist of three levels of abstractions:

Task: We define *task* as a representation of experience at the high semantic level. Different robots are capable to execute various duties $t \in T$. The set T represents a domain of tasks. In the case of household robots, we consider object manipulation tasks as well as navigation tasks. The type of task is inferred from a sequence of robot motions.

Action: A sequence of actions is a plan resulting from a symbolic planner, such as PKS [16] as used in [4]. Actions are defined with the planning domain definition language (PDDL) [17]. In our case, we represent robots' experiences with a simplified action definition (N, L), where N is the action name and L is the parameter list which contains agent name, hand identifier, agent location and object name. For

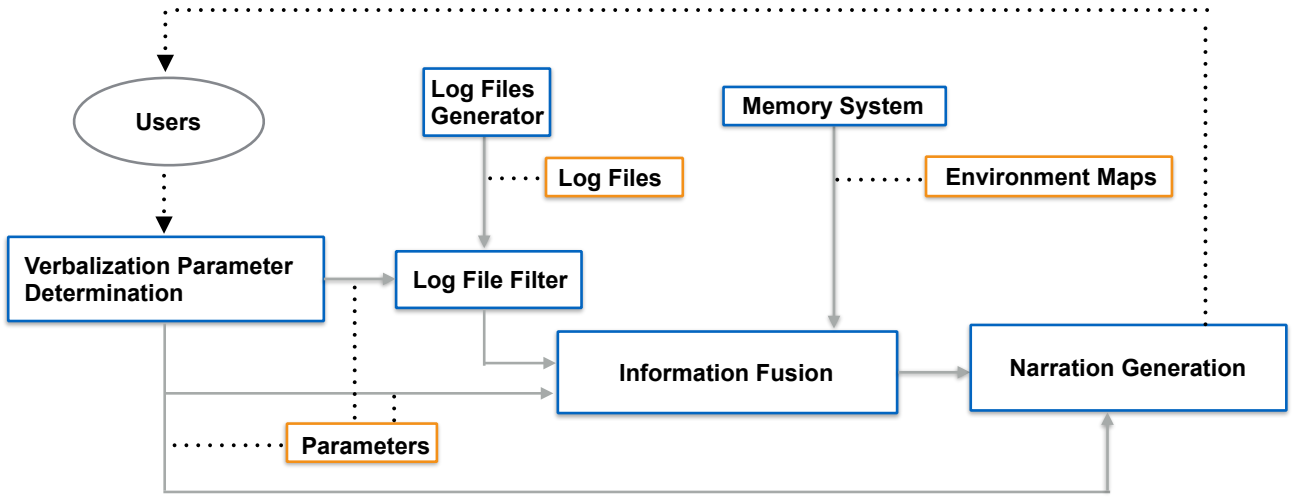

Fig. 2. The pipeline of verbalization system consists of six primary function modules which are shown in blue blocks. These blocks are organized in order. Another three orange blocks that between two function modules mean outputs or data that processed by function modules.

example, the following is a PDDL representation of grasping action:

```
(:action grasp
  :parameters (?a : obj_agent_robot,
  ?h : obj_hand, ?l : loc_surface,
  ?o : obj_graspable))
```

Primitive Action: Primitive actions are elementary basic actions or events that constitute a motion. Besides, primitive actions are the most low-level representation of robots' experiences which are directly detected in raw sensorimotor data. In our case, we distinguish two types of primitive actions:

- *Platform* related primitive action: Platform primitive actions are segments of the whole navigation route, which can be represented by $(P_{\mathrm{i}}, P_{\mathrm{j}}, E_{\mathrm{ij}})$. The P_{i} is the start pose and P_{j} is the stop pose. E_{ij} is the direct path from point i to point j.
- *Tool Center Point (TCP)* related primitive action: TCP primitive actions contain TCP primitive movements and grasping/releasing events. TCP primitive movements are segments of trajectory which has the form $(P_{\mathrm{i}}, P_{\mathrm{j}})$, where P_{i} and P_{j} are points on the trajectory and P_{i} is accessed earlier than P_{j}.

