Robotics: Science and Systems 2008
Zurich, CH, June 25-28, 2008

Bridging the gap of abstraction for probabilistic decision making on a multi-modal service robot

Sven R. Schmidt-Rohr, Steffen Knoop, Martin Lösch, Rüdiger Dillmann

***Abstract*— This paper proposes a decision making and control supervision system for a multi-modal service robot. With *partially observable Markov decision processes* (POMDPs) utilized for scenario level decision making, the robot is able to deal with uncertainty in both observation and environment dynamics and can balance multiple, conflicting goals. By using a flexible task sequencing system for fine grained robot component coordination, complex sub-activities, beyond the scope of current POMDP solutions, can be performed. The sequencer bridges the gap of abstraction between abstract POMDP models and the physical world concerning actions, and in the other direction multi-modal perception is filtered while preserving measurement uncertainty and model-soundness. A realistic scenario for an autonomous, anthropomorphic service robot, including the modalities of mobility, multi-modal human-robot interaction and object grasping, has been performed robustly by the system for several hours. The proposed filter-POMDP reasoner is compared with classic POMDP as well as MDP decision making and a baseline finite state machine controller on the physical service robot, and the experiments exhibit the characteristics of the different algorithms.**

I. Introduction

Service robots are meant to act autonomously and robustly in real world environments. Yet, observations of the physical world by robots are limited and noisy, thus the environment is partially observable. Also, the course of events in the real world is never completely deterministic but stochastic. Both aspects of uncertainty need to be regarded by decision making of an autonomous service robot.

In general, decision making of a multi-modal robot uses perceptions of multiple sensors together with background knowledge to choose one of the available actions which will contribute most likely to mission success. The chosen action is performed by coordinating available actuators.

This paper introduces a decision making and supervision system considering uncertainty, which utilizes *partially observable Markov decision processes* (POMDPs) for symbolic, scenario-level decisions. A main focus is bridging the gap of abstraction between symbolic POMDPs and multi-modal, real world perception as well as multiple actuators. Sensor information, including uncertainty, is filtered and fused into belief states. Abstract POMDP decisions are executed by processing sequential task programs to execute more complex and deterministic sub-tasks.

The presented approach is evaluated on a physical, autonomous, anthropomorphic service robot within a realistic waiter cup-serving scenario.

Supported by European Community's projects DexMART, COGNIRON
Institute of Computer Science and Engineering (CSE), University of Karlsruhe, Germany {srsr|knoop|loesch|dillmann}@ira.uka.de

II. State of the Art

Research has approached the challenge of building powerful reasoning systems for real world environments from two directions: construction of reasoning and supervision systems for robots in environments with assumed simplified properties like *fully observable*, *deterministic* or *discrete* on the one hand and the development of probabilistic decision theory coupled with algorithms for decision retrieval on the other hand.

A hierarchical approach for the design of reasoning and control systems for robots has proven to help coping with the complexity of task environments [1]. Three layer architectures are a very popular design [2]. The first layer performs low level processing of sensory data and reactive controlling of actuators. The second layer usually supervises an ongoing task on a symbolic level while the third layer handles the deliberative selection of abstract tasks. The supervisor and deliberative layer of most systems use classical planning which is not aimed at dealing with uncertainty.

Uncertainty in observation and environment dynamics can be handled by using probabilistic techniques. Probabilistic decision theory deals with reasoning of rational agents in the presence of uncertainty. A very promising framework within general probabilistic decision theory are partially observable Markov decision processes (POMDPs), especially the class of discrete, model based POMDPs. A POMDP is an abstract environment model for reasoning under uncertainty [3], [4]. A POMDP models a flow of events in discrete states and discrete time. A specific POMDP model is represented by the 8-tupel $(S, A, M, T, R, O, \gamma, b_0)$. S is a finite set of states, A is a discrete set of actions and M is a discrete set of measurements. The transition model $T(s', a, s)$ describes the probability of a transition from state s to s' when the agent has performed action a. The observation model $O(m, s)$ describes the probability of a measurement m when the intrinsic state is s. The reward model $R(s, a)$ defines the numeric reward given to the agent when being in state s and executing action a. The parameter γ controls the time discount factor for possible future events. The initial belief state is marked by b_0. As POMDPs handle partially observable environments, there exists only an indirect representation of the intrinsic state of the world. In POMDPs, the belief state, a discrete probability distribution over all states in a scenario model, forms this representation. At each time step, the belief state is updated by Bayesian forward-filtering.

