The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Programming by demonstration of probabilistic decision making on a multi-modal service robot

Sven R. Schmidt-Rohr, Martin Lösch, Rainer Jäkel, Rüdiger Dillmann

Abstract— **In this paper we propose a process which is able to generate abstract service robot mission representations, utilized during execution for autonomous, probabilistic decision making, by observing human demonstrations. The observation process is based on the same perceptive components as used by the robot during execution, recording dialog between humans, human motion as well as objects poses. This leads to a natural, practical learning process, avoiding extra demonstration centers or kinesthetic teaching. By generating mission models for probabilistic decision making as *Partially observable Markov decision processes*, the robot is able to deal with uncertain and dynamic environments, as encountered in real world settings during execution. Service robot missions in a cafeteria setting, including the modalities of mobility, natural human-robot interaction and object grasping, have been learned and executed by this system.**

I. Introduction

Domestic service robots have to perform missions in complex, dynamic environments autonomously. Such a mission, e.g. a waiter duty, consists of many tasks, constraints and utilization of different skills. While a mission in a dynamic environment is not a fixed sequence of tasks, these missions have certain structural properties. They describe certain environment states, how the environment is usually changed when a specific action is performed in a certain state and which states are desirable to reach for the robot.

A powerful concept to represent and to perform autonomous decision making, while considering both the dynamic and stochastic nature of the course of events as well as limited perception of robots in real world environments, are *Partially observable Markov decision processes* (POMDPs).

Policies, representing a decision plan for all possible courses of events in a mission, can be computed with different techniques. The model-free approach calculates the policy by reinforcement learning, using a large number of trials. This is infeasible for abstract, high-level missions on a multi-modal service robot. The model-based approach calculates the policy efficiently from a symbolic-numeric model of the mission.

Yet, the question remains how to obtain the explicit model. This paper presents an approach and system to utilize *Programming by Demonstration* (PbD) to let the robot obtain and then execute a model of reasonable size for a mission.

First, closely related work is discussed. Then, the multi-modal robotic platform and decision system are shortly presented. Subsequently, the PbD approach is described, followed by experimental evaluation on the real system.

Institute for Anthropomatics (IFA), Karlsruhe Institute of Technology, Germany Email:{srsr|loesch|jaekel|dillmann} @ira.uka.de

II. Related work

Decision making considering uncertainty in observation and environment dynamics has been investigated in AI and robotics, leading to powerful, general probabilistic techniques. In a probabilistic decision making framework, a rational agent reasons in the presence of uncertainty, either concerning the perception of the environment, the cause of events or both. Discrete *Partially observable Markov decision processes* (POMDPs) consider both and are an abstract model for planning under uncertainty [1], [2]. Under the assumption of a discrete POMDP. the course of events is discrete as well as the set of states, representing possible configurations of the environment. A specific POMDP (mission) model is formed by an 8-tupel $(S, A, M, T, R, O, \gamma, b_0)$. S is a set of discrete states, A a set of actions, the agent can perform and M is a set of measurements, the agent can perceive. The stochastic environment causality is described by the transition model T. A single transition $T(s', a, s)$ models the probability of a transition between states from s to s' when the agent performed action a. The observation model O describes imperfect perception where $O(m, s)$ models the probability of a measurement m when the true world state is s. The reward model R defines the motivations of an agent, giving a numeric reward $R(s, a)$ to the agent when performing action a while the true world state is s. Possible future rewards are discounted by the parameter γ. The initial belief of the agent is set to b_0. In a POMDP, the agent has no knowledge about the true state of the world s_t, but only an indirect belief b_t, a probability distribution over all states in the model. During execution, the agent updates the belief by Bayesian forward-filtering. For the decision, the agent queries a policy with its current belief distribution b_t and receives in turn a most favorable action, concerning expected long-term rewards. In the model-based approach, the policy is calculated from the model, balancing the probabilities of the course of events with the accumulated reward which has to be maximized. While computing exact, optimal policies is intractable [3], recent investigations into approximate solutions have made good progress. State of the art policy calculation algorithms as PBVI [4], HSVI [5] and SARSOP [6] produce good policies for models with many states and slightly complex transition structure (reachable belief) as present in realistic service robot mission models.

