OfficeMate: A Study of an Online Learning Dialog System for Mobile Assistive Robots

Steffen Müller, Sina Sprenger, Horst-Michael Gross
Ilmenau University of Technology
P.O. Box 10 05 65, 98684 Ilmenau, Germany
Email: Steffen.Mueller@tu-ilmenau.de,
Sina.Sprenger@gmx.de, Horst-Michael.Gross@tu-ilmenau.de

***Abstract*—Service robots in the near future are supposed to live together with humans in their private homes for a longer time period. In this situation, experience and attitudes of the users change and thus, the robot has to develop its behavior, too, and it has to adapt to the user's way of interaction and the user's needs. The contribution of this paper is a probabilistic decision planner implementing the idea of online learning dialog strategies for a mobile service robot in long-term interaction. The planning system is part of a modular multi-modal dialog system and allows for an autonomous personalization of the robot's actual interaction behaviors. A model of observed transitions and user's rewards using mixtures of discrete samples is proposed for efficient inference in a factor graph model. The practicability of the dialog system and the rewarding mechanism have been evaluated in a ten day realworld experiment with 16 users.**

Keywords–online learning; dialog system; probabilistic planner

I. INTRODUCTION

The work presented has been conducted in the scope of the research group SERvice RObotics for health (Gesundheits) Assistance (SERROGA) [1], which intends to develop demonstrators for robotic applications in the context of prevention and assistance for elderly people living alone in their home environment. As a vision, we see a robot, that is living together with the human in a long-term interaction situation. Furthermore, we expect the development of an emotional binding of the user to his or her personal robot over the time, which is reinforced by the ability of the system to adapt to the user's needs and preferences. The intended platform to be used in the SERROGA project is a Scitos-G3 (see Fig. 1), that has been developed in the EU funded project CompanionAble [2][3]. An intuitional communication is realized by a multi-modal user interface consisting of a touchscreen, touch sensitive cover, and a touch sensitive patch of fur used for petting the robot. Actual work in progress intends for inclusion of speech recognition as an additional input channel. For output, the robot can use synthesized voice, the screen, as well as an artificial face consisting of two eye displays.

A laser range finder and a Kinect sensor, together with a differential drive enable autonomous localization and navigation skills. A fish-eye camera is used for person detection and tracking, which is a key functionality for successful interaction. Additionally, for giving explicit feedback by a user, special positive and negative reward buttons are placed on top of the screen.

The aim of this work is to enable and understand a long-time development and adaptation of a multi-modal human-robot dialog. In that context, we identified three phases in the interaction. At the beginning, in the first phase, the user needs to get to know the robot. The system has to give advice how to use it and has to introduce its capabilities. The dialog initiative is primary at the side of the robot. Later, in a second phase, when the user knows better about the capabilities of the robot, the initiative will be in the user's hand, and the robot should learn the user's preferences and needs. That means, the robot is supposed to learn which services are used in which situation and what are the user's attitudes towards the various options the robot has in its dialog behavior. Also, preferred selections are learned by the robot in order to apply that knowledge in the third phase. When that third, stable phase is reached, the robot can make use of the observations in interaction with that specific user. So it is able to act proactively depending on the current situation. Then, a mixed initiative dialog should emerge even with the limited domain of the robot's services. Nevertheless, the ability to learn and change interaction behavior should not be limited to the initial phases. Changes of user's attitudes have to be tracked continuously and life-long.

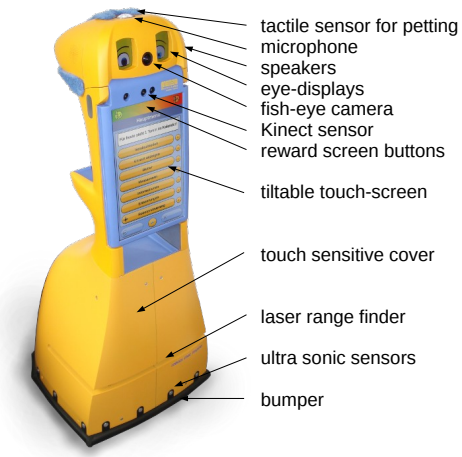

Figure 1. Scitos-G3 robot "Max" used for the evaluation.

The remaining part of this paper is structured as follows: First, a brief discussion of approaches from literature for learning dialog behavior is given, followed by the presentation of our own contribution. Thereto, in Section III, first, the dialog system is described at a glance, and afterwords, the adaptive parts, the probabilistic planner and the models used, are explained. In Section IV, our experience with the described

 ISBN: 978-1-61208-341-4

system in user test is presented, and results are discussed.