A decision about which action is most favorable for the agent when executed next, can be retrieved from a policy

which contains information about the most favorable action for any possible belief distribution. The policy incorporates balancing the probabilities of the course of events into the future with the accumulated reward which has to be maximized.

Computing a policy is computationally challenging and computing exact, optimal policies is intractable [5]. Approximate solutions as *point based value iteration* (PBVI) [6], discrete PERSEUS [7] or HSVI2 [8], however, are quite fast and yield good results for most mid-size scenarios.

For scenarios which can be modelled as *fully observable*, *Markov Decision Processes* (MDPs) can be used for decision making. In MDPs, the decision is derived from a policy function based on the known true world state.

POMDP decision making has already been applied to several different modalities in robotics, like autonomous navigation [9], dialog management [10] and grasping [11], however only for low level controlling of one modality at a time.

These recent investigations encourage to integrate abstract POMDP decision making into reasoning and supervision architectures for autonomous, multi-modal robots now and especially bridging the gap of abstraction between the physical world and discrete decision models. This paper presents such an approach and an evaluation in the following.

III. Approach: *filter*POMDP

When using real world sensory data, there are two possibilities to perform the belief update during scenario runtime when using POMDP decision making. The first is the classical approach, where a distinct observation is perceived and is then processed by using the observation and transition models of a POMDP in a Bayesian update step. The same models are used for calculating the policy. These models must be known a priori and are formulated as classical linear POMDP models. As a drawback, one cannot benefit from any dedicated models of uncertainty which might exist in the lower level algorithms.

The second approach uses dedicated Bayesian filtering methods for each sensor complex, which are then fused into a single belief state. The classic POMDP models are only used for calculating the policy with an approximation of the specific filtering methods. This *filter*POMDP approach has the advantage that the uncertainty, as determined by specific methods, is much more precise concerning the current situation than it could be delivered by a static linear observation model.

Therefore, the question about which approach to take depends on the tradeoff between uncertainty precision concerning the belief and policy precision concerning the process on which the belief is calculated. In mono-modal settings, e.g. mobile robot navigation settings as often used in conjunction with POMDPs, the classical approach is superior, because a general observation and transition model can be set up which models the behavior of both the self-localization process and the general POMDP well. In multi-modal settings, e.g. a service robot with mobility, manipulation, spoken dialogue and visual human activity recognition, the specific uncertainties are better derived from each sensor complex. In this case, the static observation model of the POMDP which is used to calculate the actual policy is in any case a simplification of the process taking place in the specific filters, while the transition model in the POMDP may be able to represent the predictions taking place in the filters.

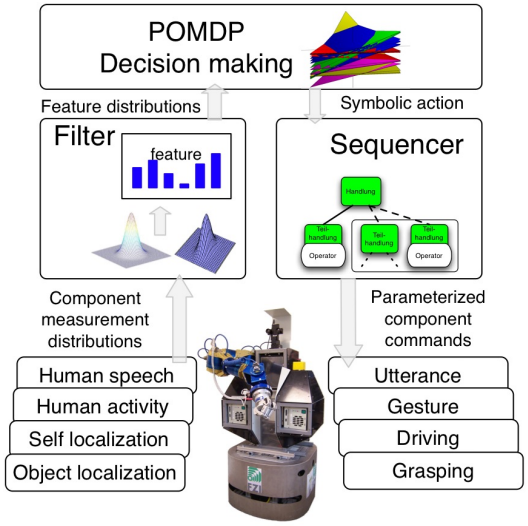

Fig. 1. The system architecture showing the three layer architecture. Low level component control for perception and actuation is at the bottom, the feature filter is on the left, the deliberative layer at the top and the sequencer at the right.