With the advent of these algorithms, POMDP decision making has be used in several different robotics and multi-modal interfaces setting, e.g. autonomous navigation [7], haptic exploration of objects for grasping [8] and dementia

patient supervision [9]. While for some scenarios, creating POMDP models describing the environment and task sufficiently is quite simple, it is highly complex for other settings, especially abstract missions. Thus, learning POMDP models for more complex robotic domains has been investigated. An interesting approach, called MEDUSA [10], refines the transition model T and observation model O during execution time by querying an "oracle" – usually an interacting human – about the true state of the world. While improving T to reflect the real environment better, this way, it depends on an initial set of states as well as actions and needs quite a high number of oracle queries to converge. Another related and more general approach is Bayes-Adaptive-POMDP [11] which integrates online learning and planning more tightly, however it is still restricted to small lookahead horizons and simple missions.

On the other hand, *Programming by Demonstration* (PbD) as been applied successfully to an MDP-style framework of movement execution on a small humanoid robot [12]. While this is kinesthetic teaching on a single, specific modality, some aspects of generating forward models relate to the work, presented here. Another PbD investigation has developed a framework for learning pick-and-place operations from user demonstrations in a recording center [13]. While it concentrates only on manipulation and no probabilistic representations are utilized, there are some similarities in the way states and actions are segmented as well as the utilization of multiple demonstrations. Another PbD approach, related to the presented work concerning the level of abstraction of learning whole robot missions has been investigated in [14].

In the following, an approach for acquiring POMDP mission models by means of PbD, supported by a knowledge base, common for all missions, is introduced.

III. System overview

A short overview presents the perceptive components.

A. Skills

On the evaluation platform (see fig. 1), several components provide perceptive and execution capabilities, referred to as *skills*, representative for a typical service robot.

Mobility is provided by a wheeled base, equipped with a laser range-finder and a differential drive system. Self-localization on an indoor map, using Bayesian updates, provides a trivariate normal distribution, indicating current pose and uncertainty: $\vec{\mu} = (x, y, \theta)$, covariance Σ. Navigation moves the platform to a desired target position.

Natural human-robot-interaction (HRI) is provided by two modalities: spoken dialog and body configuration. Speech recognition uses an onboard microphone and the Sphinx4 [15] speech recognition engine, extended to deliver discrete probability distributions, indicating likely human utterances: $p(utter_1), ..., p(utter_n)$.

Markerless human body tracking and recognition of symbolic human activities is realized by using an onboard *Swissranger R3000* 3D-time-of-flight camera, supported by onboard color cameras, which are used by the human body tracking system *VooDoo*. Relevance criteria select features of the body model configuration which are used to label symbolic human activities [16]. Each likely activity is labeled with a certain probability, thus this skill delivers a set of discrete probabilities over known symbolic activities: $p(act_1), ..., p(act_n)$ On the action side, speech synthesis enables robot utterances and arm-hand movements enable robot gestures.

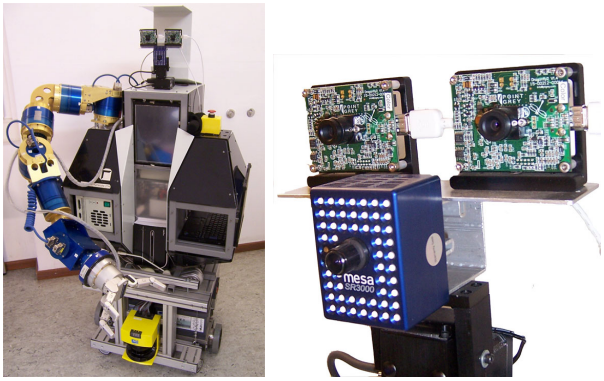

Fig. 1. The multi-modal service robot Albert (left) and its vision system with stereo color camera, 3D-time-of-flight sensor and pan-tilt-unit (right).