II. Related Work and Discussion

In literature, several approaches for dialog modeling can be found mainly in the field of speech-based dialog systems, which also have capabilities of adaptation. A common approach is using Reinforcement Learning techniques for optimization of a dialog flow, even when inputs are uncertain. The Partially Observable Markov Decision Process (POMDP) is used here, but this comes along with high computational effort, which is intended to be overcome by several approaches for simplification [4][5]. These approaches try to find a policy in order to maximize the discounted future reward, that mainly is generated by a system internal reward function. Therefore, the system designer has to define in advance what the long-term goals during upcoming dialogs will be.

Pineau et al. [6] applied a POMDP model for learning an optimal dialog behavior for a service robot called Pearl. Although they applied a hierarchical decomposition of the assistance task, the complexity of the realizable functionality is very limited. That approach also relies on a hand crafted reward function and the policy is computed offline before application. Thus, the robot can not consider individual user's characteristics discovered at runtime.

One disadvantage of many similar Reinforcement Learning-based approaches in the domain of dialog learning is the batch update, where in a training phase interaction data is acquired, and afterwards the optimization run is applied offline in order to generate the productive dialog system.

A model for learning a behaviour online from direct user rewards is called "Training an Agent Manually via Evaluative Reinforcement" (TAMER) [7][8]. This approach explicitely models, which feedback a user gives for a certain state-action pair, and then acts greedily in order to get the maximum reward for the next action. The argumentation for this λ_0 strategy is the idea, that the human supervisor estimates the utility value of a state-action pair and already represents this in the reward signal. This might be correct to a certain degree, but has a clear disadvantage. The system can only act in order to achieve user goals, but is not able to incorporate internal goals or wishful target states. One interesting aspect of the TAMER model is the reward model. This allows to predict the user rewards and apply them internally - even if the user is not giving feedback for each action. Thus, this model allows that the user has to get active by giving feedback only if s/he wants to modify the behavior, not if s/he is pleased with it.

An alternative dialog system applying probabilistic inference is presented in [9] and later in [10]. Inference techniques have been applied to a statistical model of the dialog in order to reason the goals the user might have in mind and, therefore, decide which information needs to be asked or given in the next steps of dialog. Unfortunately, this idea is not directly transferable to our scenario, where the goal of the dialog is not only determined by the user but also by the system itself (e.g., the robot should engage the user in communication or physical activities). Additionally, the required direct reward to be given by the user, which is used to modify the way things are communicated, is hard to introduce in that approach.

In our approach, we need to combine system internal goals, which mainly are i) fast task completion and ii) less correction steps by the user with explicit and implicit rewards given directly by the user during the interaction. As explicit reward, we consider positive or negative feedback by means of pushing the respective like or dislike buttons on the screen, while implicit rewards are unconscious signals like the rate of petting the robot, or simply ignoring the robot's attempts to interact with the user.

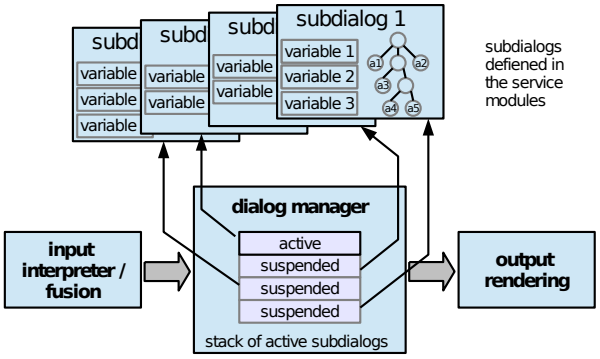

Figure 2. Architecture of our dialog system - The dialog manager holds a stack of subdialogs each defined by as set of state variables a decission tree of possible actions.

To get capability of online learning/adaptation, we would like to get rid of the complicated optimization problem of finding a complete policy each time we get new observations, although we are only interested in the optimal action for the current situation. This is possible by means of an online planning mechanism and also allows for changing optimization goals as well as discovering new states of the dialog at runtime, which would be difficult or even impossible for implicit planning methods.

Dialog modeling techniques existing today are mostly related to a very complex application development process. Our aim is to provide a framework for rapid application development, which is realized by combining simple frame-based multi-modal dialog with the capabilities of optional adaptation without introducing additional configuration effort. A key to a manageable design effort is the possibility for problem decomposition. According to hierarchical abstract machines [11], that are also used in the Reinforcement Learning domain for restriction of the action space, in our approach, individual subdialogs are defined as independent modules, each restricting the policy to a reasonable subset and having the ability of calling other subdialogs on demand. How this modularization is realized in our system is explained in the next section.