B. Encoding of Episodic Memory Items

Based on the representation of robot behaviors, episodic memory items are divided into four types. The first type of memory item is related to the robot's experience which is represented on the *action level*. The other three types of memory items are related to the robot's experience which is represented on the *primitive action level*. We encode different types of memory items by defining formats of log file entries separately. The first data field is reserved to indicate the record type and each record has at least one timestamp. We list the format definition as follows:

- In our robot system, the domain of actions is constrained to *grasp, put-down, move, open, close, pour, shift-right, shift-left, stir,* and *wipe*. However, developers can scale the action set as wanted. Corresponding to the action definition, items in the episodic memory are length varying from action to action because of varied parameters. We set up a look-up table to indicate data fields' meaning of the action parameter list. The record item of a robot action is formalized in a tuple as

$$I_{\mathrm{action}} = \langle ID,\ Name,\ List_{\mathrm{para}},\ T_{\mathrm{start}},\ T_{\mathrm{stop}} \rangle$$

  where $ID =$ "*action*" and *Name* is the name of action which is defined in the action domain. $List_{\mathrm{para}}$ is the parameter list of the action which is defined through the symbolic planning system. T_{start} and T_{stop} are the start and stop timestamps.
- A record item of platform primitive action is defined as

$$I_{\mathrm{platform}} = \langle ID,\ Pose_{\mathrm{start}},\ Pose_{\mathrm{stop}},\ T_{\mathrm{start}},\ T_{\mathrm{stop}} \rangle$$

  where $ID =$ "$platform$", $Pose_{\mathrm{start}}$ and $Pose_{\mathrm{stop}}$ are start pose and stop pose. An instance of pose is represented by (x, y, θ) where (x, y) are coordinates in the two-dimensional space and θ is the orientation angle. Both are given in the Cartesian world coordinate system. T_{start} and T_{stop} are the start and stop timestamps for the primitive action.
- Similar to the record of the platform, the record item of a TCP primitive action is formalized as

$$\begin{aligned} I_{\mathrm{end\_effector}} = \langle & ID,\ Range,\ HandID,\ Position_{\mathrm{start\_world}}, \\ & Position_{\mathrm{start\_base}},\ Position_{\mathrm{stop\_world}}, \\ & Position_{\mathrm{stop\_base}},\ T_{\mathrm{start}},\ T_{\mathrm{stop}} \rangle \end{aligned}$$

  where $ID =$ "$endEffector$" and *Range* specifies the range of TCP's movement. End-effectors manipulate objects when the platform does not move, so the system

can infer the *Range* through the position of the platform. *HandID* indicates if the TCP is the left hand or the right hand. *Position* is a point $(x,\ y,\ z)$ in the coordinate system which can either be world system or arm base system. The subscript shows the related coordinate system at the start position and end position. T has the same meaning as in platform primitive action records.

- The remaining two TCP actions are grasping and releasing. The memory item is encoded as

$$I_{\text{end\_effector\_grasp}} = \langle ID,\ Object,\ T_{\text{happen}} \rangle$$

The value of *ID* is "*grasp*" or "*release*", and *Object* is manipulated kitchen accessories which are grasped or released. T_{happen} records timestamp as events happened. The record *"grasp vitaliscereal 1490757888 03/28/17 23:24:48.011"* indicates the robot grasped a Vitalis Cereal at the time `03/28/17 23:24:48.011`. The location and hand identifier information can be inferred from other memory items, so we do not encode this information in the memory items of grasping and releasing events in order to avoid redundancy.