Discrete POMDP models, including all computationally tractable POMDP policy calculation algorithms, for real world scenarios are always approximations of the real dynamics. Thus, in real world scenarios, the policies are an approximation of an ideal policy in both approaches. In practice, in a multi-modal scenario, the approximate POMDP models and policy generated from it will be quite similar, while the belief reflects the true sensor uncertainties better in the second case.

This paper presents the *filter*POMDP approach and compares it in real settings with classic POMDP and MDP reasoning as well as a FSM controller on a multi-modal service robot system.

IV. System architecture

Fig. 1 shows the general architecture of the presented approach. It is designed along the classical structure of a three-layer architecture, with low-level control on the lowest, measurement filtering and execution supervision on the middle and decision making on the top level.

A. Control layer

Three different capability domains exist on the control level: mobility, human-robot interaction and manipulation. Low level modules control the corresponding hardware directly, closely managing low level commands while processing sensor readings and delivering measurements, including uncertainty, to the layer above. The available capabilities

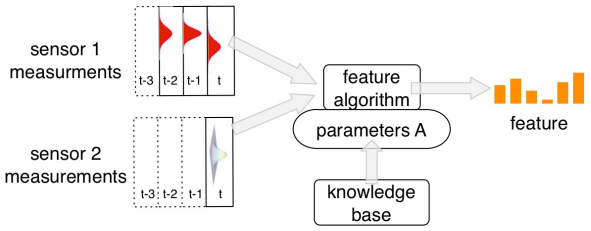

Fig. 2. Schematic computation of a feature.

in the mobility domain are driving and self-localization. Human-robot interaction is possible by speech output, speech recognition, robot arm/hand gestures and human body tracking with symbolic activity recognition by a state-of-the-art procedure [12], [13]. Manipulation capabilities include arm movement, hand grasping as well as force-torque measurements. All measurements delivered by low level modules are probability distributions - either continuous, parametric (e.g. self localization) or discrete, non-parametric (e.g. speech recognition). Actuator commands use a wide range of parameter types - e. g. symbolic utterances or numeric positions.

B. Filter

The filtering module handles processing of measurement data from low level modules. It is basically a modification of the Bayesian forward filter of the POMDP belief state. This forward filter has been sourced out from the main POMDP reasoning, split into a variable number of individual filters, each of which uses an algorithm specialized on a certain observation element and performs a discretization of the continuous data of the logical sensor at the same time. For some components, measurements already include Bayesian updates on the lowest layer, as e.g. for self localization.

Fig. 2 shows the filtering process of an exemplary feature. An individual feature filter takes the data of one or several low level modules, which can also include past observations and applies its specific algorithm, parameterized by the knowledge base. The result is a discrete probability distribution over a set of symbolic categories having their origin in the feature parameter set.

With a new observation, all features have to be updated and after the updates, there exists a new *feature state* which is a set of m discrete probability distributions p, defined over n_j sets of categories $c_{i,j}$ with $i \leq m, j \leq n_j$.

$$\textit{feature state} = \begin{matrix} p(c_{1,1}) & \dots & p(c_{1,n_1}) \\ \dots & \dots & \dots \\ p(c_{m,1}) & \dots & p(c_{m,n_m}) \end{matrix} \quad (1)$$

In the state model, each state in a POMDP scenario model is defined by at least one category from each feature. This mapping connects the abstract, symbolic state through the numerical descriptions of the feature categories to properties of the real world. Given both models, the space of all state models loaded from the knowledge base and the feature state created by feature extraction, the belief state can easily be calculated by calculating the probability b of each state s_k:

$$b(s_k) = \prod_{i=1}^{m} \sum_{j, c_{i,j} \in s_k} p(c_{i,j}) \quad (2)$$

This can also be seen as *fusing* all feature distributions into a single belief state by the means of state descriptions. The set $b(s_1)...b(s_k)$ is the probability distribution of the new belief state.

In the following some specific feature filters are presented.