Autonomous manipulation of objects as the third domain is provided by a compound of two components. Object localization through onboard stereo color cameras combines a global-appearance based approach and a model-based approach for the 6d recognition and localization of solid-colored objects in real-time [17]. Information about perceptive uncertainty is provided for both type and location of an object. For each object candidate, the likelihood of the object belonging to a known type is given: $p(type_1), ..., p(type_n)$, while for the location, a covariance Σ is provided.

Concerning manipulation execution, motion planning uses rapidly-exploring random trees (RRT) for arm movement and a pre-shape based grasp planner for grasping.

B. Autonomous decision making

A centralized decision making system parametrizes and coordinates skills to enable autonomous behavior of the robot while considering perceptive and action effect uncertainty. This system is portable, as it treats a multi-modal robot as a collection of skills. Thus, the presented evaluation robot platform is just one possible example of application.

The decision making and task execution system include layer two and three in a typical three layer architecture, where the skills represent layer one (see fig. 2). Layer two contains a component for filtering and fusion of the data received from the perceptive skills as well as a component for supervising the execution of chosen tasks.

The filter collects the measurement distributions of all skills and performs Bayesian updates on those, where none is performed in the skill – e.g. for self-localization it is done in the skill component, for dialog it is not. Continuous distributions are discretized, e.g. self-localization, and then fused into a single belief state. In layer three, the decision

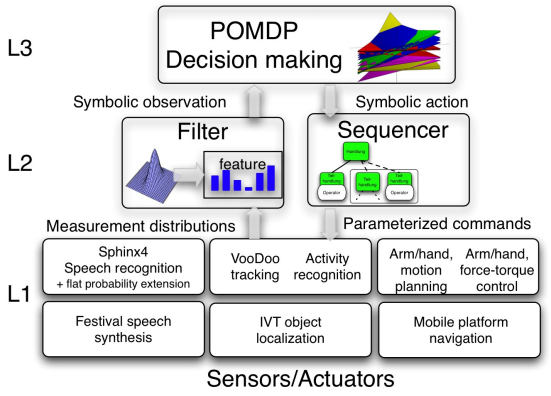

Fig. 2. A rough schematic view of the autonomous execution architecture.

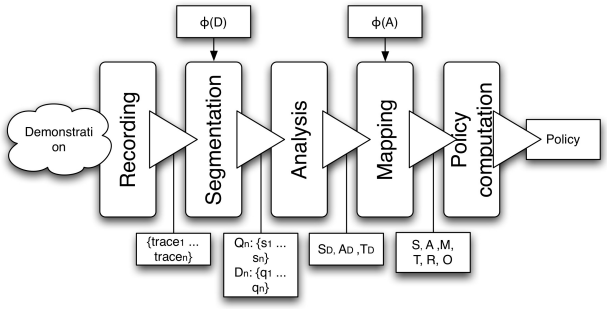

Fig. 3. A rough schematic view of the PbD process.

making process queries a policy, pre-calculated from the POMDP mission model, with the belief and retrieves a decision about the next symbolic action. This action, represented by a Hierarchical Task Network (HTN) coordinating skill execution, is selected when the previous action (HTN) has terminated [18].

IV. Learning process

In state of the art POMDP applications, usually one specific, hand tailored and still simple model is utilized for a certain niche application. The arising question for the described system is how to obtain sufficiently descriptive POMDP mission models for arbitrary, multi-modal service robot missions.

Here, we present a solution utilizing a mixture of observations of natural demonstrations by humans and fixed background knowledge, shared for all missions. Background knowledge contains characteristics of the skills on the specific robot platform which cannot be deduced from human demonstrations. However, information about the structure of the mission as well as typical action results can be derived by observing and analyzing human demonstrations of missions as humans have profound, often implicit, domain knowledge.

The PbD process consists of the following steps: *recording*, *segmentation*, *analysis*, *mapping* and finally *policy computation* as shown in fig. 3 and described in the following sections.