C. Trajectory Segmentation and Events Detection

The *Log Files Generator* creates and saves a new entry through segmenting the TCP trajectory or platform route, as well as detecting specific events. First, we focus on events which usually occur in the context of grasping and manipulation. In order to segment trajectory, when the TCP moves the system samples points on the trajectory with constant frequency. The sampling rate is adjusted to the TCP velocity in order to get appropriate segments. We use a Finite-State Automaton (FSA) to monitor TCP related sensor data and detect the events when the TCP starts to move. Then FSA generates a record for the TCP primitive action every time a new point is sampled. For the platform primitive action, the FSA oversees the current target of the mobile base in the low sensor-motor level and generates a record as the target changes as shown in Fig. 3. The current target is set by the path planning module and is an intermediate goal in the whole path, which the robot can directly reach by a straight line. As for hand grasping or releasing events, the FSA uses sensor data to check the curvature of fingers' joint. If the change of curvature is larger than a threshold and has a closing tendency, the grasping action is assumed to have happened, otherwise the releasing action happens. Furthermore, the FSA monitors currently executed steps of the plan and generates a record if the new step begin to be carried out. Our ArmarX [18] robot development environment offers a mechanism called *Statecharts* to implement FSA [19].

IV. Information Fusion and Verbalization

So far, the original data of robot experience has been logged in files. However, the raw data is not intuitive for common users. We fuse environmental information with original information and infer new knowledge from past experience. Finally, the system organizes all sources and generates descriptions following a set of rules.

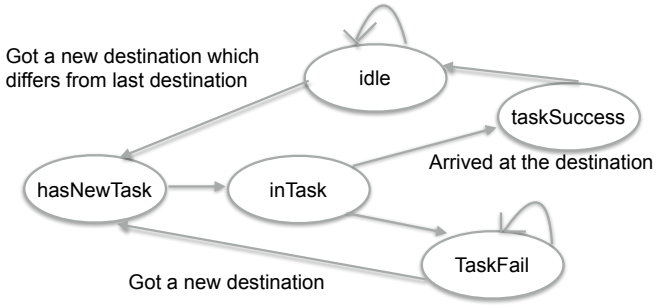

Fig. 3. A memory item generator for platform primitive actions which is based on FSA implemented as ArmarX statecharts.

A. Fusion of Log Files and Maps

For the purpose of describing past primitive actions fluently and naturally, we interpret position information referring to environment knowledge. The location of the mobile robot platform that is represented by coordinates in the log files can be annotated with a nearest prominent facility or manually defined marks , if the distance is below a defined threshold. Considering different sizes of the TCP workspace and the platform movement range, we individually assign appropriate environmental landmarks to the TCP and the platform. Three types of maps are used to represent environment information as follows:

Static Kitchen Map: The static kitchen map indicates prior knowledge about the open kitchen area. This map is a metric map that consists of several points $p = (n,x,y)$, where n is the static kitchen facilities' ID name, and (x,y) is the corresponding coordinate in the kitchen. This prior knowledge is stored in the robot's memory system and is loaded to working memory during the execution time. The data processing of platform related records is based on this map.

Static Marks Map: The static marks map reflects topological relationships between manually defined marks and static kitchen facilities. The map comprises several significant points that on static facilities $p = (m,x,y,z,r)$, where (m,r) indicates that a static mark m is on the related static kitchen facility r and (x,y,z) is the position of the static mark in the world coordinate system. Such a map is helpful to annotate TCP localization constrained in a limited range.

Dynamic Objects Map: The dynamic objects map represents the world states which are the locations of kitchen accessories. These locations may vary because robots can manipulate the objects. We formalize points in the dynamic objects map as $p = (a,r,x,y,z,c)$, where (a,r) stands for the kitchen accessory a is on or in the kitchen facility r, and c is the characteristic of the kitchen accessory, for instance, the flavor or color. The (x,y,z) means the accessory's position. Besides, the dynamic kitchen accessories map has different instances during the task execution. We formalize this aspect by $M = (m,t)$ where m is the map ID and t is the timestamp.

Our verbalization system is extendable because of these maps. These maps enable the robot to work in others kitchen environment. In order to annotate coordinates with environment reference, we use two strategies. One is to find the nearest neighbor on the map based on the Euclidean distance. The other is to take orientation into account and find the object at which the robot is looking.