1) Robot mobility: The low level self-localization module delivers a trivariate gaussian describing position and uncertainty of the robot. The *region occupation* feature filter computes the cumulative distribution function (CDF) of the robot's position distribution for a set of certain, predefined regions. As the orientation is neglected, the CDF over each region is computed over the bivariate by a state-of-the art numerical algorithm [14]. It currently works with rectangles, and larger, complex regions describing a single category can be constructed out of many, smaller cells:

$$p(r_{ij}) = \int_{x_{rp1}}^{x_{rp2}} \int_{y_{rp1}}^{y_{rp2}} N(x, y; \vec{\mu_{pos}}, \Sigma_{pos})\, dx\, dy \quad (3)$$

$$p(c_j) = \sum_{r_i \in c_j} p(r_i) \quad (4)$$

This approach is much more flexible than the usual primitive, one square per state, grid-based one. The resulting discrete probability distribution $p(c_1), ..., p(c_n)$ describes the probability of the robot being in the corresponding region. The regions are defined in the knowledge base.

2) Dialog and human activity filter: POMDPs are also very suitable for spoken dialog [10]. However, because no decision takes place at the filter level, the dialog filter uses a hidden Markov model (HMM), although enhanced with properties found only in MDPs. The dialog HMM model is constructed in the following way:

S is a set of abstract states in which the dialog can be, which reflects human intention. U is a set of possible utterances of the robot, and M is a set of possible human utterances, the robot can detect. $T(s', u, s)$ represents the transition model for each robot utterance, while $O(m, s)$ is the observation model mapping human utterances to states.

Because not any dialog step might consist of alternating human and robot utterances, *idle* utterances fill gaps. In perception, idle utterances are recognized as longer periods of no utterance by either human or robot. The transition model takes into account actions (utterances) of the robot which is not a characteristic in standard HMMs, but in MDPs. A dialog filter probability distribution is forward filtered by the following equation:

$$f'(s') = \alpha(\sum_{m} P(m)\, O(m, s'))\, (\sum_{s} T(s', u, s)\, f(s)) \quad (5)$$

With $P(m)$ being the probability for a specific utterance concerning the last detection as delivered by the speech

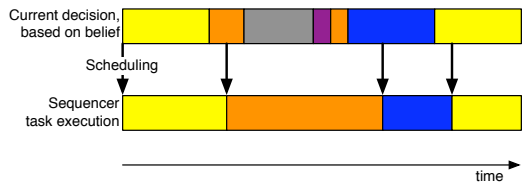

Fig. 3. Connection between reasoner and sequencer. The reasoner continually generates a decision, which is handed to the task sequencer whenever the previous task is finished.

recognition module which delivers discrete probability distributions over recognized utterances. The models for certain dialog scenarios are defined in the knowledge base.

The human activity filter works in the same way. The human activity recognition module delivers discrete probability distributions over a set of perceived, symbolic activities M. U is a set of specific actions of the robot in this case. The states S are a set of true activities.

C. Sequencer

The sequencer receives and processes the commands to execute symbolic programs which represent the actions selected by the reasoner. These symbolic programs represent basic actions for the reasoner. On the sequencing level, they are expanded into complex robot tasks, which are then triggered from the sequencer and executed within the control layer. The task is described as a hierarchical network of basic actions which is processed with a depth-first left-to-right search strategy. A detailed description of the task description called *Flexible Programs* can be found in [15].

The task set in the task database for the presented experiments comprises the tasks *DriveToPos*, *GraspObject*, *SpeakText*, *MonitorHumanActivity* and *PlaceObject*. By decoupling the atomic sensor and actuator controlling from abstract reasoning, it is possible to reduce the decision state space to computationally reasonable dimensionality. The reasoning system decides on the global task to be carried out, while the sequencer performs the actual subtasks that reach the associated goal.

The connection between reasoner and sequencer is depicted in fig. 3. The reasoning system continually generates a decision for a most promising action, based on the current belief state. This decision is passed to the sequencer for execution each time the previous task in the sequencer is finished. Execution of an action may take a non-fixed time span (e.g. *DriveToPos*), while a sensory filter update is quite fast (self-localization, speech recognition etc.). The feature filter runs at a frequency of 20 Hz and updates the belief many times during execution of an action. However, as each action is performed completely and the new action is chosen based on the latest belief, the discrete-step (PO)MDP principle is not violated, while the latest belief reflects the world more precisely.