III. Adaptive Modular Dialog Systems

The implementation of the dialog system is based on the robot middle-ware MIRA [12][13] and implements the control layer in our software architecture [2]. Therefore, it integrates well with the other robot software components for navigation, perception skills, and graphical user front-end and realizes an interface to the robot infrastructure for the individual services. The software architecture supports a modular design, where each subdialog (greeting, weather info, news, entertainment, etc.) is an independent module defining a service. Thus, it is easy to add new functionality and refer to, or combine existing dialog capabilities in new dialogs, such that the borders of the modules get blurry for the user, who is perceiving the robot as one personality. The software modules implement a

 ISBN: 978-1-61208-341-4

content specific back-end functionality as well as define the subdialogs needed. The configuration for a subdialog is mainly a definition of the state space S (variables holding user inputs and context information) and a set of output actions available $A = \{a_1, \ldots, a_K\}$ (see Fig. 2 upper part). For instance, actions are expressing a greeting multimodally or asking which website the user likes to see. In order to reduce the necessity for exploration in an unordered set of actions and in order to prevent from selection of irrational sequences frustrating the user, it is necessary to limit the set of selectable actions in each state S. In our system this is done by a manually designed decision tree over the state variables S. Each node in the tree decides between two alternative branches, each consisting of sets of possible actions or further subtrees. These sets allow to define options from which the dialog manager has to choose later on by means of the probabilistic planner. By defining trees with only one action in the branches, it is possible to realize deterministic dialog strategies as well. In that case, the system behaves like a Finite State Machine. For realizing action sequences, it is necessary to consider the history of executions of the actions in the dialog state. Therefore, each possible action a_k has a counter H_k, holding the number of executions of a_k since the latest activation of the subdialog.

The dialog state of an independent subdialog, besides the counters for the actions, comprises a set of variables $\{V_1, \ldots, V_N\}$ representing user inputs, but also system internal data and events that are of relevance for the decision on the next action of the robot. For example, the number of appointments to be reminded or the answer (yes/no) to the question if appointments should be listed are such state variables in a "reminder" subdialog. The variables have a specific range that can either be discrete or real-valued, which has to be considered later in the respective similarity functions. The range is defined by the type of a variable that also defines which inputs can be filled in it by the input interpreter. Additionally, all variables are labeled with a certainty value $\{C_1, \ldots, C_N\}$ that expresses to what degree the respective information is known, unconfident, or unknown. This, for example, allows to model the ambiguity of speech recognition inputs or other probabilistic observations.

Therefore, the state representation for one subdialog is a discrete vector:

$$S = (V_1, \ldots, V_N, C_1, \ldots, C_N, H_1, \ldots, H_K) \qquad (1)$$

A. Control Flow in the Dialog System

Having defined the structure of a subdialog, now the coordination of user inputs, multiple active subdialogs, and the system output generation are explained.

In general, a turn-based control flow is realized where user input turns and system turns alternate. Once a system output action is executed, the system waits until expected user inputs are recognized or until a timeout triggers a new system turn. All multi-modal user inputs are processed in parallel by the input interpreter and will update the dialog state of the respective subdialog. Inputs or internal events are filled in variables of all subdialogs that match the respective type. Special variables may activate a subdialog if they get filled by an input or event.

The dialog manager holds a stack of active subdialogs (see Fig. 2), where the top most is that one evaluated each time a system action is necessary (start of a system turn). The dialog manager evaluates the decision tree of the top most subdialog in the stack using its current state $S = s_0$ in order to get the possible action set for the current situation. If only one action is available, it is executed by the output renderer directly. The interesting part, where adaptation takes place, is given when multiple actions are allowed by the decision tree, and the probabilistic planning process, as described in section III-C, is triggered. This planning yields a probability distribution on the actions $P_{plan}(A)$, that maximizes the probability of reaching a system internal goal state while maximizing the user's rewards on the way.

Since the system does not know neither the user's rewards and the possible transitions in the dialog states nor the goal states that are defined by the actions in these states, there is a need for exploration additionally to the aim for exploitation of the knowledge allready acquired (exploration-exploitation dilemma). Furthermore, the progress in the phase model of the long-time interaction, introduced in the intro part of this paper, also has to be considered during action selection. Thus, two additional probability distributions are used for action selection. The first represents the number of executions of each available action $P_{count}(A)$ to enforce that all possible actions are tried out equally during exploration. The second distribution $P_{prio}(A)$ allows for consideration of a priority that is depending on the progress in the long-time interaction. In this way, in the beginning phase, more explanatory actions are selected, and in the stable phase only straight actions without additional help messages and proactive actions like offering services in a certain situation are recommended. Consequently, the three influence factors $P_{plan}(A)$, $P_{count}(A)$, and $P_{prio}(A)$ are combined, and the action to be eventually executed is selected by drawing from that resulting distribution.