- The Euclidean distance based nearest neighbor is the neighbor, which has minimum Euclidean distance to a given coordinate. This criterion is helpful to fuse robot positions or end-effector positions in the log files with the static kitchen map or the static marks map. For example, if the mobile platform moved from `(3409.55,7100.15,-1.56999)` to `(2932.31,5618.54,2.3293)`, the movement can be represented in another form as "The robot moved from a point near sideboard, then arrived at a point near the control table".

- The angular distance based nearest neighbor is crucial for annotating the moving direction of TCP primitive action with a static mark. The primitive action can be represented as a vector as given in equation (1). We calculate another vector that demonstrates the relationship between primitive action and the object as given in equation (2). The $(x_{\text{start}}, y_{\text{start}})$ and $(x_{\text{stop}}, y_{\text{stop}})$ are the start and stop point of the TCP, while $(x_{\text{o}}, y_{\text{o}})$ is the location of the mark. The angle distance between TCP primitive actions and the object is the angle $\theta\langle\vec{r}_{\text{tcp}}, \vec{r}_{\text{tcp\_o}}\rangle$. For instance, we can interpret the tendency of TCP primitive movement as "the TCP moved to the table center", where the table center is the nearest static mark.
  Although the end-effectors move in the three-dimensional space, we reduce the movement onto the surface where manipulated objects are, namely x and y axes.

$$\vec{r}_{\text{tcp}} = \left[x_{\text{end}} - x_{\text{start}}, y_{\text{end}} - y_{\text{start}}\right]^{\text{T}} \quad (1)$$

$$\vec{r}_{\text{tcp\_obj}} = \left[x_{\text{obj}} - x_{\text{start}}, y_{\text{obj}} - y_{\text{start}}\right]^{\text{T}} \quad (2)$$

$$D_{\text{Angular}} = \theta\langle\vec{r}_{\text{tcp}}, \vec{r}_{\text{tcp\_obj}}\rangle = \arccos \frac{\vec{r}_{\text{tcp}} \cdot \vec{r}_{\text{tcp\_obj}}}{\mid \vec{r}_{\text{tcp}} \mid\mid \vec{r}_{\text{tcp\_obj}} \mid} \quad (3)$$

B. Narration Generation

The *Narration Generation* module maintains a multiple templates and uses a template-based approach to generate descriptions of past robots' behavior. The information is directly extracted or indirectly mined from log files. The newly mined information contains execution distance, execution time, velocity and more. Besides, new interpretation or knowledge is generated through information fusion as described above. After generating episodic memory entities and fusing log files with environmental information, the system is finally capable to convert low level sensorimotor data as well as high level plans into natural language descriptions by choosing the appropriate information and filling them in the templates slots.

TABLE I
NARRATED INFORMATION OF TCP RELATED EXPERIENCE UNDER A SPECIFIED VERBALIZATION PARAMETER

		Abstraction		
		Level 1	Level 2	Level 3
Specificity	Level 1	start point coordinates, stop point coordinates, execution time	nearest static marks or the nearest kitchen accessories of start and stop points, execution time	moving tendency, execution time
	Level 2	start point coordinates, stop point coordinates, velocity, moving distance, hand identifier	nearest static marks or the nearest kitchen accessories of start and stop points, velocity, moving distance, hand identifier, object characteristic	moving tendency, velocity, moving distance, hand identifier

1) Verbalization Space: The robot can already simply narrate past experience by using function modules described above, if user inclination is ignored. However, the robot should consider various user preferences and interests. For example, our robot researchers are interested in detailed and precise narration while common users care about a concise description. Therefore, we use the verbalization space to quantify the user preference. The space consists of three orthogonal parameters (E, A, S), where E stands for the semantic level that specifies the representation of the robot's behavior, A is a parameter that determines the abstraction degree of description, and S is the specificity that defines the information amount of description. Once the verbalization parameters are determined, the system filters out log files of interest, generates useful information as wanted, and chooses corresponding templates to generate utterances.

