Rapid specification and automated generation of prompting systems to assist people with dementia

Jesse Hoey\textsuperscript{a,*}, Thomas Plötz\textsuperscript{b}, Dan Jackson\textsuperscript{b}, Andrew Monk\textsuperscript{c}, Cuong Pham\textsuperscript{b}, Patrick Olivier\textsuperscript{b}

\textsuperscript{a}School of Computing, University of Dundee, Dundee, DD1 4HN, Scotland
\textsuperscript{b}School of Computing Science, University of Newcastle, Newcastle upon Tyne, NE1 7RU, UK
\textsuperscript{c}Department of Psychology, University of York, York, YO10 5DD, UK

Abstract

Activity recognition in intelligent environments could play a key role for supporting people in their activities of daily life. Partially observable Markov decision process (POMDP) models have been used successfully, for example, to assist people with dementia when carrying out small multi-step tasks such as hand washing. POMDP models are a powerful, yet flexible framework for modeling assistance that can deal with uncertainty and utility in a theoretically well-justified manner. Unfortunately, POMDPs usually require a very labor intensive, manual setup procedure. This paper describes a knowledge driven method for automatically generating POMDP activity recognition and context sensitive prompting systems for a complex tasks. It starts with a psychologically justified description of the task and the particular environment in which it is to be carried out that can be generated from empirical data. This is then combined with a specification of the available sensors and effectors to build a working prompting system. The method is illustrated by building a system that prompts through the task of making a cup of tea in a real-world kitchen. The case is made that, with further development and tool support, the method could feasibly be used in a clinical or industrial setting.

Keywords:
Behavior analysis, dementia, activity recognition, prompting systems, POMDP

1. Introduction

1.1. Dementia: A suitable case for pervasive computing

Dementia is an important problem with serious effects on the society because of the changing demographic towards an aging population. The dependency ratio (the ratio of those typically not in the labor force and potentially needing care to those typically in the labor force and thus able to provide care) is increasing. For example, the old age dependency ratio (the projected number of persons aged 65 and over expressed as a percentage of the projected number of persons aged between 15 and 64) in the European Union is projected to increase from 25.9 in 2010 to 50.4 in 2050 [1]. At the same time the number of people with dementia is increasing. The number
of persons with Alzheimer’s disease worldwide, for example, is expected to double and will top 100 million by the year 2050 [2]. This means that the burden of care will have to shift from the professional arena (e.g. hospitals and clinics) into the home and community. In Canada, for example, the number of hours of informal (i.e., unpaid) care is expected to triple from 231 million hours to 756 million hours by the year 2038 [3].

Many people with dementia wish to remain living in their own homes as long as possible. However, they generally require some assistance in order to do so. Difficulties performing activities of daily living at home, such as preparing food, washing themselves, or cleaning, may trigger the need for personal assistance or relocation to residential care settings [4]. Moreover, it is associated with diminished quality of life, poor self-esteem, anxiety, and social isolation for the person with dementia and their caregiver [5].

Technology to support people in their need to live independently is currently available in the form of personal and social alarms and environmental adaptations and aids. Looking to the future, we can imagine intelligent, pervasive computing technologies using sensors and effectors that help with more difficult cognitive problems in planning, sequencing and attention. A key problem in the construction of such intelligent technologies is the automatic analysis of people’s behaviors from sensory data. Activities need to be recognized and – by incorporating domain specific expert knowledge – reasonable conclusions have to be drawn which ultimately enables the environment to perform appropriate actions through a set of actuators. In the example of assisting people with dementia, the smart environment would prompt whenever the residents get stuck in their activities of daily living.

1.2. Automatic generation of prompting systems for people with dementia

The technical challenge of developing useful prompts and a sensing and modeling system that allows them to be delivered only at the appropriate time is hard but achievable. Certainly the most sophisticated of these is the COACH system [6]. COACH uses computer vision to monitor the progress of a person with dementia washing their hands and prompts only when necessary. COACH uses a partially observable Markov decision process (POMDP), a temporal probabilistic model that represents a decision making process based on environmental observations. The COACH model is flexible in that it can be applied to other tasks [7, 8]. However, each new task requires substantial re-engineering and re-design to produce a working assistance system, which currently requires massive expert knowledge for generalization and broader applicability to private home scenarios. An automatic generation of such prompting systems would substantially reduce the manual efforts necessary for creating assistance systems, which are tailored to specific situations and tasks, and environments. In general, the use of a-priori knowledge in the design of assistance systems is a key unsolved research question. Researchers have looked at specifying and using ontologies [9], information from the Internet [10], logical knowledge bases [11, 12], and programming interfaces for context aware human-computer interaction (HCI) [13].

People with dementia who are living at home will generally have mild to moderate dementia and many will be living with a spouse or relative. The first design challenge for an automatic assistance system is to understand the context in which they are living and to select tasks that are important to people with dementia and their carers. Previous work [14] has identified simple kitchen tasks as being particularly important in this respect, for example, making a hot drink. Occupational therapists and psychologists have provided systematic data describing how people with dementia behave when carrying out domestic tasks, e.g., [15]. Strategies of support focus on utilizing the skills and abilities that remain. For example, a person with dementia may be able to rely on automatic rather than controlled processes [16]. The importance of contextual
cues for people with dementia has also been well documented [17] and the role of such cues is widely acknowledged in dementia care practice. Home-based interventions often involve environmental interventions, or “stimulus control” [18], such as the removal of clutter and the display of task relevant items (e.g. placing the clothing to be worn on the bed).

There is thus a body of work that can inform decisions regarding what tasks are most useful to support and the prompting strategies that are likely to be most fruitful. In addition, POMDP models have been demonstrated as being capable of modeling real life tasks and prompt generation. However, one problem remains if this approach is to be put forward as a practical solution for people with dementia living in their own homes: how to obtain the context sensitive prompting models? This paper will focus on assistance with simple kitchen tasks. Every kitchen is different. The appliances, tools and ingredients used, and where everything is kept in the kitchen will vary from home to home. There is considerable variation in the tasks that people will choose and the ways in which they will want to carry them out. The COACH system took several man-years to build and only does hand washing in a particular washroom, how will it be practical to build an activity recognition and prompting system for each new task and kitchen? The key to proper assistance systems is the incorporation of psychologically justified expert background knowledge. This knowledge must be integrated as early as possible in the design phase of such systems. The method we propose provides a principled approach to integrate this background (re-usable) knowledge with the specific knowledge about a particular user from a person who has access to their daily practice.

1.3. Contributions & Outline

In this paper we present a knowledge driven method for automatically generating POMDP activity recognition and context sensitive prompting system for a kitchen task, as an example. The developed approach starts with a description of a kitchen task and the kitchen in which it is to be carried out that is relatively easy to generate. Interaction Unit (IU) analysis [19], a psychologically motivated method for transcoding interactions relevant for fulfilling a certain task, is used for obtaining a formalized, i.e., machine interpretable task description. This is then combined with a specification of the available sensors and effectors to build a working model that is capable of analyzing ongoing activities and prompting someone. The method is illustrated by building a system to prompt someone through the task of making a cup of tea in a particular kitchen.

The long-term goal of this approach is to allow end-users, such as health professionals, to specify and develop their own context sensitive prompting systems for needs as they arise. In the fullness of time, a proper evaluation would involve implementing the idea in a clinical or commercial service and then evaluating that service in terms of the value to a specific population of people that it is intended to serve. At this stage we are just concerned with demonstrating: (i) that it is possible to automatically generate a working POMDP model from a description of a task and the kitchen in which it is to be carried out and a set of sensor specifications, and (ii) that the method could feasibly be developed into a practical service for people with dementia.

Sections 2 to 5 describe the method and make the case that, with further development and tool support, it could feasibly be used in a clinical or industrial setting. To make this demonstration convincing we have chosen a task that has not been modeled in a POMDP before, that is making tea, and a set of sensors rather different to the computer vision algorithms used in COACH. The POMDP prompting system built in the process of developing this method is demonstrated in Section 6 and shown to behave appropriately.
2. Overview of the method

2.1. Partially observable Markov decision process models

A POMDP is a probabilistic temporal model of a system interacting with its environment [20], and is described by (1) a finite set of state variables, the cross product of which gives the state space, \( S \); (2) a set of observation variables, \( O \) (the outputs of some sensors); (3) a set of system actions, \( A \); (4) a reward function, \( R(s, a, s') \), giving the relative utility of transiting from state \( s \) to \( s' \) under action \( a \); (5) a stochastic transition model \( Pr : S \times A \rightarrow \Delta S \) (a mapping from states and actions to distributions over states), with \( Pr(s'|s, a) \) denoting the probability of moving from state \( s \) to \( s' \) when action \( a \) is taken; and (6) a stochastic observation model with \( Pr(o|s) \) denoting the probability of making observation \( o \) while the system is in state \( s \). Figure 1(a) shows a POMDP as a Dynamic Bayesian network (DBN) with actions and rewards, where arrows are interpretable as causal links between variables. The model is typically shown using only two time slices representing a transition from \( s \) to \( s' \) when action \( a \) is taken, with primed variables \( (s') \) denoting the post-action states and unprimed variables \( (s) \) denoting pre-action states. POMDPs can be defined with continuous or discrete variables, observations and actions. In this paper, however, we assume all are discrete-valued to enable efficient solutions.