D. Reasoner

The deliberative layer utilizes established algorithms for discrete decision making: it takes a concrete belief state or a state distribution as input, uses it to query a policy and retrieves a favorable action to perform. The action command is sent to the sequencer to be processed there. The policy for a scenario model can either be computed offline or incrementally calculated online by the PBVI anytime algorithm.

For the results and comparison presented in this paper, MDP and POMDP decision making algorithms are used on the reasoning level.

V. Model design

A fundamental and mostly unsolved question is how to obtain the observation and transition models for real-world scenarios. For model generation, we propose a rule-based approach.

The stochastic behavior of a single feature can often be described by parametric rules (e.g. taking the distance into account when modeling the uncertainty in driving around). Based on the definition of the states, actions and features of the scenario, we define parametric rules as Rule $= \{M, S, S', F, Op, P\}$ with M corresponding matrix, S corresponding origin state, S' target state, F sub state space, Op operation name and P the parameters. Wildcards can be used in M, S, S' which applies a rule to a row, column, matrix or the whole transition model.

The application of such rules shall be addressed by using an example of those actually used in our system. The mobile platform with topological navigation on a graph is more likely not to reach the goal when driving long sections and over many nodes. It is more likely to get stuck in the origin or close to the goal than in between, however most of the time it reaches a goal successfully. Applied to a POMDP transition model T this means that for all actions including a *Goto* command a_g, transition probabilities between states which represent different locations have to include the aforementioned characteristics. It can be achieved by two functions, one realizes getting stuck in the origin: $b(i,j,loc) = \begin{cases} p(\text{stuck}), & \text{if } s_i = s'_j \\ 0, & \text{otherwise} \end{cases}$ and one calculates the likelihood to end up at the goal $p(g)$ and less likely, depending on distance d, somewhere close before, with scaling $\phi()$:

$$b(i,j,loc) = \begin{cases} \max\left(0, p(g) - \phi(d)\right), & \text{if } s'_j \text{ on shortest} \\ & \text{path to goal from } s_i \\ 0, & \text{otherwise} \end{cases}$$

A few dozen rules can describe a quite complex scenario with non-uniform transition models containing several thousand probability entries.

VI. Experiments and results

A. Scenario design

The system was evaluated on an anthropomorphic robot acting in a typical service scenario where the mission is to fetch a cup for persons expressing their interest and bring it to a location they choose. The robot can move to several locations or wait for humans interested in interaction. In case

a person interacts, the robot can find out if it shall fetch a cup or not. When finally holding the cup, it can interact with the person to find out where to bring the cup. The scenario design is open, not strictly sequential, thus the interaction can stop at any moment, as the human might leave. Therefore, the human behavior is modeled as stochastic - the intention of a person is not deterministically predictable. The duration of a scenario is not fixed, as the robot may wait for humans and serve cups indefinitely.

The scenario is realized as a POMDP model with specific states and actions, as well as dynamics of uncertainty. Complex, deterministic tasks are modeled as single actions (Flexible Programs) within the POMDP decision process, taking advantage of the sequencer. There are 11 actions, consisting of idleing, driving to different locations, human activity information gathering, utterances of the robot as well as pick and place actions. The state space is composed of the sub-spaces of self-localization, dialog and human activity. By combining redundancy in the state-space, the state-space can be reduced to 28 distinct and relevant states.

The observation model is derived from rules describing the uncertainty of self-localization, the speech recognizer and the human activity recognizer. The transition probabilities model the behavior of each action, e.g. glitches in moving and also stochastic human behavior in the scenario. Finally, the reward model contains the rules to define the mission motives of the robot. Actions (except idle) cost a small penalty while fetching the cup when desired as well as delivering it to the correct location give a reward.

The POMDP policy is generated from this model and used for decision making during the experiments.

B. Setup for experiments with a physical robot

To evaluate the presented *filter*POMDP approach on a physical robot, it had to be controlled exclusively and completely autonomously by the presented system, running on onboard-computers, with the sole input being the sensor domains as presented in sec. IV. For evaluation, the system was also provided with the ability to use the MDP as well as the classical POMDP approach for decision making while using the same models and in the POMDP case the same policy. In these cases, the observation probability distributions of the sensor complexes were discarded and instead the observation with the highest probability assumed as distinct measurement. Additionally in one experiment, the robot was controlled by an enhanced version of Flexible Programs exclusively, which include dynamic branching and recursive tree expansion, being able to model a complete finite state machine (FSM).