The kitchen robot conducts tasks mainly through coordinated motion of the mobile platform and the two end-effectors at sensorimotor level, which has different characteristics. In order to generate proper interpretation of past experience, we respectively define ranges of *specificity* and *abstraction* for the platform and the end-effector. Table I shows how to instantiate verbalizations for TCP related experience under specified parameters.

Semantic E: The semantic level indicates which level should be used to represent robot behaviors. At the high semantic levels 1 and 2, the system generates descriptions based on the task and the executed plan which represent the robot experience. On the contrary, level 3 shows the sensorimotor related primitive actions. *Abstraction* and *Specificity* works when semantic level is 3.

Abstraction A: Abstraction decides how to represent the primitive action and how to add prior knowledge of the environment into the interpretation. This parameter also determines the corpus for the description generation. Level 1 is the most concrete level, the system gives out world coordinates of the start point and stop point of an action. For the platform, the position is interpreted by the nearest static

kitchen facilities in level 2. In level 3, the narration only gives out a rough region of moving. For the end-effector, the start and stop point of a TCP primitive actions are encoded with static marks or manipulable kitchen accessories in level 2. In abstraction level 3, the verbalization explains the moving tendency of TCP such as "leftwards", "upwards".

Specificity S: Specificity quantifies the information amount. Level 1 provides the least information that only contains execution time. For platform, level 2 and level 3 add more information to the description. Level 2 adds information about turn angle, moving distance, and movement speed. Level 3 appends the information about the facing direction at the start and stop point. As for the end-effector, level 2 complements several information such as velocity and moving distance.

2) Variable Verbalization Algorithm: The complete variable verbalization algorithm is given in Algorithm 1 in pseudocode. The algorithm accepts a user query as input and synthesizes natural language description of past experience. The user query includes time and objects of interest, as well as the values of verbalization parameters. When the system determines the user preference, entries in log files are selected according to the user interest. Afterwards, the system fuses filtered records with environmental information and generates a variety of new information. Finally, the system generates narration by filling in the corresponding templates with proper information.

Algorithm 1 Algorithm for Verbalization

Input: *query_user*
Output: *narrative*

// Record the history of robot's behavior
1: $logFiles \leftarrow$ generateLogfile($plan\_executed, sensorData$)
// Determine values of verbalization parameters
2: $(E, A, S) \leftarrow$ getParameterValues($query\_user$)
// Determine time and objects of interst
3: $(time, object) \leftarrow$ getInterest($query\_user$)
// Filter log files according to users' interest
4: $logFilesFiltered \leftarrow$ filterLogfile($E, time, object, logFiles$)
// Load maps from memory system
5: $maps \leftarrow$ loadMaps($workingMemory$)
// Annotate locations with conspicuous marks
6: $infoPool \leftarrow$ fuseInformation($maps, logFiles$)
// Choose appropriate corpus
7: $corpus \leftarrow$ getTemplates(E, A, S)
// Generate narration for past experience
8: $narrative \leftarrow$ generateNarration($infoPool, logFiles, E, A, S$)

V. Evaluation And Application

We evaluated the proposed system in context of the humanoid robot ARMAR-III [2] with two 7-DoF arms, two pneumatic five finger hands, a 7-DoF head with two stereo camera systems (foveal and peripheral), and a holonomic platform with 2D laser scanners. ARMAR-III is able to perform tasks in a kitchen environment such as grasping objects, opening the fridge, pouring, mixing, wiping the table. We integrated our verbalization system in the robot system and evaluated the system in the context of a pick-and-place task as shown in Fig. 4. The robot system allows to use a simulation or the real robot without needing to change anything in high-level components like the proposed verbalization system. The proposed verbalization system is evaluated with the kinematic simulation of ArmarX. This allows robust execution of the manipulation skills without any failures during execution due to inaccuracies in the execution or perception.