Since the system state is not known with certainty, a policy, \( \pi \), maps either belief states (i.e., distributions over \( S \)) or action observation histories into choices of actions. A policy is computed to maximize some aggregate measure of utility over time. One such measure is the expected discounted sum of rewards, \( \sum_t \gamma^t r_t \), where \( r_t \) is the reward obtained at time \( t \), and \( \gamma \in [0, 1] \) is a discount factor that makes large and distant (in time) rewards equivalent to small and immediate rewards. We will not delve into details of POMDP solution methods, but note that current research has enabled the approximate solution of very large POMDPs, and we refer to [21] for an overview of POMDP concepts and algorithms. We compute approximate policies for the POMDPs in this work using the SymbolicPerseus package [22]. It implements a factored, structured point-based approximate solution technique based on the Perseus algorithm [23].

![Figure 1: Two time slices of (a) a general POMDP and (b) a factored POMDP for modeling interactions with cognitive assistive technology.](image)
POMDPs can be used to monitor a person’s progress in a task by using Bayes’ rule to update a belief state, \( b(s) \), that gives a probability that the model is in state \( s \). If the system is in belief state \( b(s) \), action \( a \) is taken and \( o \) is observed from the sensors, then the new belief state, \( b(s') \) is computed using

\[
b(s') = P(s'|a, o, b(s)) = \sum_s P(s', s|a, o, b(s)) = \sum_s \frac{P(o|s', a)P(s'|s, a)b(s)}{\sum_{s'} P(o|s', a)P(s'|s, a)b(s)}
\]  

which involves a multiplication of the belief state by the transition and observation functions and summing over the states, \( s \).

The progression of this belief state through time is what the POMDP attempts to optimize by taking actions that lead to belief states with high reward values. A POMDP operates on an ’event-driven’ time scale, with each event causing an update to the POMDP. Events can be actions of the user as signaled by the sensors, or can be from a timeout on a real clock. The timeout is necessary to detect when a user does ’nothing’.

### 2.2. Partially observable Markov decision process models for assisting people with dementia

A POMDP for assistance of persons with dementia breaks the state space down into three key factors as shown in Figure 1(b): states describing elements of the functional task in the real world, \( T \), e.g. whether the water has been boiled or not (the ”task factor model”), states capturing the user’s cognitive capacities, \( Y \), e.g., to remember what they are supposed to do next (the ”ability factor model”), and states capturing an inferred history of what the user has actually done since the last update, \( B \), e.g. fill the kettle (the ”behavior factor model”). We use the word ”behavior” here to describe actions of the user to distinguish them from actions of the system (i.e., prompts to the user). A preliminary version of this model was explored in [24]. As we will see in Section 5, these three factors relate to the psychological analysis, and keeping them separate will ease the translation between the two.

Jointly, \( S = \{T, B, Y\} \) is known as the state. The transition function can be factored as

\[
Pr(S'|S, A) = Pr(T', B', Y'|T, B, Y, A) = Pr(T'|B', Y', T, B, Y, A)Pr(B'|T', Y, T, B, Y, A)Pr(Y'|T, B, Y, A)
\]

Notice that the task state, \( T' \), is independent of the ability, \( Y' \), given the behavior, \( B' \). The idea is that changes in the task states are caused by the behaviors of the user (and possibly the system actions), independently of the user’s ability given the behaviors. The factorization in Equation (4) breaks down the problem of describing the complete model dynamics \( Pr(S'|S, A) \) into three smaller problems of specifying each of the factors. This breakdown is exactly what allows for the efficient specification for a particular task, and will be returned to in Section 5.

The observations \( O \) and \( O' \) are states of the sensors at times \( t-1 \) and \( t \). It is convenient to divide \( O \) into the states of sensors relevant to the task environment \( K \) and states of sensors relevant to user behavior \( V \). Whereas the first type of sensors monitors, for example, movements of objects in the environment, the latter corresponds to virtual sensors that, for example, monitor the activities the acting person pursues. So \( O = \{K, V\} \) are assumed to be generated by the task and behavior variables, \( T \) and \( B \), respectively, through some observation functions \( Pr(K|T) \) and \( Pr(V|B) \).
The system’s actions are various prompts or memory aids the system can use to help a user remember things in the task. For persons with dementia, simple audio prompts are often very effective, and are used in the COACH system along with video demonstrations [25, 6]. Finally, the observations are any information from sensors in the environment that can give evidence to the system about the user’s behaviors or the task.

2.3. Generating a psychologically justified model for each new task and environment

A key problem is the initial specification of the model for a particular task. The COACH system has been specified manually by technology designers after consultation with end users (persons with dementia and carers) over a number of years. This process is extremely time-consuming and difficult to generalize across tasks. In this paper, we describe a method to automatically generate a POMDP-based prompting system using a description that captures the circumstances of a particular user, task, and environment. This description is "syndetic" as it describes the conjunction of cognitive and environmental precursors for each action. We call the resulting prompting system a SNAP (SyNdetic Assistance Process).

The methodology is an interaction between two designers: a trained human factors annotator, and a ubiquitous sensing technician. First, the human factors annotator uses the syndetic Interaction Unit (IU) modeling technique described below, and applies it to videos showing a person attempting to do the task with the help of a human assistant. The end-result is an IU analysis that uncovers the various elements of the task, the user’s cognitive abilities, and the user’s behaviors. The analysis also implicitly includes a specification of assistance actions that should be taken, one for each user ability that is required to complete the task. Second, the ubiquitous sensing technician considers the states, behaviors, and actions specified by the human factors annotator, and proposes a set of sensors and actuators that can be retrofitted to the user’s environment for the particular task. The ubiquitous sensing technician then provides a specification of the sensors that consists of three elements: (1) a name for each sensor and the values it can take on (e.g. on/off); (2) a mapping from sensors to the states and behaviors as given by the human factors annotator showing the evidentiary relationships, and (3) measurements of each sensor’s reliability at detecting the states/behaviors it is related to in the mapping. The ubiquitous sensing technician also implements a simple software input-output interface to the sensors and actuators that can be used during execution.

The IU analysis and the sensor specification are then fed into a software that automatically generates a SNAP POMDP for this particular user/task/environment combination. The SNAP POMDP is solved to yield a policy of action, and a generic software controller then runs the SNAP POMDP, taking input from the sensors through the ubiquitous sensing technician’s interface. In the following, we first describe the IU analysis (Section 3), followed by the sensor modeling (Section 4), and the conversion to a SyNdetic Assistance Process (Section 5).

3. Specifying the task and environment: Interaction Unit Analysis

The starting point for the automatic generation of a POMDP prompting system is a psychologically justified description of the task and the particular environment in which it is to be carried out. To illustrate the method we are using we thus needed a real example of someone with dementia carrying out a real task that has not previously been modeled using a POMDP. Serendipitously we had access to videotapes of the same woman (JF) making a cup of tea on two occasions in her own kitchen. These are part of a collection that was the basis of an analysis
of the problems that people with dementia have with kitchen tasks [26]. JF has dementia of the Alzheimer’s type and lives with her husband who now does all the cooking. They store tea and coffee making items on a tray on the counter as they believe this helps her when making hot drinks for herself. She can do other tasks alone (e.g., dressing and cleaning). They lived in the same house before the onset of dementia when she used to do all the kitchen tasks.

The POMDP prompting system was built into the Ambient Kitchen, a high fidelity prototyping environment for pervasive technologies [27] at Newcastle University (figure 2). The videos were used to select appliances and utensils similar to those used by JF. Note that, in the ideal case, the POMDP prompting system would be retrofitted into JF’s kitchen (from where the videos originated). For the purposes of this paper, we are going to implement the POMDP in the Ambient Kitchen as a demonstration of the feasibility of the approach.

![Figure 2: Set-up of the test environment used for the SNAP evaluation. Left: Overview of the Ambient Kitchen in Culture Lab at Newcastle University. Right: Close-up of the main work surface for the tea preparation task, from left-to-right: milk bottle, cup (with spoon and tea bag), sugar box, tea box and kettle.](image)

### 3.1. Choice of notation

The job of the Human Factors Annotator is to describe the task that the user wishes to carry out with the help of the prompting system, along with the particularities of the kitchen environment in which it is to be done. This specification has to be sufficiently detailed, and in a format suitable, for the automatic generation of a POMDP.

Task analysis has a long history in Human Factors [28] and Occupational Therapy [29] where tasks are referred to as “activities of daily living” (ADL). In both cases the emphasis is on describing the actions taken by a user and the intentions (goals and sub-goals) that give rise to those actions. There has been less emphasis on how actions are driven by the current state or changes in the environment. Syndetic modeling [30] remedies this omission by describing the conjunction of cognitive and environmental precursors for each action. Modeling both cognitive and environmental mechanisms at the level of individual actions turns out to be much more efficient than building separate cognitive and environmental models [19].

Ryu and Monk [31, 19] propose IU Analysis as a syndetic notation that allows a user interface developer to make explicit the elements that stimulate and are changed by a user action. At the same time the developer specifies the assumed goals and sub-goals that are necessary for that action to occur and how they are changed when the action is taken. The notation thus allows the developer to model the conjunction of cognitive and environmental pre-conditions and post-conditions at the level of individual actions. These elements together specify an IU. Ryu and Monk [19] then specify rules for seeking out assumptions that might lead to user errors so that
the developer can refine the design. The notation is proposed as a tool for use in the development of novel interaction techniques.