When the action is executed, mainly screen and speech outputs are generated, that may refer to values of the variables in the subdialogs, and communicate content suitable for the current situation (asking questions, confirming inputs or giving answers). Also special actions, like activation of other subdialogs or canceling an active subdialog, are possible. If a new subdialog is activated, the former top most in the stack gets suspended and can execute one more action in order to react to that special situation. If the interrupting subdialog is finished, the suspended one returns to the top of the stack and gets resumed. This resuming is a special action that may be used to bring the user back into the context of the former conversation.

B. Modelling of Interactions

For the planning and adaptation, the system needs to represent knowledge on the history of interactions with the user. This is done by means of several probabilistic models which are described in the following. The dialog manager first builds up a persistent probabilistic transition model at runtime that is representing the probability of reaching a certain state S' given a predecessor state S and the executed action A. Here, user specific decisions and reactions are learned as well as the internal restrictions on the state sequences, such that the planning system does not need to be configured with that knowledge in before. The representation of that model is a weighted sum of samples $s_j = (S', S, A)_j$ each of them equipped with a weight w_j. These samples have a very high dimensionality (keep in mind the elements of the state vector S) and, therefore, it is very unlikely that exactly the same states

 ISBN: 978-1-61208-341-4

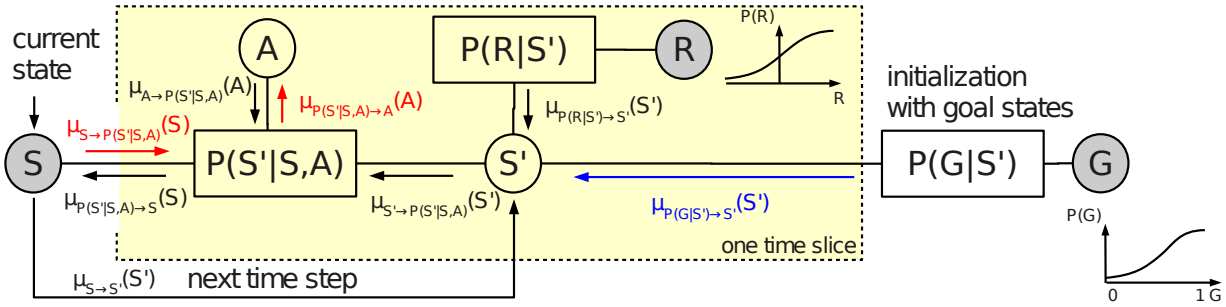

Figure 3. Dynamic factor graph for planning of next action A, given the current state S, a goal state distribution G and a reward proposal distribution R.

appear very often. To generalize consequences among similar states, a similarity function $\delta_{S',S,A}(s_a, s_b) \mapsto [0,1]$ defines a neighborhood among samples s_a and s_b. The subscript S', S, A determines the dimensions considered in that similarity. Later, when operating with marginal distribution,we need to compare samples on certain dimensions only. The probability of a particular transition $t = (S'_t, S_t, A_t)$, by means of that set of all samples s_j and similarity function is defined as follows with a normalization factor η:

$$P(S'_t, S_t, A_t) = \eta \sum_j w_j \delta_{S',S,A}(t, s_j) \quad (2)$$

For realizing a goal directed planning of action sequences, additionally a model of goal states $P(G, S)$ and a model of rewards $P(R, S)$ gained in a certain state, are defined similarly also as weighted set of discrete samples. Here, G can be 1 for a success state, e.g. the dialog was successfully completed, or it can be 0 for a fail state that has to be avoided in future interactions. This is for example the case, if a dialog turn times out without any user reaction.

The rewards are only recorded if positive or negative reward events took place. By ignoring zero rewards, the policy remains stable, even if the user is pleased with it and does not reward every action individually.

The goal and reward models can be used to evaluate a probability for a state S to be a goal state, as well as the probability to get a high reward in that state. The models return 0.5 if there are no similar observations in the sample set.

Initially, these user-specific models for each subdialog are empty and have to be filled during the interactions by observing and counting the real transitions, rewards, and occurrence of goal labels.

C. Probabilistic Action Planning

This section describes the planning mechanism used to deduce a probability distribution $P_{plan}(A)$ for the available actions, that maximizes the probability of reaching a goal labeled state while gaining as much reward as possible on the way to a goal state. For that kind of problems, the probabilistic model of our dialog is represented as a dynamic factor graph (see Fig. 3) for which a message passing algorithm called max product algorithm [14][15][16] exists. The max product algorithm can find the marginal distributions for all unobserved variables in the factor graph that maximize the probability given the set of fixed or observed variables.