Because of the open-ended nature of a scenario, time is the only relevant measure of duration of an experiment when comparing gathered rewards. In case a method takes a lot of reassurance actions to determine a human intention, it will be able to fetch and bring a cup less often. Thus, all four experiments had exactly the same duration and starting condition. Concerning the behavior of interacting persons during experiments, it was assured that the behavior of the human corresponded on average to the probabilities in the transition model. Finally, true requests and actual robot behavior were recorded by a human supervisor.

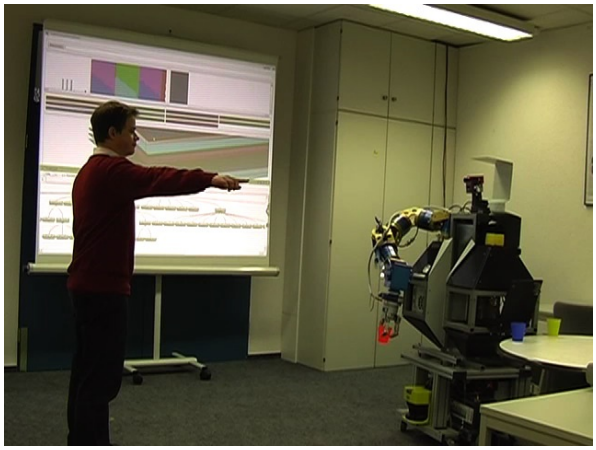

Fig. 4. The robot during experiments, after fetching a cup and awaiting a destination order. The camera-head contains a stereo-color-camera and a 3D-time-of-flight camera for human activity recognition. Speech recognition is performed using an onboard microphone. The platform uses a laser-scanner for self-localization. A live-visualisation of the belief state, a 3D cut of the 28D value function and the current Flexible Program as sent from the robot over wireless can be seen projected at the background.

C. Results

A representative set of four runs with our robot Albert 2 (see fig. 4), each lasting exactly 30 minutes, but controlled by different methods: FSM, MDP, classical POMDP and *filter*POMDP shall be further analyzed to compare the techniques. The following tables show these correlations between the action requested (Req.) and the actually performed behavior (Perf.) for the most important parts of the scenario. Desired behavior is shown on the main diagonal, the first column shows reassurance actions of all kinds, while other entries indicate bad behavior.

1) Finite state machine experiment:

Req. \ Perf.	Reassure	Fetch cup	Put to A	Put to B
Other	0	1	0	1
Fetch cup	3	4	0	0
Put to A	5	0	2	0
Put to B	7	0	0	2

While the state machine does not make many big mistakes, it is conservative and annoys the human with many reassurance questions. Thus, it is not able to perform many delivery actions in the given time, as it has no inherent risk assessment as POMDP methods.

2) MDP experiment:

Req. \ Perf.	Reassure	Fetch cup	Put to A	Put to B
Other	0	0	0	3
Fetch cup	0	8	0	0
Put to A	5	0	2	1
Put to B	2	0	0	2

First, it should be noted that the reassurance actions in this case are performed, when only one of the two human

indicators (*say* and *point*) of the intention where to bring the cup, are present. The MDP has problems with contradicting *point* and *say* indicators because as soon as one of the two indicators is measured with highest probability - even if it is only slightly ahead of the other - the MDP will decide for the wrong location. In that case a POMDP still performs information gain actions.

3) Classical POMDP experiment:

Req. \ Perf.	Reassure	Fetch cup	Put to A	Put to B
Other	0	4	0	1
Fetch cup	1	5	0	0
Put to A	6	0	4	1
Put to B	3	0	0	3

In the POMDP, reassurance actions also include actions indicating typical POMDP information gain. Concerning the location to bring the cup to, the POMDP performs information gain until it is quite sure that both indicators (*say* and *point*) refer to the same location, thus it performs better than the MDP. However, it tends to bring the cup even when not requested, because of the reliance on a uniform distribution concerning prediction of human intention and the static observation model when calculating the belief.