The approach taken here uses an adapted form of the IU analysis notation to build a model of the user task and how it is to be achieved in a particular kitchen. The analysis starts from a video of a user in their natural environment (e.g., home), and makes a list of necessary user actions divided into subtasks and a specification of the environment in terms of a state space. IUs are then written that specify the conjunction of cognitive and environmental pre- and post-conditions for each action. The state space and IUs are then refined in an iterative process.

3.2. Making a list of user behaviors

In the video tapes analyzed to produce an IU analysis, JF made errors and occasionally had to be prompted, so the first step was to make an 'ideal’ sequence of behaviors, i.e., the simplest sequence that would have got the job done in that kitchen. This is the "behavior factor model” (see Section 2.2) and is given in Table 1. The primitives for describing these behaviors, Move, Alter, etc. were adapted by Wherton and Monk [26] from the Action Coding System [32], a notation that is widely used by Occupational Therapists to describe the behavior of patients when carrying out a task. The Human Factors Annotator generates this list by repeatedly watching the video tape in order to generate what is essentially a transcript of the user’s behavior using these action primitives. This transcript is then edited to give the simplest sequence that would have got the task done in that kitchen, while matching the sequences actually generated by the user as closely as possible.

This list of behaviors is then divided up into the logical sub-tasks that structure this task (e.g., preparing the cup, boiling the kettle and so on). The division of a task into logical subtasks is a standard cognitive strategy and people with dementia commonly stall at the boundary between subtasks [16, 33]. For example, they might get the cup ready and then come to a halt because they do not appear to know what to do next. The list of ideal action sequence, as it was divided into subtasks, is provided in Table 1.

3.3. Describing the state space

The video was also used to enumerate the environmental component of the state space (the "task factor model”). States are generally mutually exclusive (e.g., cup position: on tray, on work surface). This is awkward for Cup Contents and these states may be combined (e.g., cup contents: water+teabag). The state space for JF’s kitchen is given in Table 2. One also needs to specify the start (S) and end (E) states, given by superscripts in Table 2.

3.4. Writing the IU analysis

Table 3 gives the full IU analysis for JF making tea. An IU analysis can be thought of as an informal production system. The rightmost column is simply the list of user behaviors from Table 1. The other columns are pre-conditions for the user exhibiting that behavior. The second column (labeled Current Goals) describes the goal stack needed for the user doing that behavior. The goals are all specified as desired states of the environment and so use the state space specified in Table 2. In IU2, for example, there is the overall goal "Final” (the end state described in Section 3.3, having a cup of tea, the teabag in the bin and so on), plus a subgoal to have a teabag in the empty cup on the work surface, "cup TB”. The goals stack for each IU consists of the overall goal ‘Final” and, where appropriate, one of the subgoals specified in Table 1.
Goals

Have a cup with a TB in it on WS (teabagcupws)
- Move cup from tray to WS
- Alter box to open
- Move TB from box to cup
- Alter box to closed

Have boiling water and TB in the cup (kettleboilcup)
- Alter kettle to filled
- Alter kettle to on
- Move water from kettle to cup

Have sugar in cup (sugarcup)
- Alter SC to open
- Take spoon
- Move sugar to cup
- Give spoon to WS
- Alter SC to close

Have milk in cup (milkcup)
- Alter fridge to open
- Take MB
- Alter fridge to closed
- Move milk to cup
- Alter fridge to open
- Give MB to fridge
- Alter fridge to closed

Have cup of tea (without TB) (tbtobin)
- Take spoon from WS
- Take teabag from cup
- Alter cupboard to open
- Alter bin to open
- Give teabag to bin
- Alter cupboard to closed
- Give spoon to WS

<table>
<thead>
<tr>
<th>Goals</th>
<th>Behaviors</th>
</tr>
</thead>
</table>
| Have a cup with a TB in it on WS (teabagcupws) | Move cup from tray to WS
Alter box to open
Move TB from box to cup
Alter box to closed |
| Have boiling water and TB in the cup (kettleboilcup) | Alter kettle to filled
Alter kettle to on
Move water from kettle to cup |
| Have sugar in cup (sugarcup) | Alter SC to open
Take spoon
Move sugar to cup
Give spoon to WS
Alter SC to close |
| Have milk in cup (milkcup) | Alter fridge to open
Take MB
Alter fridge to closed
Move milk to cup
Alter fridge to open
Give MB to fridge
Alter fridge to closed |
| Have cup of tea (without TB) (tbtobin) | Take spoon from WS
Take teabag from cup
Alter cupboard to open
Alter bin to open
Give teabag to bin
Alter cupboard to closed
Give spoon to WS |

Table 1: JF’s ideal behaviors, as it was divided into subtasks. The subtask end goal is shown on the left, with the name used in to refer to it in parentheses. WS = work surface, TB = tea bag; SC = sugar container, MB = milk bottle.

The next column are the critical parts of the current environment, again specified in terms of the states in Table 2. The third column in Table 3 specifies the cognitive factors necessary in the current state of the environment to lead to the behavior. These factors are the basis for the knowledge of where people with dementia will have problems. These are standard terms in human factors and cognitive psychology but bear some explanation here.

Recognition (Rn) requires the user to see or hear something and understand its meaning in the context of this task. Thus in IU20 (see Table 3) the user has to recognize that there is no milk in the tea before they add milk to the cup. This can be contrasted with the cognitive operation recall. Recall (Rl) requires the user to remember something without being able to see the required information directly. The equivalent behavior to add sugar in IU13 will only occur if the user recalls that they have not already done so as whether the cup has sugar cannot be ascertained by visual inspection. Recall is generally a problem for people with dementia and we expect most errors to occur for IUs containing recall operations. Recognition is only generally a problem if the object has low saliency, is unfamiliar or is surrounded by similar objects, particularly if the
user has semantic memory problems.

*Affordance (Af)* implies recognition of the meaning of some part of the environment in terms of an action it makes possible. Thus it includes an implied recognition operation. Affordance is shorthand for “recognizing the affordance for taking the action specified in the next column”. For example IU4, moving the teabag from the tea box to the cup has the precondition"Af TB in box cup”. Affordance, like recognition, is a cognitive process that only presents problems for people with dementia if the object concerned is unfamiliar or has low saliency.

The IU analysis codes the effect of the action on the goals stack and the environment implicitly, through the new goal stack and critical environment in the subsequent IU. Thus, each group ends in a dummy IU that shows the goal stack and environment after the last action.

Similarly, each group of IUs starts with a special IU with the cognitive operation to recall the step (‘Rl step’) where the only pre-conditions are some element of the environment and the knowledge that they are making a cup of tea. This IU has no action and simply serves to push the goal for this subtask. People with dementia have trouble with recalling steps and will often stop behaving at the junction between subtasks.

We will see in Section 5 how the environment will map to the task factors of the POMDP, the Rn/Rl/Af cognitive factors map to the ability factors in the POMDP, and the user behavior maps to the behavior factors in the POMDP. The goals will map to the reward function, and the sequence of rows in Table 3 will correspond with the transition function of the POMDP.

### 3.5. Specifying initial cognitive abilities

The SNAP POMDP includes a state describing the users cognitive ability for each cognitive operation in Table 3 (the "ability factor model" described in Section 2.2). These are adjusted as the SNAP observes the user’s behavior through the sensors but one needs to set initial values.

As mentioned above, any IU with a recall cognitive operation is likely to give someone with dementia problems and hence would be set with a low initial ability value. The five 'Rl step'

<table>
<thead>
<tr>
<th>State Variable</th>
<th>values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cup position</td>
<td>on tray&lt;sup&gt;S&lt;/sup&gt; on WS&lt;sup&gt;E&lt;/sup&gt;</td>
</tr>
<tr>
<td>Cup contents</td>
<td>empty&lt;sup&gt;S&lt;/sup&gt; TB water tea&lt;sup&gt;E&lt;/sup&gt; milk&lt;sup&gt;E&lt;/sup&gt;</td>
</tr>
<tr>
<td>Box position</td>
<td>on tray&lt;sup&gt;S&lt;/sup&gt;E on WS</td>
</tr>
<tr>
<td>Box condition</td>
<td>open closed&lt;sup&gt;E&lt;/sup&gt;</td>
</tr>
<tr>
<td>TB position</td>
<td>in box&lt;sup&gt;S&lt;/sup&gt; on WS in hand in cup in bin&lt;sup&gt;E&lt;/sup&gt;</td>
</tr>
<tr>
<td>TB condition</td>
<td>unused&lt;sup&gt;S&lt;/sup&gt; used&lt;sup&gt;E&lt;/sup&gt;</td>
</tr>
<tr>
<td>Spoon</td>
<td>on tray&lt;sup&gt;S&lt;/sup&gt; in hand on WS&lt;sup&gt;E&lt;/sup&gt;</td>
</tr>
<tr>
<td>Kettle position</td>
<td>on stand&lt;sup&gt;S&lt;/sup&gt;E in hand</td>
</tr>
<tr>
<td>Kettle condition</td>
<td>empty&lt;sup&gt;S&lt;/sup&gt;E full cold full hot</td>
</tr>
<tr>
<td>Kettle power</td>
<td>on off&lt;sup&gt;E&lt;/sup&gt;</td>
</tr>
<tr>
<td>MB</td>
<td>in fridge&lt;sup&gt;S&lt;/sup&gt;E in hand on WS</td>
</tr>
<tr>
<td>Cupboard</td>
<td>open closed&lt;sup&gt;E&lt;/sup&gt;</td>
</tr>
<tr>
<td>Bin</td>
<td>open closed&lt;sup&gt;E&lt;/sup&gt;</td>
</tr>
<tr>
<td>SC position</td>
<td>on tray&lt;sup&gt;S&lt;/sup&gt;E on WS</td>
</tr>
<tr>
<td>SC condition</td>
<td>open closed&lt;sup&gt;E&lt;/sup&gt;</td>
</tr>
</tbody>
</table>