The idea of that algorithm is to perform local operations in the nodes of the factor graph and propagate the results in form of messages $\mu_{sender \to receiver}(domain)$ along the tree structure of the graph in order to get the result in the node of interest, which is the A node of the first time step in our case. Unfortunately, in our dynamic factor graph, the number of time steps to reach a goal state is not known in before, but, since we are only interested in the next action, the inference can be executed in a loop, with one iteration for each time step to be looked in the future. This is shown by the message $\mu_{S \to S'}(S')$ in the figure. This is only possible in an acyclic factor graph, which is the reason for the complex state S and the respective factor models we have chosen in the model. A further factorization would possibly simplify the factor potentials, but the complete factor graph would not longer be acyclic afterwards. This would make the time step loop trick impossible since belief propagation needs to iterate on the complete structure of a loopy factor graph.

A central decision for the factor graph algorithm is the form of representing the probability distributions in the nodes of the factor graphs and in the messages sent between them. Normally, Gaussians or discrete distributions are used for that, but in our case the dimensionality of the distributions' domains is too large for that. Hence, we propose a representation as mixture of discrete samples as already introduced for building up our transition model $P(S', S, A)$. In the following, we briefly show, how the operations required by the max product algorithm have been implemented for that kind of distribution representation.

The **initialization** of the planning loop requires setting the distribution of the state S' to the desired distribution of goal states we expect. For that, the model of goal probabilities $P(G|S')$ for each observed state provides the fraction of occurred goal labels in the actions leading to that state. Asuming, that the target probability for reaching a goal is distributed like the little diagram shown in Fig. 3 bottom right, the $P(G|S')$ table is used to weight the states S' of the transition model, thus a set of weighted S' samples results. This is our initial state distribution $P(S')$ of the last time step in the planning horizon (blue message $\mu_{P(G|S') \to S'}(S')$).

Then the planning loop starts with the **step 1**: According to the max product algorithm, the distribution $P(S')$ has to be multiplied by the message $\mu_{P(R|S') \to S'}(S')$ from the reward model. This reward model $P(R|S')$ is also a set of state samples s_i, that maps to the average reward r_i gained in that state. We use the proposition, which is maximizing the reward along the sequence of states, and define the distribution of the expected reward as a sigmoid function (shown in the little diagram besides the R node in Fig. 3). Knowing this, for each sample s_j in the S' distribution, we can compute an

updated weight $w_j^{(S')}$ by comparing the states of these samples to states of the samples s_i in the reward model and applying the sigmoidal proposition distribution indicating the desirability of the respective reward. Adding an ϵ ensures that weights will be vaild, even if no samples of the reward model do match in the similarity function.

$$w_j^{(S')} = w_j \frac{0.5\epsilon + \sum_i \frac{1}{1+e^{-r_i}} \delta_{S'}(s_j, s_i)}{\epsilon + \sum_i \delta_{S'}(s_j, s_i)} \quad (3)$$

To reduce the complexity for the next computations, samples with low weights are omitted, and samples with high similarity $\delta_{S'}(s_j, s_i)$ are merged until a maximum number of samples remains. We used 20 samples at maximum in all message distributions.

In **step 2**, this sample set $P(S')$ is sent to the transition model as message $\mu_{S' \to P(S'|S,A)}(S')$ to get multiplied to the conditional probability distribution $P(S'|S, A)$.

Since we only have counters $w_j^{(S',S,A)}$ for the occurrences of the various state transitions, we need to divide $P(S', S, A)$ by $P(S, A)$ to get the conditional, which is done by calculating new weights $w_j^{(S'|S,A)}$ for the samples s_j according:

$$w_j^{(S'|S,A)} = \frac{w_j^{(S',S,A)}}{\sum_i w_i^{(S',S,A)} \delta_{S,A}(s_j, s_i)} \quad (4)$$

This can be done before planning starts and has only to be updated when a new transition has been observed.

In the $P(S'|S, A)$ node, the product of the factor potential and all incoming messages has to be calculated. Therefore, each sample of the transition model gets a new weight $\hat{w}_j^{(S'|S,A)}$ (5), which incorporates the similarities of samples s_i in the incoming message $\mu_{S' \to P(S'|S,A)}(S')$. The message $\mu_{A \to P(S'|S,A)}(A)$ is assumed to be uniform and is not considered in the weight computation. It would be possible to incorporate priors on actions here to realize a dependency of the actions on the progress in the long-term interaction phase.

$$\hat{w}_j^{(S'|S,A)} = w_j^{(S'|S,A)} \sum_{i \in \mu_{S' \to P(S'|S,A)}(S')} \delta_{S'}(s_j, s_i) w_i^{(S')} \quad (5)$$

The max product algorithm now needs to find the maximum probability for each value of the variable for the outgoing message. In our case, that is the $\mu_{P(S'|S,A) \to S}(S)$, and thus the goal variable is S. For each sample, all other samples are compared using $\delta_S(s_i, s_j)$, and only the one with the maximum weight will be used. As result, we get a message $\mu_{P(S'|S,A) \to S}(S)$, that can be processed further in the variable node S.