A. Observation process

During the demonstration process, the robot observes the demonstrating human representing the robot – *robot actor*, (roAc) – and the scene, which may include a human representing an interaction partner – *human actor* (huAc) – with its sensors and perceptive skills as described in sec. III. On the evaluation platform, human body tracking is used to determine position and orientation of both humans relative to the robot. Object localization retrieves the pose of objects relative to the robot. Self localization is used to translate relative human and object poses into absolute positions and orientations in the scene. Human activity recognition delivers labeled human activities. Speech recognition does not use the onboard microphone as during execution, because two speakers cannot be distinguished. Instead, headsets are used for each roAc and huAc. During demonstrations, the observation is assumed fully observable, as using headsets and performing accentuated movements leads to sufficiently robust recordings. For skills k delivering discrete distributions, the maximum likely perception: $c_k = max\ p_i(c_k)$ is assumed, for continuous distributions, variance is dismissed.

During the first step of the PbD process, *recording*, the robot follows the *robot actor* actively with its neck (pan-tilt unit) to keep the human and the surrounding scene always in its view. Each time, there is a change – above a noise threshold – in one or several skill perceptions c_k, a data point p is recorded:

$$trace \leftarrow \begin{cases} \emptyset, & \forall k : |c_{k_{t-1}} - c_{k_t}| < \epsilon \\ p_t, & \exists k : |c_{k_{t-1}} - c_{k_t}| >= \epsilon \end{cases} \quad (1)$$

The sequence of all data points p_t of a demonstration forms a *trace* while several demonstrations of a mission lead to a set of traces: $\{trace_1 ... trace_n\}$.

B. Segmentation and Analysis

For further processing, each trace containing numerical data of each perceptive modality is segmented into a sequence of symbolic demonstration states $Q : \{s_1 ... s_n\}$. A potential state s_p of a sequence Q is described as:
$s_p := roAc.Activity \times roAc.Utterance \times roAc.InRegion \times huAc.Activity \times huAc.Utterance \times objectLayout$. A state description mapping is used to assign a state to a recorded datapoint: $s_j = \phi(p_i)$ which utilizes the same discretization as the runtime system (sec. III-B). In case two successive data points are assigned to the same state, those are joint:

$$Q \leftarrow \begin{cases} \emptyset, & \phi(p_{t-1}) = \phi(p_t) \\ s_t = \phi(p_t), & \phi(p_{t-1}) \neq \phi(p_t) \end{cases} \quad (2)$$

After all traces have been seqmented, the first analysis step is to determine the state space S_D of the demonstration:

$$\forall s_i \in Q_1 ... Q_n : s_i \in S_D \quad (3)$$

In the next step, analysis of the segmented demonstrations has to reveal all relevant actions A_D of the mission. Navigation actions can be deduced easily from changes in the robot actor region: $s_{t-1}(roAc.InRegion) \neq s_t(roAc.InRegion) \Rightarrow$

$a_{Goto(s_t(roAc.InRegion))} \rightarrow A_D$. Utterance actions can be directly taken from the utterances of the robot actor: $s_t(roAc.Utterance) \Rightarrow a_{Say(s_t(roAc.Utterance))} \rightarrow A_D$. All other actions have to be derived from observed human body activities which includes gestures and manipulation actions. Gestures can be directly derived: $s_t(roAc.Activity)$ is gesture $\Rightarrow$ $a_{Gesture(s_t(roAc.Activity))} \rightarrow A_D$. Manipulation actions, on the other hand are object specific, yet the activity recognition system just delivers an object independent classification, as e.g. *Pick*, *Handover*. To specify the action more precisely, it is analyzed which object has been modified in the *objectLayout* as an effect of the action: $s_t(roAc.Activity)$ is manipulation, $s_{t-1}(obj, objPosition) \neq s_t(obj, objPosition) \Rightarrow a_{Manip((obj\ with\ s_t(roAc.Activity))} \rightarrow A_D$.