Table 2: State space for JF’s kitchen. Start states are signified as S and end states as E
<table>
<thead>
<tr>
<th>IU</th>
<th>Current Goals</th>
<th>Current Environment (critical)</th>
<th>Recognition/Recall/ Affordance</th>
<th>User behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Final</td>
<td>cup empty on tray</td>
<td>阮 cup on tray, RI step</td>
<td>No behavior</td>
</tr>
<tr>
<td>2</td>
<td>Final, cup TB</td>
<td>cup empty on tray</td>
<td>阮 cup on tray WS</td>
<td>Move cup tray→WS</td>
</tr>
<tr>
<td>3</td>
<td>Final, cup TB</td>
<td>cup empty on WS, box closed</td>
<td>RI box contains TB, Af box closed</td>
<td>Alter box to open</td>
</tr>
<tr>
<td>4</td>
<td>Final, cup TB</td>
<td>cup empty on WS, box open</td>
<td>Af TB in box cup, Move TB box→cup</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Final</td>
<td>box open</td>
<td>Af box open, Alter box to closed</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Final</td>
<td>kettle on stand, cup TB</td>
<td>阮 kettle on stand, RI step</td>
<td>No behavior</td>
</tr>
<tr>
<td>7</td>
<td>Final, cup TB+water</td>
<td>kettle on stand empty</td>
<td>阮 kettle empty off, Af kettle on stand</td>
<td>Alter kettle to filled</td>
</tr>
<tr>
<td>8</td>
<td>Final, cup TB+water</td>
<td>kettle on stand filled+cold off</td>
<td>阮 kettle filled off, Af kettle on stand</td>
<td>Alter kettle to on</td>
</tr>
<tr>
<td>9</td>
<td>Final, cup TB+water</td>
<td>kettle on stand boiled off</td>
<td>阮 kettle boiled, Af kettle on stand</td>
<td>Move water kettle→cup</td>
</tr>
<tr>
<td>10</td>
<td>Final, cup TB+water</td>
<td>cup TB+water</td>
<td>RI no sugar, Af SC closed</td>
<td>Alter SC to open</td>
</tr>
<tr>
<td>11</td>
<td>Final, cup sugar</td>
<td>cup TB+water, SC closed</td>
<td>Af SC closed, Alter SC to open</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Final, cup sugar</td>
<td>cup TB+water, spoon on WS</td>
<td>Af spoon, Take spoon</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Final, cup sugar</td>
<td>cup TB+water, spoon IH, SC open</td>
<td>RI no sugar, Af SC open, Move sugar to cup</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Final</td>
<td>cup TB+water, spoon IH, SC open</td>
<td>RI has sugar, Af WS, Give spoon to WS</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Final</td>
<td>cup TB+water, SC open</td>
<td>RI has sugar, Af SC open, Alter SC to closed</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Final</td>
<td>cup TB+water</td>
<td>RI no milk, RI step</td>
<td>No Action</td>
</tr>
<tr>
<td>17</td>
<td>Final, cup milk</td>
<td>cup TB+water, fridge closed</td>
<td>RI milk in fridge, Af fridge closed</td>
<td>Alter fridge to open</td>
</tr>
<tr>
<td>18</td>
<td>Final, cup milk</td>
<td>cup TB+water, fridge open</td>
<td>RI no milk, Af MB, Take MB</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Final, cup milk</td>
<td>cup TB+water, fridge open, MB IH</td>
<td>Af fridge open, Alter fridge to closed</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Final, cup milk</td>
<td>cup TB+water, MB IH</td>
<td>RI no milk, Af MB IH, Move milk to cup</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Final</td>
<td>cup TB+water, fridge closed, MB IH</td>
<td>Af fridge closed, Alter fridge to open</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>Final</td>
<td>cup TB+water, fridge closed, MB IH</td>
<td>Af fridge open, Give MB to fringe</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>Final</td>
<td>cup TB+water, fridge open</td>
<td>RI has milk, Af fridge, Alter fridge to closed</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>Final</td>
<td>cup TB+water</td>
<td>RI TB in tea, RI step</td>
<td>No Action</td>
</tr>
<tr>
<td>25</td>
<td>Final, cup tea/no tb</td>
<td>cup TB+water, spoon on WS</td>
<td>Af spoon, Take spoon from WS</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>Final, cup tea/no tb</td>
<td>cup TB+water, spoon IH</td>
<td>Af spoon, Take teabag from cup</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>Final, cup tea/no tb</td>
<td>cup tea, spoon with TB IH, chrd closed</td>
<td>Af chrd closed, Alter chrd to open</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>Final, cup tea/no tb</td>
<td>cup tea, spoon with TB IH, chrd open, bin closed</td>
<td>Af bin closed, Alter bin to open</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>Final, cup tea/no tb</td>
<td>cup tea, spoon with TB IH, chrd open, bin open</td>
<td>Af bin open, Give teabag to bin</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>Final</td>
<td>cup tea, spoon IH, chrd open</td>
<td>Af chrd open, Alter chrd to closed</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>Final</td>
<td>cup tea, spoon IH</td>
<td>Af WS, Give spoon to WS</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: IU analysis for JF making tea. The first column gives the sub-goal name from Table 1.
IUs necessary to start the major sub tasks might be expected to result in inaction errors and so require prompting on occasions. IUs 14 and 15 require the user to recall that they have already put sugar in the cup and so might result in repeats. In contrast, IU23 only requires the user to recognize that there is milk in the cup and so one would not expect repetition errors there. They do however have to recall that the milk bottle is in the fridge (IU17) and that the tea box contains teabags (IU3).

The affordances that involve familiar objects such as the spoon or the cup are unlikely to give problems even for quite demented people and so would have high initial ability values. The kettle and the fridge could be less familiar and might result in intermediate initial ability values. Similarly, none of the objects other than the kettle, are unfamiliar or obviously confusable with other visible objects so we would not expect recognition problems and high initial ability values can be set. The recognition operation in the first IU in each group has a special status like "R1 step". This is the object in the environment that we expect to act as a visual prompt for the subgoal. For example, seeing the cup on the tray (IU1) is hypothesized to be the cue that starts the whole subgoal of putting a teabag in the empty cup.

4. Hardware specifications for activity recognition – the choice of suitable sensors

One of the challenges that will be faced by real-world deployments of prompting systems is that homes differ significantly in their physical configuration and composition. It is one of the strengths of POMDP models that sensors with widely disparate characteristics and reliability can be relatively easily substituted for one another. Thus, the system designer has considerable choice as to the sensor that best responds to the requirements arising from the IU analysis. For example, the kitchen-based task requires the monitoring of several container lids and doors to determine whether they are open or closed. A variety of sensors could measure this property, including mechanical switches, light sensors, or accelerometers; each of which can be readily accommodated within the POMDP framework. The ubiquitous sensing technician is responsible for this sensor selection. The technician's methodology is to apply their expert knowledge to choosing sensors suitable for the measurable states, behaviours and actions specified in the model. The most appropriate sensors tend to be sensitive only to the physical property being measured while minimally affecting that property. Care must be taken with the digital sampling of the measured quantity to ensure that the value quantization and update frequency are suitable for the required sensitivity and range.

The sensing framework must take account of the errors that are likely to arise as a result of both signal noise and measurements during transitions. Consequently, some sensors benefit from the use of temporal filtering, or the application of hysteresis to prevent instability at boundary values. All sensors have an intrinsic uncertainty and during the modeling phase each sensor is assigned a reliability function that is used to model the sensor’s state. This function is empirically established using the hit rate and false positive rate for the sensor. Furthermore, to minimize uncertainty in the sensor values, data fusion of multiple (homogeneous or heterogeneous) sensors can be performed. We use the terms 'virtual sensor' to denote such a sensor whose values are generated by modeling data from one or more physical sensors. A number of activities cannot be measured using a simple (single) sensor, for these we use activity recognition on the stream of data from physical sensors, and present the result as a virtual sensor reading. For example, the output from a wireless tri-axial accelerometer rigidly attached to the kettle is processed to determine whether a ‘pouring activity’ is occurring.
As pointed out in section 1, to make this exercise a realistic test of the method, sensors were chosen to be very different to the computer vision based sensors used in the COACH project. We used the Ambient Kitchen in Culture Lab at Newcastle University [27] to realize a distributed sensor architecture that allows multiple nodes to be used to collect and process sensor data. A web-service provides a single point of contact for clients to access sensor readings. A number of sensor technologies were employed in this kitchen setting; Table 4 lists the full set of sensors.

As our environment has controlled lighting, we found that light sensors proved to be a highly reliable and unobtrusive method for determining whether doors and lids are open. High-frequency RFID readers embedded in the work surfaces, with temporal filtering to remove transient read failures, worked well for coarsely locating tagged objects. Small, low-frequency RFID readers proved reliable for local, single-tag presence detection. A current sensor was used to monitor kettle usage, a temperature sensor to monitor the water in the kettle, and an aggregation of force sensors on the base of the kettle was used to determine presence and approximate mass. Finally, wireless accelerometers and a “pouring” virtual sensor were used for both the kettle and milk bottle. The “pouring sensor” was based on a sliding window procedure, which extracted frames of contiguous accelerometer readings (x,y,z). For each of these frames statistical features were extracted (for details see [34]) and simple yet robust and efficient template matching using dynamic programming was used for activity recognition.