Our kitchen environment consists of basic kitchen facilities such as oven, fridge, sink, dish washer, table, etc. Several static marks are manually defined beforehand, e.g. table center, table corner, fridge handle, etc., relating to static kitchen facilities. The following section shows segments of the static kitchen and marks map. Each map occupies a segment in the memory system of the robot. In the static kitchen map, the coordinates show the positions of the facilities in the kitchen map. In the static marks map, the coordinates show the positions of the marks in three-dimension. We test our verbalization system in a pick-and-place scenario taking place in this kitchen environment.

End users usually have a different preference when they query the past experience of the robot. Some users want to get a natural and detailed narration while others prefer concise narration. Therefore, we demonstrate two typical use cases using the following two examples.

$$\left\{\begin{array}{ll} \ldots & \\ (static\ kitchen\ map) & \\ dishwasher & (2500, 9000) \\ fridge & (4050, 9260) \\ sideboard & (3400, 7000) \\ sink & (2600, 9500) \\ stove & (3110, 9500) \\ \ldots & \end{array}\right.$$

$$\left\{\begin{array}{lll} \ldots & & \\ (static\ marks\ map) & & \\ tableCenter & (4300, 7165, 1032) & sideboard \\ tableCorner1 & (3800, 6540, 1032) & sideboard \\ refrigeratorHandle & (4150, 9360, 1100) & fridge \\ \ldots & & \end{array}\right.$$

Example 1: Natural and Detailed Verbalization

The user wants to get natural detailed information about certain events that happened in a specific day. To represent this user preference we select level 3 for *semantic*, to get sensorimotor related narration. To get the natural narration of the TCP and platform primitive actions, we select level 2 for *abstraction* and for the detailed information of the TCP and platform primitive actions, we select level 2 for *specificity*. Afterwards, the system fills in the provided templates, which cover all possible actions. The following shows all the templates for the verbalization parameters we have previously assigned:

"$[\text{I}]_N$ $[\text{started}]_V$ in the vicinity of $[\text{---}]_{\text{kitchenFacilityNearStartPose}}$ and arrived around the vicinity of $[\text{---}]_{\text{objectNearEndPose}}$"
"$[\text{I}]_N$ $[\text{moved}]_V$ around $[\text{---}]_{\text{NearestKitchenFacility}}$"

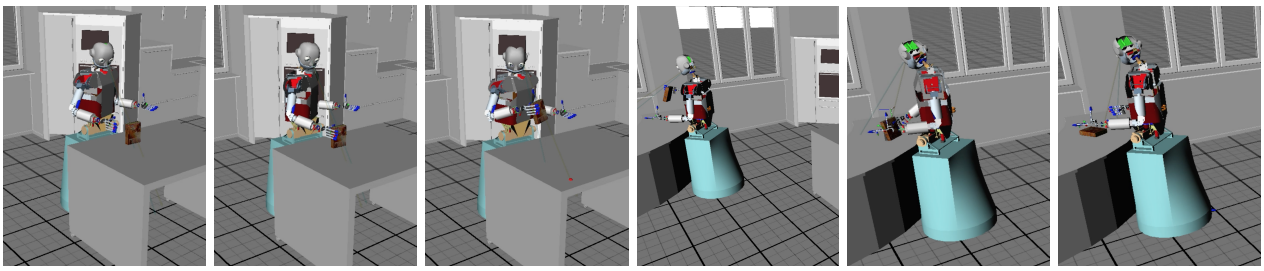

Fig. 4. A sequence of images demonstrates that the robot executes a pick and place task in the kitchen. The robot (1) started in front of the sideboard and moved right hand towards a box of cereal, (2) grasped the box, (3) raised right hand, (4) moved from the sideboard to the control table, (5) put down the box, (6) released the hand.

"$[\text{I}]_N$ $[\text{moved}]_V$ a total of $[\text{---}]_{\text{distance}}$ meter with a speed of $[\text{---}]_{\text{velocity}}$ m/s and $[\text{rotated}]_V$ $[\text{---}]_{\text{rotationAngle}}$ degrees with a speed of $[\text{---}]_{\text{rotateVelocity}}$ degree/s which $[\text{took}]_V$ $[\text{---}]_{\text{timeToCompleteTask}}$ seconds"
"$[\text{I}]_N$ $[\text{moved}]_V$ my $[\text{---}]_{\text{handIdentifier}}$ hand from $[\text{---}]_{\text{staticMark}}$ to $[\text{---}]_{\text{staticMark}}$ for a total of $[\text{---}]_{\text{distance}}$ mm with a speed of $[\text{---}]_{\text{velocity}}$ mm/s"
"$[\text{I}]_N$ $[\text{grasped}]_V$ $[\text{---}]_{\text{object}}$."