Here, in **step 3** we have to test, if the current state s_0 of our subdialog matches the inferred predecessor state distribution $P(S)$ for the planned sequence to the goal state of a length of the current iteration. In case of sufficient probability, the action A can be deduced, that maximizes probability of going one step towards the goal state starting from s_0. This is done by sending the message $\mu_{S \to P(S'|S,A)}(S)$, which is simply our current state s_0 with weight $w_0 = 1$ to the factor node (red message in Fig. 3).

There, again the product and maximization has to take place, which is realized by re-weighting the samples of the transition model again. Here, the intermediate weights already containing the message from S' can be reused.

$$\tilde{w}_j^{(S'|S,A)} = \hat{w}_j^{(S'|S,A)} \delta_S(s_j, s_0) w_0 \quad (6)$$

With these new weights, the maximization along the S and S' dimensions can be done in oder to get a probability for each action to reach a goal within the current time horizon. This is done by grouping the samples of the transition model s_j by the discrete action dimension A and checking for the maximum weight $\tilde{w}_j$ in each group.

Before going to the next planning time step by sending the $\mu_{S \to P(S'|S,A)}(S)$ message renamed as $\mu_{S \to S}(S)$ and starting over with step 1, for each action in the set of available actions, the maximum probability over all time steps is stored. Afterwards, this is used to gain the $P_{plan}(A)$ distribution used for action selection as described before.

D. Prediction of User Preferences

In many situations in a repeatedly conducted dialog between the robot and the user, the annoying questions for options (e.g., which website should be shown) can be omitted, when using the former choices in a similar situative context. The transition model we built from the dialog history, exactly contains these information.

Thus, instead of asking the user for the information, the robot can try to infere the desired value, which is changing the state S of the subdialog similarily without any further inputs from the user. Depending on the outcome of that, the dialog can continue either with a confirmation of that fact or with a question for specification of the information, if the inference did not yield a significant probability for either option.

By means of that, also the proactive situation dependent offer of services can be realized easily, where the correctness of the suggestions strongly depends on the context variables considered in the subdialogs state vector.

IV. Experimental Evaluation

The correct function and ability of considering observed reactions of the user during robot's action selection first has been validated by means of a set of simulated dialog sequences not embedded in a complete robot application. This functional test showed the learning capabilities as expected, but it is hardly possible to simulate a more complex application realistically. Even more, it is not possible to predict the impact of our dialog system on real users.

Due to these circumstances, the proposed system has afterwards been evaluated in a separate application before being applied for realizing the user interface of our health assistant robot. Caused by restrictions on access to the robot as well as the number of test users needed in combination with the intended long-term interaction, an evaluation scenario has been chosen, that involved 16 members of our lab in parallel. The robot has acted as an "Office Mate" by visiting each trial participant once a day and offering its services after the user confirmed his/her supposed identity and the respective persistent interaction models have been loaded. The experiment took ten workdays in order to give time for an observable change of the robot's behavior.

The participants had to fill out two questionnaires with a first focus on the long-term acceptance of a robot with an adaptive dialog behavior, and a second focus on the practicability of

 ISBN: 978-1-61208-341-4

our realization. Questions mainly followed the Almere Model [17] , while questions regarding the perception of the robots adaptation skills have been added.

The first questionaire had to be filled in before the experiments to get the baseline and find preferences regarding the form of being addressed by the robot (formal or informal), as well as to get hints on the set of websites to be provided by the robot during the interactions. A second questionnaire after the experiment asked for the personal experience of the participants.

The tests have additionally been accompanied by an observer to reveal usability problems, and aforementioned questionaires have been used for a qualitative evaluation due to the low number of participants.

A. *Office Mate Services*

The application for our Office Mate scenario covers a couple of services. For giving an impression on the capabilities and the options for adaptivity, these services are described in the following.

The adaptivity was mainly realized in the **main menu** subdialog, which had a set of optional actions that comprise (M1) offering a normal selection menu, which of the services to execute next, (M2) proposing an unused service to introduce it to the user with the question whether to start it or not, (M3) executing a service proactively depending on the prediction of a selection. The user in that case only has to confirm or deny, whether that the service is going to be started. The context state variables for the prediction of the next service to be selected by the user include one counter for each service available counting the number of activations of that service in that session. By means of these context variables, the system can learn arbitrary sequences of services used.

However, actions (M4) to (M6) are the same as (M1) to (M3), except that a written advice on how to interact with the robot in that situation is given on the touch screen additionally. These later actions are designed for the first phase in the long-time interaction, where the user should learn how to use the robot.