Sensor reliability differed widely between the different physical and virtual sensors used. These were characterized by two numbers: a hit rate, and a false positive rate. The hit rate gives the fraction of time the sensor correctly detects the appropriate value of the state variable it is associated with. The false positive rate gives the fraction of null events that are incorrectly detected by the sensor. For example, the high frequency RFID tag in the work surface correctly detects the cup on the work surface only 62% of the time when it is on the work surface (a hit rate of 0.62), but only detects the cup on the work surface 0.1% of the time when it is not there (a false positive rate of 0.001).

5. The automatic generation of a SNAP prompting system

Once the IU analysis is complete, and the sensors have been defined and measured, a synthesizer converts this into a complete POMDP specification for each IU group. A controller then sequences the groups, as described in Section 5.4. As the IU analysis is qualitative, but the POMDP model is quantitative, some additional parameters must be specified. We have built a simple software interface that allows the human factors annotator to enter this additional information. In this section, we give an overview of how the POMDP models are automatically generated based on the IU analysis, the sensor definitions, and this additional information. We use the first IU subgroup in tea-making as a demonstrative example here (Table 3, IUs 1–5). The goal of this group is to get the cup onto the work surface with a teabag in it.

5.1. Information from IU analysis

The POMDP state space is compiled directly from the IU analysis in the following way. The task variables are given by the current environment and current goal factors in Tables 1 and 2. For the first IU group, this consists of the cup’s position (on the tray or on the work surface), whether the teabag box is open or closed, and whether the teacup is empty or has a teabag in it. The behavior factors are given by the final column in Table 3. In the first IU group, the behaviors are moving the cup from tray to work surface, opening/closing the box, and moving the teabag from
the box to the cup. In addition, two special behaviors are always present: doing nothing, and doing something off-task (other). Finally, the ability factors are given by the $Rn/Rl/Af$ column in Table 3. There is one variable for each recognition, recall or affordance that is related to each IU in this IU group, with values 'yes' and 'no'. Thus, a recall variable with a value of 'yes' means that the user has no trouble for this particular recall.

There is one system action for each ability factor. The actions are a prompt or signal that will help the user with this particular cognitive ability, if missing. For example, to help with recognition of the cup, a light could be shone on it, or an audio prompt could be delivered. In our current system, all system actions are audio prompts relating to the ability. At this stage in our work, these prompts were generated manually as placeholders for testing, but in the future, they will need to be specified by end-users (e.g. by recording them themselves).

The ‘start’ state for task variables is given in the IU analysis explicitly in Table 2. For example, the cup starts on the tray, the box starts closed, etc. The initial state for cognitive abilities is specified manually as described in Section 3.5, and requires an analysis of a particular person’s abilities. These can be obtained from a generic model of dementia, and based on a current diagnosis for a particular person.

POMDP task dynamics are given by a deterministic function that relates the behaviors to the task states, and gives the evolution of the task states over time. This is a specification of the first factor in Equation (4), $Pr(T'|B', T, A)$, and is given by each subsequent set of task states, $t, t'$, in the IU analysis (Table 3), and the associated behavior, $b'$. In the following, we use capital letters to denote random variables, and small letters to denote particular values for those variables. The effect on a particular task variable, $T$, is given by

$$Pr(T'|T, B', A) = \delta(T' = t)\delta(B' \neq b') + \delta(T' = t')\delta(B' = b'),$$

(5)

where $\delta(x) = 1$ if $x = 0/$false and $\delta(x) = 0$ otherwise. This function indicates that, for any row in Table 3, when the behavior $B'$ is equal to the behavior in the rightmost column, $b'$, the task state, $T'$ becomes the task state in the subsequent row, $t'$. Otherwise, when the behavior $B'$ is anything else ($B' \neq b'$), the task state remains the same, $T' = t$. Each system action only has an effect on the associated user ability, and so no additional information is required to specify the dynamics other than the ‘generic’ dynamics of an ideal trace through the task. The basic idea is that the POMDP dynamics models this ideal trace as an effect of user behavior on task variables. Deviations from this ideal trace are induced by a lack of cognitive ability, which can be influenced by system actions (see Section 5.3).

5.2. Information from sensors

One observation variable is defined for each sensor with values as the readouts of the sensor. Observation variables are assigned to one of the sets $K$ or $V$ depending on whether they relate to a task or behavior variable. For example, the RFID tag on the tray is in set $K$ as it will detect (yes, no) if the cup is on the tray (a task variable). The other sensors in the first IU group are all in the set $K$ and include the RFID tag on the work surface (yes = cup on work surface), a sensor on the box lid (yes=open), and an RFID tag on the teabag ticket (yes=on work surface, which

---

1In general, as shown in Figure 1(b), the task state can be influenced by the system action (e.g. if the system has some actuator like a robotic arm that can physically change the environment [8], or if the domain includes a computer interface that can be modified by the system [7]), but we assume here that $Pr(T'|T, B', A) = Pr(T'|T, B')$, as this assumption will hold for our kitchen experiments, and for any systems that only give audio cues.
indicates that the teabag is in the tea). In the second IU, a virtual sensor detects the motion of pouring water from the kettle, and so is related to the user behavior and will be in set \( V \). The POMDP observation functions, \( \Pr(V|B) \) and \( \Pr(K|T) \), are then given by the reliability measures of the sensors as described in Section 4.

5.3. Additional information

In order to complete the quantitative POMDP model, four additional pieces need to be specified by the human factors annotator or by end users. These are as follows.

**Cognitive ability constants (CASs).** One constant, \( c(y, b) \), for each cognitive ability, \( Y = y \), and behavior, \( B = b \), indicates how much this ability is implicated if the user does not perform \( b \) in states where \( b \) is relevant. For example, if the person has the sugar container open, but the spoon is on the work surface, we normally expect them to pick up the spoon. There are four cognitive abilities at play in this situation: recalling the step, recalling that the tea has no sugar in it, and seeing the a box is relevant in a state where the box is closed). The cognitive ability relevance functions are defined negatively, so typically \( c(y, b, t) = \) large for \( b = \) do nothing and small or zero for other behaviors: if a person lacks ability \( y \), they are more likely to do nothing. These effects are used to define the dynamics of behavior as

\[
\Pr(b|y', b, t, A) \propto \sum_{b \in B} f_b(t) \delta(b = b_t) + \sum_{y' \in Y} (1 - \delta(y')) c(y, b', t),
\]

where \( \delta(y') \) is shorthand for \( \delta(y' = 'yes') \), and \( 1 - \delta(y') \) therefore indicates \( \delta(y' = 'no') \). This definition shows that our expectations about a person’s behaviors are that they will do the relevant behavior (the first term), but this is modified by the effect of a person not having a cognitive ability (given by \( 1 - \delta(y') \)) in the second term. If the person lacks a relevant ability \( y \), then the probability of \( b \) not being appropriate is increased.

\(^2\)As for the task dynamics, the system action actually has no effect here, so that \( \Pr(B'|Y', B, T, A) = \Pr(B'|Y', B, T) \); the system’s actions cannot directly change a persons behaviors (e.g. by physically moving the person in a domain with a powered wheelchair [24]).
Responsiveness constants. These constants specify how each prompt (system action) will impact on the user’s cognitive abilities. Responsiveness constant \( \beta(y, a) \) gives the probability that the cognitive ability \( y \) will change to 'yes' if it is 'no' if the system takes action \( a \). For example, if an audio message reminds the user about the cup, how likely is the user to recognize the cup if they can’t already. A separate set of constants, \( \beta'(y, a) \), gives how long this effect lasts.

The responsiveness constants are used to build the third factor in Equation (4), \( Pr(Y'|Y, A) \), as follows

\[
Pr(y'|y, a) \propto \delta(y) \left[ \delta(y')(1 - \beta'(y, a)) + (1 - \delta(y'))\beta'(y, a) \right] + (1 - \delta(y)) \left[ \delta(y')\beta(y, a) + (1 - \delta(y'))(1 - \beta(y, a)) \right].
\] (8)

The second term shows that \( y \) will increase by a factor \( \beta \) if it is 'no', and the first term shows that \( y \) will decrease by a factor \( \beta' \) if it is 'yes'.

Rewards / Costs. All rewards and costs are specified up to an overall linear scale factor, such that they can always be mapped to lie in \([0, 1] \) [35]. In our case, we use some reasonable values (in \([0, 15]\)) so that rewards can be specified as integer values (to avoid the user having to input fractions or decimal numbers). The final (goal) state of each IU group (see Table 2) is assigned a reward of 15, and the initial state a reward of 0. Intermediate states with some value (e.g. getting the tea box open) are assigned values in the range 0-15 according to how much progress has been made towards the goal in that state. Each system action also carries a cost. The actions are costly because the system should let a user do it themselves if they can. Thus, in any situation where the user is expected to complete a step by themselves, the 'do nothing' action will be preferred. Monetary costs can also be included in these (e.g. the cost of power, human resources, etc). In our example, we ignore the monetary cost and consider only user-impact costs. In general, the more complex prompts, such as recalling an entire step are less costly than the more specific ones. The costs are assigned in the range 0-7, with more costly prompts being ones that are less desirable. This is user-dependent, and in our experiments we model affordance prompts to be most desirable of all prompts with a cost of 0.5, recognition prompts to be second with a cost of 1.0, and the recall prompts to be least desirable, with a cost of 7. This choice follows by viewing the recall prompts as being most invasive or potentially offensive to a user, while the more specific affordance prompts are less likely to be offensive. However, the inverse mapping could also be used, depending on user preferences.