With S_D and A_D available, all transitions T_D are accounted for in the next step, first filling the *Counting* transition model TC_D:

$$\forall Q_1..Q_n : \forall s_t \in Q_i : TC_D(s_{t-1}, a_t, s_t) + 1 \qquad (4)$$

From TC_D, T_D - the forward model - can be obtained by calculating the transition prior probabilities for each state-action pair (s, a) from the observed posteriors:

$$\forall (s, a, s') : T_D(s, a, s') = \frac{TC_D(s, a, s')}{\sum_{s'_i} TC_D(s, a, s'_i)} \qquad (5)$$

Stochastic properties of the world are reflected in T_D, e.g. in a certain mission, most of the time - but not always - a human wants a drink and not a snack first, would be reflected as:

- $T_D(s_{greeting}, a_{say-offer-services}, s'_{bring-tea}) = 0.75$,
- $T_D(s_{greeting}, a_{say-offer-services}, s'_{bring-appetizer}) = 0.25$

Finally, the reward model R_D corresponding to the demonstrations is created, which means deducing key (a, s)-pairs of demonstrated missions, representing goals.

(Sub-)goals in a mission - and an observed segmentation - are the final results of chains of actions with according state changes. The chains of actions between two sub-goals can been seen as an episode with the final action and its result being the key action, state pair (a_k, s_r). Within a single observation, two episodes can be distinguished if an action a_i occurs in a state $s_{t_1} = s_i$ and again in the state $s_{t_2} = s_i$. If there is no such case, it has to be assumed, that only the final configuration is the desired goal, leading to just one episode and thus a_k. With several demonstrations, it is possible to segment episodes by checking across demonstrations. A state s_r might be the result of different $a_1, ..., a_n$ chains in two demonstrations Q_i and followed by differing action chains. This demonstrated "crossing of courses of events" then splits episodes in both demonstrations, leading to a sub-goal.

During execution, however, the autonomous decision making process is not necessarily bound to the demonstrated chain of actions as it may be able to reach the reward with another chain of transitions than the demonstrated one.

The state s_r resulting from a_k in a sequence Q, is assigned a reward $R_D(a_k, s_r)$ calculated from all the generic penalties (see next section) of actions a_i which had to be performed to reach s_r since the last key action: $R_D(s_r, a_k) = \nu \sum_i |R(*, a_i)|$ with $\nu > 1$. By these means, the positive reward of the goal can outweigh the penalty (effort) to reach it, which is important for policy computation. An example of generated information: it is desirable to *Place* a *cup* at *east-table* when the dialog had produced a *"bring me tea"*.

Where the *Cup* can be picked, that it needs to be picked at all before being placed or how the dialog can reach *"bring me tea"*, is not explicitly encoded in the reward model, but instead in the generated transitions.

C. Mapping

The preliminary MDP model (S_D, A_D, T_D, R_D) created by the analysis has to be enriched by background knowledge to complete the POMDP model $(S_E, A_E, M_E, T_E, R_E, O_E)$. At this stage, knowledge about the behavior of the robotic system has to be added, which cannot be acquired from observing human demonstrations, as e.g. navigation glitches, speech recognition error characteristics, etc.

As human demonstrator and executing robot act in the same workspace and the same perceptive components are used during both demonstration and execution, state descriptions are identical and no extra mapping is needed. Yet, states have to be added from background knowledge to account for robot behavior. E.g. navigation glitches might lead to the robot leaving the regions in which the demonstration took place, thus the state space has to be extended by failure states: $S_E = S_D \cup S_F$. Robot glitch characteristics are stored in T_B for some known combinations (s, a) or can be dynamically calculated for some a, e.g. all navigation actions. Failure states S_F can thus be determined: $\forall a \in A_D, \forall s \in S_D : T_B(s, a, s') > 0, s' \notin S_D \implies s' \in S_F$.

To leverage the capabilities of the POMDP reasoning process, the set of actions is extended by specific information gain actions A_G, e.g. requesting the last utterance from a human again or looking for an object again, which cannot be acquired from demonstrations: $A_E = A_D \cup A_G$. For each modality position, utterance, object and body configuration, an generic information gain action $a_{g_{mod}} \in A_G$ is added, with a small penalty in R_B for its effort and a stationary transition model in T_B (see below). Finally, the *Do nothing* action a_{idle} is added.