Further variability is implemented in the **greeting** subdialog, which is always started at the beginning of an interaction session with a participant. Besides a deterministic greeting, here the optional actions were: (G1) giving a tutorial on how to use the robot, (G2) asking for wellbeing, and (G3) quitting the greeting subdialog and continuing with the main menu subdialog. During the greeting subdialog, a sequence of these actions is also possible.

The first two services offered are **news** and **entertainment** via websites in a browser. These subdialogs had optional actions as well, which were (N1) presenting a longer list of respective websites for selection, or (N2) suggesting a website based on the predicted selection known from previous interactions. Since the number of different websites visited is part of the state variables of that subdialog, it is possible that the system can learn and predict a sequence of sites preferred by the user. Also in these services, the available actions could be executed with an additional advice on how to use the screen menu.

The third service, a **weather forecast**, had the only option to present weather warnings, if available, automatically or to wait until being asked for. In all cases, the current temperature and weather conditions as well as a two day forecast are presented on the screen.

Two more rather simple services are answering questions for current **date and time** as well as showing the **menu of the refectory**. Since these services do not require further decisions by the user, there are no optional actions.

A last service offered was a **reminder service**. The user was able to edit and show appointments in a list for the current week or the current day. On the main menu, there was an indication on the number of reminders for that day. Since this number is also used in the context variables of the main menu, the proactive presentation of reminders was possible if the user teached that behavior by selecting the reminder presentation manually some times. Unfortunately, the calendar was not synchronized with the Google Calendar usually used in our lab. Therefore, that service had not been used consequently by the participants.

To quit a session with the robot and send it to the next user, the option for a good bye dialog was available in the main menu.

B. *Results and Discussion*

When asked for their expectation to the Office Mate robot, among a couple of additional features for such a robot (sending it to others, sending it to get coffee, using it as avatar by a video conference), few generic aspects regarding adaptation have been mentioned by the participants beforehand. One aspect was that the robot is supposed to know when the user can be disturbed and when not. Unfortunately, that is only possible if detailed context information on the user and the situation at all is available to the robot.

Not all participants of the experiment have been available all the ten days; thus a difference in the experience for different durations of interaction could be observed. All participants who had more than four interactions did notice that the robot learned their preferences and also changed its behavior over time. That mostly is related to the prediction of user's choices (see III-D), leading to a suggestion of following services if the user's attitude is stable. Here, the robot developed an individual sequence of suggestion of services from the main menu as well as individual sets of websites preferred for each user.

Unfortunately, most of the users mentioned, that they were a bit confused by the option of rewarding the robot. They wanted to reward explicitly special aspects of the complex behavior like diction or level of advice. It was not transparent to them, what aspects are variable in this situation and to which aspect the reward refers. This is an indication for using more implicit and system-internally generated rewards in future, that the users are not aware of. By means of that, the user does not need to know the alternatives the robot has in certain situations. Concluding this, offering explicit good/bad buttons is not a practical way for getting rewards from the user.

Those users who used the reward buttons could also influence the dialog flow and the way of presentation of information. If the tutorial in the greeting subdialog had been punished, it occured less often during the following days. Also, the written advice on how to interact in certain situations appeared less often during the experiment if the user punished that behavior by means of negative rewards. However, the user's perception of the exploration of alternative actions is critical. It was confusing for many users, that the robot, although they had already rewarded a behavior, acted again

 ISBN: 978-1-61208-341-4

in that unwanted way. The reasons for this were the complex state model on the one hand and the exploration strategy on the other hand. In many situations, the user might think that s/he already had passed exactly that sequence, but in the robots state variables there is a difference in the history or in the certainty values, causing that the situation is unknown to the robot, thus an exploarative action is selected that the user confused. Additionally, even if the robot knows the situation from past interactions, the need for exploration of longer sequences of actions yields further executions of unwanted actions regardless of the history in the reward model.

For a better generalization over states that are different in only a few dimensions, the planning step in the transition model needs to be improved in order to recombine parts of samples. That will help to reduce the number of observations the system needs to learn a possible path through the state space. Furthermore, it may be possible to apply a more greedy action selection strategy as proposed in the TAMER system [8] to reduce unexpected repetitions of actions that result from exploration steps. Unfortunately, this conflicts with the ability to find an action sequence that reaches system internal goals.

An additional possibility for reducing the number of rewarding events and improving generalization skills is the handling of recurring variations of actions in different situations. In the Office Mate implementation, there was an alternative with or without additional output in many situations. It would be better to introduce a global property "use additional support outputs" which is followed if possible and switch it on and off by only one action instead of generating almost similar copies of arbitrary actions. Therefore, the negative reward for one output in situation A could influence the form of output in situation B without having seen situation B in before. Besides the degree of advice needed, also the volume of voice outputs or the form of addressing the user (formal or informal) are candidates for such global properties.