Timeout constants. Each state, \( s \), of each subgoal POMDP will have a timeout constant associated with it, \( \tau(s) \), that indicates the length of time one would normally expect a person to take to complete the relevant behavior at that state. The actual timeout at any given time, \( \Delta t \), is computed from the belief state and the timeout constants as \( \Delta t = \sum_s b(s)\tau(s) \), an average over all timeout constants weighted by the current belief state.

5.4. Hierarchical control

The IU analysis breaks an ADL like making a cup of tea down into a number of sub-tasks, or sub-goals. For tea making, there are five sub-goals, as shown in Tables 1 and 3. This decomposition arises naturally according to the major elements of recall noted in the videos from which this IU analysis was made. The five sub-goals are partially ordered, e.g. sub-goal 2 could be done before sub-goal 1, and 3 and 4 can be interchanged. However, sub-goal 5 must come after sub-goal 2. The partial ordering can be specified as a list of pre-requisites for
Each sub-goal giving those sub-goals that must be completed prior to the sub-goal in question. Since each sub-goal is implemented as a separate POMDP controller, a mechanism is required to provide hi-level control to switch between sub-goals. The controller we use is very simple, maintaining a current control index, and passing all observations made to the control sub-goal POMDP. The control is switched when either the control POMDP has reached its goal (to within some threshold), or a new sensor measurement (change) is made that does not correspond to the current control sub-goal. The newly selected control is either the sub-goal that this new sensor measurement is associated with if all pre-requisite sub-goals are complete, or the first pre-requisite sub-goal if not. This allows a user to switch sub-goals during execution, but only to those that respect the partial ordering referred to above.

Each sub-goal controller, guided by the central controller, essentially loops through the following steps:

1. query policy based on current belief state, \( b \), to get next action to take, \( a \)
2. actually perform the action \( a \)
3. query sensors for changed observations, \( o \), or a timeout, whichever comes first
4. update the belief state based on \( b \), \( a \) and \( o \).

However, since steps 4 and 1 are computationally intensive, taking up to 3 or 4 seconds each for larger models, it can be the case that, at step 2, the observations, \( o \), have changed since they were last sampled (at step 3) and used to compute the belief state on which the choice of action \( a \) was based. If this occurs, then the action should not be performed at step 2, but rather replaced with a default “do nothing” action. This slight modification addresses issues where the person does something while the subgoal controller is processing. While a solution to this problem is to speed up steps 1 and 4, there may still be fast behaviors that are missed, and a last-minute check is usually advisable.

5.5. Implementation issues

The central controller described in the last section and the POMDP controllers for each sub-goal are implemented in Java, and run as separate processes on three PCs with 2 GHz processors and 2GB of RAM. The sensors are sampled at 1 second intervals by an observer process. The central controller polls the observer (at step 3 in the algorithm above), and receives the most recent observations. Note that this polling arrangement may miss brief events. For example, if a person opens and then closes a box during the time when the controller is processing at step 1, the most recent sensor values will be read, and no change will be registered. We will see how this “sensor memory” issue affects our results in the next section.

There are a number of solutions for the sensor memory problem. The simplest is to adjust the sensor reliabilities to reflect this. The POMDP controllers will then be able to adjust their policies to take into account the additional uncertainty related to this. However, this solution is not ideal as it will lead to less “confident” policies, i.e., policies that are based on belief states with more uncertainty, and only due to a lack of a proper temporal model for the sensor readings. The second solution is to include some additional “sticky” virtual sensors that indicate an event happening in the past (e.g. the box has been opened and then closed again). Our experiments in the following section avoided this heuristic solution in order to clearly demonstrate the abilities of the POMDP model to deal even with incorrectly specified sensor reliabilities.
5.6. The feasibility of developing the method into a practical service for people with dementia

The purpose of this paper is to demonstrate: (i) that it is possible to automatically generate a working POMDP model from a description of a task and the kitchen in which it is to be carried out and a set of sensor specifications, and (ii) that the method could feasibly be developed into a practical service for people with dementia. Our success in terms of the first question, "did we generate a prompting system that works?" is addressed in the next section. Here we will address the second question of feasibility in more practical circumstances.

There are 3 basic parts to the specification process: generation of the IU analysis, hardware specification and entry of the model parameters for automatic generation. IU analysis is a relatively skilled job requiring a good understanding of the user and the user's circumstances. As indicated in Section 3, this is a skill set that might be expected of an occupational therapist or a human factors analyst. Making tea is a relatively simple kitchen task and the IU analysis took about half a day to complete. One can imagine tool support in the form of libraries of IUs for common subtasks that the analyst can adapt to a particular user and user's kitchen. One could also imagine tool support for writing IUs to check for consistency of labeling and so on. Together this could reduce the effort required considerably.

In this exercise hardware specification included a good deal of sensor design, i.e., building, testing and then rebuilding. In a realistic clinical or commercial setting one would expect the set of sensors the technician is choosing from to be well understood and with well established reliabilities. A home visit, or access to videos, would be required at the specification stage and some testing of the eventual POMDP prompting system to check sensor reliability in the final implementation, but overall the effort required for this stage would not appear to be too onerous.

The final step, entering parameters generated from the IU analysis, sensor specification could be to large extent automated and driven from the tools provided for IU and sensor specification. However, in addition, there are Cognitive Ability, Responsiveness and Timeout constants also the rewards/costs parameter to be added. In the current implementation this was done by the first author drawing on his wide experience with POMDP prompting systems. In future systems we imagine this process could also be largely automated, with a small amount of input from the human factors analyst with regard to user abilities.

In conclusion, while we have not demonstrated a method that would be practical in a real clinical or commercial context in this case study, we can see how it could be made practical and are encouraged to go on to try and do so.

6. Demonstrative Examples: Did it work?

The previous sections have detailed the proposed method for automatically generating prompting systems and illustrated it by applying it to user JF making tea. The question then arises as to whether the resulting system does in fact work, that is, does it prompts at appropriate moments. To test this we asked two volunteers who had no knowledge of the system to work through two scenarios derived from the original scenario of JF making tea. In the first, All-Unresponsive, the participant was instructed to wait for prompts before doing anything, but to respond to all instructional prompt, i.e., to follow exactly the commands given by the system. In the second scenario, Subgoal-Unresponsive, the participant was instructed to hesitate after each subgoal has been completed, and to wait for a prompt before continuing. The subgoals are implicitly presented to the participant by prompts, which are less detailed and describe the next status to be reached (“Place a cup, with a teabag in it, on the work surface.”) rather than giving detailed
instructions (“Get the milk from the fridge”). In both scenarios, participants were asked to ignore inappropriate prompts although these were recorded for our analysis. The two scenarios are motivated by our prior observations of people with dementia engaged in similar activities [26], although we must emphasize that they are only intended as scenarios capable of shedding light on SNAP’s performance, rather than test cases for usability by a person with dementia.

Our general criteria for success in this test of the system were that in the All-Unresponsive scenario the system would come up with the appropriate prompt for each action at the right time, while in the Subgoal-unresponsive scenario the system would only prompt at the junction between subgoals. In general, these criteria were met (see Tables 6 and 7). Where they were not we were able to infer some possible refinements that could be made to the process. These detailed results are described below.

Note that our choice of volunteer actors to work through these scenarios was simply to avoid the possibility that a technically savvy actor could get better performance out of the system than was realistic. We are not suggesting that these actors are in any way representative of people with dementia. Nor are we suggesting that the single task chosen and the environment used (the Ambient kitchen) represent the true range of kitchen environments and activities people with dementia might present with. Note also that for the purposes of this exercise little effort was put into prompt design. Prompts were generated using the Microsoft SAPI text-to-speech engine and were written to reflect the intent of the prompt. Whilst we acknowledge that the design of appropriate prompts (in terms of both modality and content) is crucial to the development of an effective practical system for supporting people with dementia, in this case our prompts were chosen only to be recognizable by our participants.

6.1. Reliability of Recognition

Table 4 gives an overview of the sensors (both actual and virtual) used for tea preparation monitoring, and their reliabilities as tested in isolated reproducible experiments. Note that reliability values for observations are required for POMDP parametrization, which is reflected in the organization of the table. To obtain these values, twelve participants were asked to perform relevant activities (e.g. ”Place a cup on the work surface”, ”Get a teabag from the tea box”, ”Fill kettle with water and put it on the stand”). No further instructions or constraints were given and the capabilities of the particular recognition sub-systems (e.g. cup detector, teabag detector, kettle recognizer) were tested by counting the number of false and correct detections. A wide range of reliability values was observed across sensors, from almost perfect recognition results (e.g. detection of kettle being switched on) to relatively poor performance (e.g. teabag detection). Whereas light sensors and accelerometers worked well, object detection in our cluttered environment kitchen environment often involved fast moving objects and limitations in the set-up of our high-frequency RFID sensors, which are embedded into the work surface resulted in significant error rates (due to polling times and the limited spatial scope of the readers). Note that the second set of RFID readers (small, low-frequency) proved more reliable. However, these readers do not allow sensing of multiple tagged objects that are located spatially close to each other and are prone to interference reasoned malfunctions when utilizing multiple readers. Unfortunately, these practical limitations prevent their broader use.