All elementary actions are mapped to the corresponding robot skill functions e.g. *pick cup* or *goto serving table* which can be executed in the HTN when requested.

T_D contains only prior distributions about environment behavior, but not specific priors modeling characteristics of the robotic system. Therefore for each action, glitch characteristics from T_B have to be included:

$\forall s, s' : T_E(s, a, s') = \frac{T_D(s,a,s') * T_B(s,a,s')}{\sum_{s'_i} T_D(s,a,s'_i) * T_B(s,a,s'_i)}$

E.g. a navigation glitch of the robot could be modeled as

- $T_B(s_{at-c}, a_{goto-t-left}, s'_{at-t-left}) = 0.9$,
- $T_B(s_{at-c}, a_{goto-t-left}, s'_{at-t-right}) = 0.1$

and learned environment variations as:

- $T_D(s_{at-c}, a_{goto-t-left}, s'_{at-t-left\ \&\ cup}) = 0.7$,
- $T_D(s_{at-c}, a_{goto-t-left}, s'_{at-t-left\ \&\ pringlescan}) = 0.3$

Leading to (s' at-t-right not shown) final transition priors:

- $T_E(s_{at-c}, a_{goto-t-left}, s'_{at-t-left\ \&\ cup}) = 0.63$,

- $T_E(s_{at-c}, a_{goto-t-left}, s'_{at-t-left\ \&\ pringlescan}) = 0.27$

In the reward model, for each action $a \in A_E$, small penalties are included: $\forall s_i : R_E(s_i, a) = R_D(s_i, a) + R_B(s_i, a)$ which represent the effort of the action - usually execution duration. Additionally, larger penalties are added for each undesirable failure state $s \in S_F$: $\forall a_i : R_E(s, a_i) = R_D(s, a_i) + R_B(s, a_i)$. By these means, learnt positive rewards (representing goal directed aspects) are complemented by negative rewards (representing risk aversion) from robot knowledge about its own capabilities.

Observations m_E are generated for each region, utterance, object and body configuration encountered in S_E, thus are indirectly generated by the learning process.

The observation model O_E is generated from background knowledge, modeling e.g. typical localization errors of the robot and objects as well as speech recognition errors by acoustic similarities and human activity recognition errors by body configuration similarities. E.g. for the latter, a body configuration similarity metric can be applied to derive a noise model, based on the margin $\langle c_i, c_j \rangle_{svm}$ between SVM classifications of activities c with scale factor α: $\langle m_i, m_j \rangle_{ActReco} = \alpha \langle c_i, c_j \rangle_{svm}$. According to this metric, e.g. the probability that a *Pick* movement with the right arm is perceived as a *Handover* with that arm is much higher than the probability of it being a *Pick* with the left arm.

Summed up, background knowledge used in the mapping process contains knowledge of the robotic system about itself, while the structural characteristics of a mission are learned from human demonstrations. Therefore, the background knowledge can be the same for different missions.

Finally, a policy, is computed from the model using the SARSOP algorithm and queried during runtime with a belief.

V. Experiments and results

Three service robot scenarios were evaluated, sharing the same background knowledge. The execution of each generated POMDP was compared to the performance of a hand-built finite state machine (FSM). First, for each scenario, several demonstrations of the same mission, but with differing courses of events, were performed in front of the robot. Then, using the recordings, a mission model was generated from which a policy was computed. Finally, the real robot executed the learned mission policy autonomously several times, followed by the execution of the FSM.

A. Missions

All missions shared a common space with a simple storage area, a simple serving area, two objects, pick, place, handover, throw-away and several dialog options. Mission 1 (Mi1) encompassed bringing a desired object to a dining table, when verbally requested after some initial dialog. Mission 2 (Mi2), without any verbal dialog, included throwing an object, handed by the human, away or placing it on the storage table, depending on object type. Mission 3 (Mi3) comprised handing a human a desired object directly and included both verbal and non-verbal (body tracking) dialog during execution time.