The system selects memory records based on the time that users are interested in. Records which were recorded at the same date given by the user will be selected. Afterwards, the system generates a corpus based on the assigned *abstraction* level, e.g. the nearest landmark, execution time in "seconds", velocity in "mm/s", and rotation angle in "degree". Finally, the system replaces the placeholders in the templates using the information from the corpus, and generates utterances for each log file entry.

```
I moved my right hand from tableCenter to
    tableCorner1 for a total of 44.345375
    mm with a speed of 8.869075 mm/s.
I moved my right hand around tableCorner1
    for a total of 92.628616 mm with a
    speed of 18.525723 mm/s.
...
I grasped vitaliscereal.
I moved my right hand from tableCenter to
    tableCorner1 for a total of 99.198570
    mm with a speed of 19.839714 mm/s.
I started in the vicinity of sideboard and
     arrived around the vicinity of
    controltable. I moved a total of
    1.556575 meter with a speed 1.556575 m/
    s and rotated 223.412857 degrees with a
     speed 223.412857 degree/s which took
    1.000000 seconds.
...
I released vitaliscereal.
I started in the vicinity of controltable
    and arrived around the vicinity of
    sideboard. I moved a total of 0.378236
    meter with a speed 0.378236 m/s and
    rotated 0.048699 degrees with a speed
    0.048699 degree/s which took 1.000000
    seconds.
```

Example 2: Concise Verbalization

In contrast to the previous example, the system can also produce concise verbalization. The user wants to know the experience of ARMAR-III on a different day and the explanation has to be concise. The semantic level therefore has to be set to level 2. To get the narration, first the system checks the look-up table to decode action related memory items. Afterwards, the system finds matching templates and use them to generate narrative as follows:

```
I moved from sideboard2 to the sink. It
    took 1 seconds.
I picked up green cup on the sink with my
    right hand. It took 4 seconds.
I moved from sink to placesetting1. It
    took 1 seconds.
I put down the green cup. It took 1
    seconds.
```

We conducted our testing and evaluation of the verbalization system on scenarios in a simulator. The system is capable of generating matching narratives according to user preferences. Preferences with a high level of specificity result in narrations with more words, and preferences with a high level of semantic result in more detailed narrations. Additionally, the higher the level of abstraction, the more human-like the narration becomes. Besides, the system also generates correct descriptions for other graspable objects in a pick-and-place scenarios.

The proposed verbalization system is also highly extensible. To use our verbalization system in a different environment, users only need to adjust the static marks map, static kitchen map and templates. To use the verbalization system with a different robot, the robot needs to provide information about the position of the hands, the position of the robot in the world, and the perceived objects. If the robot uses also the ArmarX framework, only configuration adjustments are needed.

VI. Conclusion

In this work, we presented a verbalization pipeline as a method for mapping the experience of a humanoid household robot into natural language. The verbalization system allows the robot to store past experiences and to generate

explanations of those experiences according to the given user preferences regrading abstraction and specificity. The produced explanations are not limited to sensorimotor related experiences only, but also include high level experience, e.g. the execution plan.

Our future work will focus on automatically determining user preferences from dialogs between the robot and the user. Additionally, we will also devise a method for the system to learn templates by processing a natural language corpus in order to refine our manually defined templates.

REFERENCES

[1] S. Rosenthal, S. P. Selvaraj, and M. Veloso, "Verbalization: Narration of autonomous robot experience," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. AAAI Press, 2016, pp. 862–868.