6.2. Analysis of the Prompting System

We tested SNAP for the two archetypal scenarios as described above, in which two volunteers were asked to prepare a cup of tea – All-Unresponsive and Subgoal-Responsive. Each participant
<table>
<thead>
<tr>
<th>Sensor</th>
<th>Description</th>
<th>Detection</th>
<th>States</th>
<th>FPR [%]</th>
<th>FNR [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>floor_sensor</td>
<td>Pressure sensitive floor</td>
<td>person present in kitchen?</td>
<td>on, off</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>cup_on_tray</td>
<td>RFID tag on mug, tray reader</td>
<td>mug on tray?</td>
<td>yes, no</td>
<td>0.1</td>
<td>38</td>
</tr>
<tr>
<td>cup_on_ws</td>
<td>RFID tag on mug, WS reader</td>
<td>mug on work surface?</td>
<td>yes, no</td>
<td>0.1</td>
<td>38</td>
</tr>
<tr>
<td>teabox_lid</td>
<td>Light sensor in tea box</td>
<td>tea box open?</td>
<td>open,</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>teabag_in_cup</td>
<td>RFID tag on teabag, WS reader</td>
<td>teabag in mug?</td>
<td>yes, no</td>
<td>0.1</td>
<td>75</td>
</tr>
<tr>
<td>kettle_switch</td>
<td>Current sensor in kettle</td>
<td>kettle on/off?</td>
<td>yes, no</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>water_level</td>
<td>Force sensors in kettle stand</td>
<td>kettle filled?</td>
<td>full, empty</td>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>kettle_boiled</td>
<td>Temperature sensor in kettle</td>
<td>kettle boiled?</td>
<td>yes, no</td>
<td>10</td>
<td>22</td>
</tr>
<tr>
<td>kettle_pour</td>
<td>Accelerometer (pouring virtual sensor)</td>
<td>kettle poured?</td>
<td>yes, no</td>
<td>26.1</td>
<td>0.1</td>
</tr>
<tr>
<td>spoon</td>
<td>RFID tag on spoon, WS reader (drawer)</td>
<td>spoon in drawer?</td>
<td>onws, inhand</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>sugarbox_lid</td>
<td>Light sensor in sugar box</td>
<td>sugar box open?</td>
<td>open,</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>fridge_door</td>
<td>Light sensor in fridge</td>
<td>fridge door open?</td>
<td>open,</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>milk_in_fridge</td>
<td>RFID tag on milk bottle, reader in fridge</td>
<td>milk bottle in fridge?</td>
<td>yes, no</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>milk_pour</td>
<td>Accelerometer (pouring virtual sensor)</td>
<td>milk poured?</td>
<td>yes, no</td>
<td>28.8</td>
<td>0.1</td>
</tr>
<tr>
<td>cupbud_door</td>
<td>Light sensor in cupboard</td>
<td>cupboard open?</td>
<td>open,</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>bin_lid</td>
<td>Light sensor in bin</td>
<td>bin lid open?</td>
<td>open,</td>
<td>0.1</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 4: Reliabilities for recognition sub-system of SNAP (FPR: false positive rate; FNR: false negative rate; rates of 0.1% represent perfect performance during evaluation, flooring applied to avoid zero probabilities)

ran through one scenario once. The test system’s processing requirements led to non-trivial delays in order to query the POMDP policies and update the POMDP belief states. To get an idea of the scale of these times for our particular implementation, Table 5 shows the processing times broken down by sub-goal. For each sub-goal, the number of states, actions and observation in the controller POMDP are provided. Although the processing times depend on the size of the POMDP (in states, actions and observations), this dependence is not always linear, due to data structure overheads in our code. It is evident, however, that both policy querying and belief updates have significant performance costs. Nevertheless, we will see in the next sections that the SNAP is able to handle these delays.

6.2.1. All-unresponsive

Table 6 shows the transcript of the tea preparation session for the All-Unresponsive scenario. This includes the active subgoal, the action performed by the system, the prompt actually delivered and the appropriateness of the given prompt. Snapshots of the system state for the scenario are shown in Figures 3 and 4. Each snapshot includes a still of the video footage, observations (i.e. sensor readings at the particular step), and the beliefs of the system at the time the system prompted. For the system beliefs, prefixes are used to associate beliefs to either abilities, i.e.
Table 5: Processing characteristics of the scenarios and POMDP.

<table>
<thead>
<tr>
<th>Subgoal</th>
<th>No. states</th>
<th>No. actions</th>
<th>No. observations</th>
<th>Average Belief update time (ms)</th>
<th>Average Policy query time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teabagcupws</td>
<td>12288</td>
<td>8</td>
<td>16</td>
<td>1705</td>
<td>3094</td>
</tr>
<tr>
<td>Kettleboilcup</td>
<td>5120</td>
<td>6</td>
<td>16</td>
<td>723</td>
<td>668</td>
</tr>
<tr>
<td>Sugarcup</td>
<td>7168</td>
<td>7</td>
<td>4</td>
<td>434</td>
<td>4</td>
</tr>
<tr>
<td>Milkcup</td>
<td>28672</td>
<td>9</td>
<td>8</td>
<td>1884</td>
<td>18</td>
</tr>
<tr>
<td>Tbtobin</td>
<td>16384</td>
<td>7</td>
<td>16</td>
<td>996</td>
<td>8</td>
</tr>
</tbody>
</table>

recognition (rn), recall (rl), and affordance (af); behaviors (b); task (no prefix). Eleven snapshots have been selected that correspond to significant events in the sequence.

SNAP successfully guided the participant through the tea preparation process. In total it provided 21 prompts of which 16 were appropriate, i.e., they were given at the correct time within the tea preparation process. Otherwise, the system appeared to “do nothing” at the appropriate times. The five inappropriate prompts were either: (1) appropriately timed but inappropriate for the situation (i.e., a prompt was needed but the system gave an incorrect one), or (2) inappropriately timed (i.e. not needed or an incorrect prompt). We will refer to the first type of error as misPrompts and the second type as false positive prompts. The errors in Table 6 were due to one of three possible causes: sensor errors are sensors that are not behaving according to the reliability measurements in Table 4; sensor memory errors arise from timing issues as discussed in Section 5.5, and model errors are due to incorrect specification of the controllers, i.e., transcription errors between the IU analysis and the POMDP specifications. By analyzing the snapshots, and these errors in particular, we can gain an insight into the SNAP’s behavior and overall capability.

Figure 3(a) shows the initial state, with the cup empty and on the tray, and the teabox closed. The abilities optimistically estimated to be very high (close to 1), to give the person a chance if she is capable of doing the task on her own. The first prompt, in Figure 3(b) cues the person to move the cup to the work surface, which she does, as indicated by the non-zero belief in the behavior b[mv_cup_ws] in Figure 3(c). The first false positive prompt, shown in Figure 3(d), asked for the tea box, although the tea bag had already been taken and put into the cup. The source of the false prompt was a sensor memory error (as explained in Section 5.5) that resulted from the participant rapidly opening and then closing the box without a prompt: the sensor event was missed by the SNAP controllers. When this box-lid open/close event is missed, the POMDP assigns a very low probability to this event having actually happened (it was assigned the minimum miss rate of 0.1% since it was never observed to occur in our sensor reliability study). Since SNAP also observed the teabag in the cup, which has a much lower reliability (75% miss rate), it concluded that, with high probability, the teabag is actually not in the cup, and continues to prompt the person to recognize the tea box. If the reliability of the tea box sensor had accounted for this sensor memory problem (i.e. had a lower reliability), confidence in the teabag_in_cup sensor would have been greater, and prompt rn box prompt less likely to occur. Although erroneously prompting, the subgoal was successfully completed (shown in Figure 3(e), where the goal is reached), since subsequent sensor readings lowered the system’s propensity to prompt for recognition of the box. This elegantly demonstrates the capabilities of POMDPs to cope with uncertain and noisy data and incorrect sensor reliability measurements, and to recover from potential dead ends.