B. Observation setup

The *robot actor* performed these waiter duties, while the *human actor* represented an interacting human. The robot was placed such that it had a good view on all parts of the scene when actively following the movements of the actor with its head (see fig. 4 for a snapshot of the experiment).

Ten demonstrations of each mission were performed with variations in human actor behavior and initial object placing. These variations were recorded for reproduction during the execution experiment and reflected the robot-independent stochastic properties of the setting.

C. Resulting models

S, A, M sets als well as T, R, O were automatically generated for each mission from the recorded traces:

1) Mi1: $|S| = 500$, $|A| = 12$, $|M| = 18$
2) Mi2: $|S| = 50$, $|A| = 6$, $|M| = 12$
3) Mi3: $|S| = 350$, $|A| = 14$, $|M| = 19$

While the state number seems rather large for the scope of the missions, each combination of modality sub-states results in a state, yet because of interdependencies in the resulting transition model, a fully factored representation is not possible. For Mi1, policy computation took 2 minutes to reach 1% utility precision with SARSOP.

D. Execution setup and results

For execution, the robot was placed at the same starting point as the *robot actor* during the demonstration and the autonomous decision making system was fed with the computed policy. The robot then acted fully autonomously without any external intervention, except its natural modalities of autonomous navigation, human-robot-interaction and object manipulation with a human as interaction partner (see fig. 4 for a snapshot during the experiment).

For each of the three missions, autonomous execution was performed with both the generated POMDP as well as the FSM (hand-tailored by an expert, using fixed thresholds for perception) successively 10 times with each method. The following table shows minimum, average and maximum execution times for both generated POMDP (P) and hand-built FSM (F) in minutes as well as number of mission failures.

	Mi.P	Av.P	Mx.P	Fl.P	Mi.F	Av.F	Mx.F	Fl.F
Mi1	4:25	4:50	5:50	1/10	4:40	5:10	5:35	2/10
Mi2	4:05	4:25	5:00	1/10	4:10	4:35	4:50	0/10
Mi3	5:35	6:00	7:25	2/10	5:50	6:20	7:00	3/10

As can be seen, the generated POMDP performed better than the FSM, especially in missions with spoken dialog (Mi1 and Mi3), in average time and failures, where it can handle noisy distance speech recognition best. In case of Mi2, it still is able to beat the hand-tailored FSM in average time as it is more aggressive in making a decision when facing imperfect body configuration or object detection data.

Fig. 4. Demonstration of Mission 1 by robot actor and human actor with actively watching robot (left) and autonomous execution with the robot and interacting human (right). Gloves improve in-hand object localization and headsets improve speech recognition during the demonstration phase. See accompanying video for an example of demonstration and execution of Mi1 and Mi2.

E. Discussion

It should be noted, that apart from the obvious setting variations between differing demonstrations of a specific mission, the transition probabilities implicitly encode information about where objects can generally be encountered, that they can move with the robot when picked, where they can be placed and also how a dialog can develop. The reward model encodes, e.g. that the object *Cup* should be placed at a certain place, when *bring me tea* was requested during the dialog. This information is exclusively learnt by the presented process - there is no connection between the locatable object *Cup* and the utterance *bring me tea* in the background knowledge as is no information about where it can be found and where it should be brought.

VI. Conclusion and Outlook

By encoding knowledge learnt from observations directly into a POMDP model, the robot is able to decide autonomously during mission execution while considering uncertainty and can deviate from demonstrated courses of events, e.g. unforeseen obstacles. To sum it up: a flexible representation, a POMDP model, which contains the qualitative characteristics of a demonstrated mission but ensures robust behavior during execution, is learnt from demonstrations.

However, in the current stage, several limitations remain, which have to be solved next to create a more powerful system. Most important, the challenge of model complexity, foremost concerning the size of the state space, has to be tackled by investigating the learning of hierarchical POMDP or MOMDP representations. The investigation of longtime learning by using a structured representation of learnt transition and reward knowledge is ongoing. Finally, the robot could follow the demonstrating human not only with its neck, but also the platform, making home-tour scenarios possible.