[2] T. Asfour, P. Azad, N. Vahrenkamp, K. Regenstein, A. Bierbaum, K. Welke, J. Schroeder, and R. Dillmann, "Toward humanoid manipulation in human-centred environments," *Robotics and Autonomous Systems*, vol. 56, no. 1, pp. 54–65, 2008.

[3] T. Asfour, K. Regenstein, P. Azad, J. Schröder, N. Vahrenkamp, and R. Dillmann, "ARMAR-III: An Integrated Humanoid Platform for Sensory-Motor Control," in *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, Genova, Italy, December 2006, pp. 169–175.

[4] E. Ovchinnikova, M. Wächter, V. Wittenbeck, and T. Asfour, "Multi-purpose natural language understanding linked to sensorimotor experience in humanoid robots," in *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, Seoul, Korea, 2015, pp. 365–372.

[5] T. Kim and P. Hinds, "Who should i blame? effects of autonomy and transparency on attributions in human-robot interaction," in *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*. IEEE, 2006, pp. 80–85.

[6] V. Perera, S. P. Selveraj, S. Rosenthal, and M. Veloso, "Dynamic generation and refinement of robot verbalization," in *Robot and Human Interactive Communication (RO-MAN), 2016 25th IEEE International Symposium on*. IEEE, 2016, pp. 212–218.

[7] D. Voelz, E. André, G. Herzog, and T. Rist, "Rocco: A robocup soccer commentator system," *RoboCup-98: Robot Soccer World Cup II*, pp. 50–60, 1999.

[8] M. Veloso, N. Armstrong-Crews, S. Chernova, E. Crawford, C. McMillen, M. Roth, D. Vail, and S. Zickler, "A team of humanoid game commentators," *International Journal of Humanoid Robotics*, vol. 5, no. 03, pp. 457–480, 2008.

[9] E. André, K. Binsted, K. Tanaka-Ishii, S. Luke, G. Herzog, and T. Rist, "Three robocup simulation league commentator systems," *AI Magazine*, vol. 21, no. 1, p. 57, 2000.

[10] R. Belvin, R. Burns, and C. Hein, "Development of the hrl route navigation dialogue system," in *Proceedings of the first international conference on Human language technology research*. Association for Computational Linguistics, 2001, pp. 1–5.

[11] R. Dale, S. Geldof, and J.-P. Prost, "Using natural language generation in automatic route," *Journal of Research and practice in Information Technology*, vol. 37, no. 1, p. 89, 2005.

[12] D. Bohus, C. W. Saw, and E. Horvitz, "Directions robot: In-the-wild experiences and lessons learned," in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 637–644.

[13] E. Tulving *et al.*, "Episodic and semantic memory," *Organization of memory*, vol. 1, pp. 381–403, 1972.

[14] Z. Kasap and N. Magnenat-Thalmann, "Towards episodic memory-based long-term affective interaction with a human-like robot," in *RO-MAN, 2010 IEEE*. IEEE, 2010, pp. 452–457.

[15] W. Dodd and R. Gutierrez, "The role of episodic memory and emotion in a cognitive robot," in *Robot and Human Interactive Communication, 2005. ROMAN 2005. IEEE International Workshop on*. IEEE, 2005, pp. 692–697.

[16] R. P. A. Petrick and F. Bacchus, "PKS: Knowledge-based planning with incomplete information and sensing," in *Proceedings of the System Demonstration session at ICAPS 2004*, Jun. 2004, http://www-rcf.usc.edu/

[17] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins, "Pddl-the planning domain definition language," 1998.

[18] N. Vahrenkamp, M. Wächter, M. Kröhnert, K. Welke, and T. Asfour, "The robot software framework armarx," *Information Technology*, vol. 57, no. 2, pp. 99–111, 2015.

[19] M. Wächter, S. Ottenhaus, M. Kröhnert, N. Vahrenkamp, and T. Asfour, "The armarx statechart concept: Graphical programming of robot behaviour," *Frontiers in Robotics and AI*, vol. 3, no. 33, pp. 0–0, 2016.