The overall results showed that the planning algorithm does work very well, but the way of configuring the subdialogs has to be improved in future real applications considering the findings discussed above. More effort has to be put in the deduction of meaningful reward from the user's reactions on robot's behavior. Also the perception skills of the robot always could be improved to extend the context variables of the subdialogs. By means of that, proactive behavior of the robot can be better suited to the actual situation.

V. Conclusion

We could show in an experimental setup, that a dialog manager using an online life-long learning transition model for online planning of action sequences can be realized. The very high dimensional state space of a human robot dialog can be managed by splitting the whole application into independent subdialogs. This also improves the application development process by modularization. An efficient way for the probabilistic inference during the planning process could be realized by a sample-based representation of probability distributions. At last, independent of the planning algorithm, we could identify a couple of design issues with our test application, that have to be improved in future implementation.

Acknowledgment

This work has received funding from the Federal State of Thuringia and the European Social Fund (OP 2007-2013) under grant agreement N501/2009 to the project SERROGA (project number 2011FGR0107).

References

[1] "SERROGA webpage", 2014, URL: http://www.serroga.de [accessed: 2014-04-01].

[2] H.-M. Gross, C. Schröter, S. Müller, M. Volkhardt, E. Einhorn, A. Bley, C. Martin, T. Langner, and M. Merten, "Progress in Developing a Socially Assistive Mobile Home Robot Companion for the Elderly with Mild Cognitive Impairment", in Proc. IEEE/RJS Int. Conf. on Intelligent Robots and Systems (IROS 2011), pp. 2430-2437.

[3] H.-M. Gross, C. Schröter, S. Müller, M. Volkhardt, E. Einhorn, A. Bley, T. Langner, M. Merten, C. Huijnen, H. van den Heuvel, and A. van Berlo,"Further Progress towards a Home Robot Companion for People with Mild Cognitive Impairment", in Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics (IEEE-SMC 2012),2012, pp. 637-644.

[4] T. Blaise, J. Schatzmann, and S. Young. "Bayesian update of dialogue state for robust dialogue systems." In Proc. of Acoustics, Speech and Signal Processing, ICASSP 2008, 2008, pp. 4937-4940.

[5] M. Toussaint, S. Harmeling, and A. Storkey, "Probabilistic inference for solving (PO)MDPs", Informatics Research Report 934, School of Informatics, University of Edinburgh, 2006.

[6] J. Pineau and S. Thrun. "High-level robot behavior control using POMDPs." AAAI-02 Workshop on Cognitive Robotics, Vol. 107, 2002.

[7] W. B. Knox and P. Stone. "Interactively shaping agents via human reinforcement: The TAMER framework." In Proc. of the fifth international conference on Knowledge capture. ACM, 2009, pp. 9-16.

[8] W. B. Knox, P. Stone, and C. Breazeal, "Training a Robot via Human Feedback: A Case Study." In: Social Robotics. Springer International Publishing, 2013, pp. 460-470.

[9] H. M. Meng, C. Wai, and R. Pieracciniet, "The Use of Belief Networks for Mixed-Initiative Dialog Modeling", In Transactions on Speech and Audio Processing, 2003, vol. 11, no. 6, pp. 757-773.

[10] F. F. Martinez, J. Blzquez, J. Ferreiros, R. Barra, J. Macias-Guarasa, and J. M. Lucas-Cuesta, "Evaluation of a spoken dialogue system for controlling a Hifi audio system", In Proc. of Spoken Language Technology Workshop, IEEE SLT 2008, 2008, pp. 137-140.

[11] R. Parr, and S. Russell, "Reinforcement learning with hierarchies of machines." in Advances in neural information processing systems, 1998, pp. 1043-1049.

[12] "MIRA website" URL: http://www.mira-project.org [accessed: 2014-04-01].

[13] E. Einhorn, R. Stricker, H.-M. Gross, T. Langner, C. Martin,"MIRA - Middleware for Robotic Applications" in Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2012), 2012, pp. 2591-2598

[14] C. M. Bishop, "Pattern Recognition and Machine Learning", New York: springer, 2006.

[15] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, "Factor graphs and the sum-product algorithm", In: Information Theory, IEEE Transactions on, 2001, vol. 47, no. 2, pp. 498-519.

[16] H. A. Loeliger, "An introduction to factor graphs", in Proc. Signal Processing Magazine, IEEE, 2004, vol. 21, no. 1, pp. 28-41.

[17] M. Heerink, B. Kröse, V. Evers and B. Wielinga, "Assessing Acceptance of Assistive Social Agent Technology by Older Adults: the Almere Model", International Journal of Social Robotics, 2010, vol. 2., no. 4, pp. 361-375.

 ISBN: 978-1-61208-341-4