VII. Acknowledgements

This work has been partially conducted within the german SFB 588 “Humanoid Robots” granted by DFG and within the ECs Integrated Project DEXMART under grant agreement no. 126239 (FP7/2007-2013).

References

[1] E. J. Sondik, “The optimal control of partially observable markov decision processes,” Ph.D. dissertation, Stanford university, 1971.

[2] A. R. Cassandra, L. P. Kaelbling, and M. L. Littman, “Acting optimally in partially observable stochastic domains,” in *In Proceedings of the Twelfth National Conference on Artificial Intelligence*, 1994.

[3] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, “Planning and acting in partially observable stochastic domains,” *Artif. Intell.*, vol. 101, no. 1-2, pp. 99–134, 1998.

[4] J. Pineau, G. Gordon, and S. Thrun, “Point-based value iteration: An anytime algorithm for POMDPs,” in *International Joint Conference on Artificial Intelligence (IJCAI)*, August 2003, pp. 1025 – 1032.

[5] T. Smith and R. Simmons, “Focused real-time dynamic programming for mdps: Squeezing more out of a heuristic,” in *Nat. Conf. on Artificial Intelligence (AAAI)*, 2006.

[6] H. Kurniawati, D. Hsu, and W. Lee, “SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces,” in *Proc. Robotics: Science and Systems*, 2008.

[7] A. Foka and P. Trahanias, “Real-time hierarchical pomdps for autonomous robot navigation,” *Rob.Aut.Sys.*, vol. 55, no. 7, 2007.

[8] K. Hsiao, L. P. Kaelbling, and T. Lozano-Pérez, “Grasping pomdps,” in *ICRA*, 2007, pp. 4685–4692.

[9] J. Hoey, A. von Bertoldi, P. Poupart, and A. Mihailidis, “Assisting persons with dementia during handwashing using a partially observable markov decision process,” in *ICVS*, Bielefeld, Germany, 2007.

[10] R. Jaulmes, J. Pineau, and D. Precup, “A formal framework for robot learning and control under model uncertainty,” in *Robotics and Automation, 2007 IEEE International Conference on*, April 2007.

[11] S. Ross, B. Chaib-draa, and J. Pineau, “Bayes-adaptive pomdps.” in *NIPS*, J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Eds. MIT Press, 2007.

[12] A. P. Shon, J. J. Storz, and R. P. N. Rao, “Towards a real-time bayesian imitation system for a humanoid robot,” in *Robotics and Automation, 2007 IEEE International Conference on*, 2007, pp. 2847–2852.

[13] M. Pardowitz, S. Knoop, R. Dillmann, and R. Zollner, “Incremental learning of tasks from user demonstrations, past experiences, and vocal comments,” *IEEE Trans. on Systems, Man, and Cybernetics*, April 2007.

[14] M. Tenorth and M. Beetz, “KnowRob — Knowledge Processing for Autonomous Personal Robots,” in *IEEE/RSJ International Conference on Intelligent RObots and Systems.*, 2009.

[15] W. Walker, P. Lamere, P. Kwok, B. Raj, R. Singh, E. Gouvea, P. Wolf, and J. Woelfel, “Sphinx-4: A flexible open source framework for speech recognition,” SUN Microsystems, Tech. Rep., 2004.

[16] M. Lösch, S. Schmidt-Rohr, S. Knoop, S. Vacek, and R. Dillmann, “Feature set selection and optimal classifier for human activity recognition,” in *RO-MAN*, 2007.

[17] P. Azad, T. Asfour, and R. Dillmann, “Combining apperance-based and model-based methods for real-time object recognition and 6d localization,” in *IROS*, Beijing, 2006.

[18] S. R. Schmidt-Rohr, S. Knoop, M. Lösch, and R. Dillmann, “Bridging the gap of abstraction for probabilistic decision making on a multimodal service robot,” in *Robotics: Science and Systems*, Zürich, 2008.