4) filterPOMDP experiment:

Req. \ Perf.	Reassure	Fetch cup	Put to A	Put to B
Other	0	0	0	1
Fetch cup	1	9	0	0
Put to A	3	0	5	1
Put to B	1	0	0	2

The *filter*POMDP shows to match both the strong points of the MDP and the classical POMDP. When being requested to fetch the cup, the decision is made, based on the actual speech recognition uncertainty. In the case of deciding where to bring the cup, it also performs information gain, but not as often as the classical POMDP, because there is more precise knowledge about current uncertainty.

5) Comparison:

Type \ Method	FSM	MDP	POMDP	fPOMDP
Correct fetch/put	8	12	12	16
Incorrect fetch/put	2	4	6	2
Reassurance	15	7	10	5

The table shows the performance summary. The FSM is very conservative, as it has no dynamic risk assessment, the MDP has more problems with initial human interaction than the POMDP, thus it wastes time and the total number of fetch/put actions is slightly smaller.

D. Discussion

As shown by the results, exploiting available information about specific uncertainty of current perceptions can be beneficial for decision making by a multi-modal service robot. Static POMDP observation models only contain information about average uncertainty, not about the current one. Using available dedicated methods for determining this uncertainty is especially promising with multi-modal robots where complex perceptive components like speech recognition or human activity recognition exist. However, the POMDP model is still valid for policy generation as it has to use average observation uncertainties. This also holds for the transition model.

VII. Conclusion and outlook

As shown, using the reasoning capabilities of POMDPs in real robot scenarios is promising as it leads to very robust reasoning in partially observable and stochastic domains. This work has presented a feature filter concept which incorporates sensor uncertainty directly, instead of using only an indirect, static observation model for obtaining belief states. The main remaining challenge is obtaining the transition and observation models for arbitrary scenarios. As there has been only very little research to learn models of real world domains for discrete, model based POMDPs so far [16], this should be pursued intensively now.

References

[1] R. Simmons, R. Goodwin, K. Z. Haigh, S. Koenig, and J. O'Sullivan, "A layered architecture for office delivery robots," in *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, W. L. Johnson and B. Hayes-Roth, Eds. New York: ACM Press, 5–8, 1997, pp. 245–252.

[2] E. Gat, "On three-layer architectures," *Artificial Intelligence and Mobile Robots. MIT/AAAI Press*, 1997.

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

[4] 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.

[5] 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.

[6] 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.

[7] M. Spaan and N. Vlassis, "Perseus: Randomized point-based value iteration for pomdps," *Journal of Artificial Intelligence Research*, vol. 24, pp. 195–220, 2005.

[8] 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.

[9] A. Foka and P. Trahanias, "Real-time hierarchical pomdps for autonomous robot navigation," *Robot. Auton. Syst.*, vol. 55, no. 7, pp. 561–571, 2007.

[10] J. D. Williams, P. Poupart, and S. Young, "Using factored partially observable markov decision processes with continuous observations for dialogue management," Cambridge University Engineering Department Technical Report: CUED/F-INFENG/TR.520, Tech. Rep., March 2005.

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

[12] S. Knoop, S. Vacek, and R. Dillmann, "Sensor fusion for 3d human body tracking with an articulated 3d body model," in *Proceedings of the 2006 IEEE International Conference on Robotics and Automation (ICRA)*, Orlando, Florida, 2006.

[13] M. Lösch, S. Schmidt-Rohr, S. Knoop, S. Vacek, and R. Dillmann, "Feature set selection and optimal classifier for human activity recognition," in *Robot and Human Interactive Communication 2007 (ROMAN 2007)*, 2007.

[14] A. Genz, "Numerical computation of rectangular bivariate and trivariate normal and t probabilities," *Statistics and Computing*, vol. 14, pp. 151–160, 2004.

[15] S. Knoop, S. R. Schmidt-Rohr, and R. Dillmann, "A Flexible Task Knowledge Representation for Service Robots," in *The 9th International Conference on Intelligent Autonomous Systems (IAS-9)*, 2006.

[16] R. Jaulmes, J. Pineau, and D. Precup, "Active learning in partially observable markov decision processes," in *ECML*, 2005.