The second false positive prompt, prompt rn kettle empty shown in Figure 4(a), was to fill the kettle although the participant was in the process of getting water from the tap. The
Figure 3: Experiment for All-Unresponsive case (selected scenes for overview): snapshots with observations and beliefs. Each snapshot includes a still of the video footage, observations (i.e. sensor readings at the particular step with dark color and a dot indicating the sensor is ‘on’), and the beliefs of the system at the time the system prompted. For the system beliefs, prefixes are used to associate beliefs to either abilities, i.e. recognition (rn), recall (rl), and affordance (af); behaviors (b); task (no prefix).
Figure 4: Experiment for All-Unresponsive case (continued from Figure 3)
<table>
<thead>
<tr>
<th>Subgoal</th>
<th>Action</th>
<th>Prompt</th>
<th>Correct</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>prompt_rn_cup_on_tray</td>
<td>The cup?</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>prompt.rl_step</td>
<td>Place a cup, with a teabag in it, on the work surface.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>prompt_rn_box</td>
<td>The tea box?</td>
<td>×</td>
<td>Sensor memory error</td>
</tr>
<tr>
<td>2</td>
<td>prompt_off_kettle_on_stand</td>
<td>The kettle?</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>prompt.rl_step</td>
<td>Boil water in the kettle and pour it into the cup.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>prompt_rn_kettle_empty</td>
<td>The kettle is empty... fill the kettle.</td>
<td>×</td>
<td>Sensor error: no sensor reading while filling the kettle</td>
</tr>
<tr>
<td>2</td>
<td>prompt_off_kettle_on_stand</td>
<td>The kettle?</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>prompt_rn_kettle_filled_off</td>
<td>The kettle is off... switch the kettle on.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>prompt.rl_kettle_boiled</td>
<td>The kettle has boiled... pour water into the cup.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>prompt.rl_no_sugar</td>
<td>Sugar?</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>prompt.rl_step</td>
<td>Add some sugar to your cup of tea.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>prompt.of_sc_open</td>
<td>Close the sugar container.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>prompt.of_mb_ly</td>
<td>Get some milk out of the bottle.</td>
<td>×</td>
<td>Model error: incorrect prompt milk</td>
</tr>
<tr>
<td>4</td>
<td>prompt.of_fridge_closed</td>
<td>The fridge?</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>prompt.of_mb_ly</td>
<td>Get some milk out of the bottle.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>prompt.rl_milk_fridge</td>
<td>Get the milk from the fridge.</td>
<td>×</td>
<td>Sensor memory error: milk already poured</td>
</tr>
<tr>
<td>5</td>
<td>prompt.of_spoon</td>
<td>Use the spoon for catching the tea bag from the mug.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>prompt_rn_mb_tea</td>
<td>Get the tea bag out of the cup.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>prompt.of_cbd_closed</td>
<td>The bin is in the cupboard.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>prompt.of_spoon</td>
<td>Use the spoon to remove the tea bag from the mug.</td>
<td>×</td>
<td>Sensor error: tea bag already in bin</td>
</tr>
<tr>
<td>5</td>
<td>prompt.of_ws</td>
<td>Put the spoon back.</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Transcript of All-Unresponsive scenario (16 appropriate prompts, 5 inappropriate prompts).

Prompt was issued because SNAP estimated the person’s ability to rn_kettle_empty had dropped too low. The water level of the kettle is measured implicitly by evaluating force sensors in the stand. However, the system did not recognize the fact that the kettle was being filled, assumed that no appropriate actions were being undertaken, timed out, and then prompted. Again, the system recovered when it recognized the kettle as being full (by weight on the stand), and the kettle being switched on (Figure 4(b)), even though a faulty sensor reading of kettle_hot added a confusing factor. This points to the necessity of including as many sensors as possible that are related to the task. In this case, any one of a number of additional sensors would have sufficed, for example, recognizing the presence/absence of the kettle on the stand, a water flow sensor in the tap, or using the existing accelerometer in the kettle.

The third subgoal (adding sugar) was processed without any erroneous prompting. However, two errors were made during the fourth subgoal (adding milk). Firstly, the system issued the misprompt prompt.of_mb_ly (attempting to get the person to pour the milk) at the beginning of the subgoal (Figure 4(c)), due to a model error. The second error was the prompt to get the milk
from the fridge (prompt rl_milk_fridge). This was a false positive prompt due to a sensor memory error (as in the first subgoal), although the pouring milk behavior was detected correctly by the sensor, it did not last long enough to be passed to the controller (Figure 4(d)). Nevertheless, the POMDP correctly inferred that the behavior of pouring the milk was going on from the context. The prompt was also misprompt in that the milk was already out of the fridge, which was the result of a model error in the transcription of the cognitive ability relevance functions from the IU analysis to POMDP model. In this case, the cognitive ability of rl_milk_fridge was incorrectly encoded as being relevant for the situation in which the cup had milk, the fridge was closed, and the milk bottle was in the user’s hand - which was not apparent in the IU analysis (see Table 3). Eventually the system moved on to the final subgoal.

The final subgoal controller had a faulty bin-lid sensor, which caused significant problems in both this and the second demonstration. The end result was that the SNAP controller never saw the bin-lid being opened, and assumed the spoon still contained the teabag. This resulted in the final false positive prompt in Table 6, and also in the subgoal controller not terminating properly. However, as shown in Figure 4(e) and (f), appropriate prompts to open the bin and put back the spoon were given, but the belief that the teabag had left the cup was not strong enough to overcome the evidence from the faulty bin-lid sensor. Once the participant had left the kitchen, the controller continued to prompt the person to pick up the spoon, since it did not think the entire task had completed. We do not include these additional prompts in Table 6, as the person had already left the kitchen. This also points to the necessity of embedding these controllers in a larger context, so that e.g. the person leaving the kitchen would also be modeled.

<table>
<thead>
<tr>
<th>Subgoal</th>
<th>Action</th>
<th>Prompt</th>
<th>approp?</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>prompt rl_milk_fridge</td>
<td>The cup?</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>prompt rl_step</td>
<td>Place a cup, with a teabag in it, on the work surface.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>prompt rl_no_sugar</td>
<td>Sugar?</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>prompt af_milk_mb</td>
<td>Get some milk out of the bottle.</td>
<td>× Sensor error: already done</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>prompt af_spoon</td>
<td>Use the spoon for catching the tea bag from the mug</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>prompt rl_tb_tea</td>
<td>Get the tea bag out of the cup.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>prompt rl_step</td>
<td>Move the tea bag into the bin.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>prompt af_cbrd_closed</td>
<td>The bin is in the cupboard.</td>
<td>× Sensor error: tea bag already in bin</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>prompt af_us</td>
<td>Put the spoon back.</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

Table 7: Transcript of Subgoal-Responsive experiment (8 appropriate prompts, 3 inappropriate).

6.2.2. Subgoal-Responsive

Table 7 and Figure 5 illustrate the progress of the Subgoal-Responsive experiment and the system behavior in the same way as for the All-Unresponsive case. The system prompted 11 times in total (eight appropriate / three inappropriate). Figure 5(a) shows an early prompt to
move the cup, to which the person responds by moving the cup then opening the box to get out the tea bag (Figure 5(b) where the behavior of opening the box, b_open_box, is clearly recognized). The second subgoal (boil water) was completed with only a single prompt. In the first error case (third subgoal, add sugar), SNAP prompted for the spoon to be taken for getting some sugar although sugar has already been added to the cup of tea. This was a false positive prompt, and was due to a failure of the spoon sensor (it was never detected the spoon being removed from the drawer as shown in Figure 5(c)). The second error was a misprompt to get milk from the bottle that happened as in the All-Unresponsive case (see Figure 4(c)). The system helped the person through the fourth sub-goal, and then started the last sub-goal off by reminding the person of the step (Figure 5(d)). The person then opened the drawer to remove the spoon, an activity (b_take_spoon) that was correctly recognized by the SNAP (Figure 5(e)). However, again the faulty bin-lid sensor caused a false positive prompt to open the bin. Similar to the all-unresponsive case, the system recovered and moved on processing new sensor readings. This
again underlines the flexibility of POMDP models to cope with dynamic, i.e., rapidly changing real-world scenarios. The same non-termination condition also occurred in this case, and is due to the same problem as mentioned earlier.

7. Conclusions and discussion

This paper has demonstrated a method for generating a working partially observable Markov decision process (POMDP) prompting system from a psychologically justified description of the task and the particular environment in which it is to be carried out (IU analysis) together with a specification of the available sensors and effectors. The case is made that this method could feasibly be developed into a practical service for people with dementia, where the task description is developed from one or two videos of the patient carrying out the task and the sensor and effector specification from a library of equipment data.

In the process of developing this method a working system was constructed in the Ambient Kitchen at Newcastle University. This was demonstrated by having volunteers work through certain scenarios of use. The POMDP successfully modeled uncertainty in sensor measurements and in the dynamics of interaction. The resulting controller was able to deal flexibly and gracefully with errors in sensor readings and with unpredicted user interactions. However, this exercise also pointed up a number of outstanding issues still to be addressed.

There were three sources of errors in our initial prototype: sensors that are not responding according to the reliability measurements; sensor memory errors arising from timing issues (Section 5.5), and model errors due to transcription errors between the IU analysis and the POMDP specifications. We believe that these three types of errors are relatively easy to fix. Perhaps the most challenging errors are timing problem in the implementation. The sensors are operating on a fast update schedule (≈ 1Hz), whereas the POMDP controllers are updating much more slowly, in part due to the large size of the models and somewhat inefficient implementations. The mapping between these two timescales was achieved by having the controller simply poll the sensors whenever it needed new readings. This led to sensor memory errors where an event could take place, be registered by the sensors, but go unnoticed by the POMDP controllers, since they were busy doing other processing. Solutions to this problem include “sticky” virtual sensors that maintain some memory of their previous state, ensuring that the sensor reliabilities take timing into account, and making the POMDP operations more efficient.

A related timing issue is how to specify the timeouts for each state, \( \tau(s) \) (see Section 5.3). The focus of the current paper does not address this issue, however, the COACH handwashing system uses a timeout that is dynamically adjusted based on the current estimated level of dementia (difficulty they have washing their hands) of the person. If the person has more trouble, the timeout is lengthened. This could be integrated into future versions of the SNAP system.

One avenue of future work will be to address the error sources that we encountered by constructing a more precise and flexible temporal abstraction and event queuing layer, improving the sensors, and improving the IU translation interface. The specification of some remaining constants needs to be understood and included in the analysis interface. We then plan to test how the methodology can be adapted for user by end-users including persons with dementia and their families, caregivers, or other medical professionals. Our longer term goals are to construct hierarchical controllers automatically from IU analysis, and to combine multiple, interleaved and concurrent ADL prompting systems, an important challenge for future uptake of such systems by end users.
In the development of the working example we did not address the question of prompt design, the prompts were always verbal. Previous work in this group has stressed the use of multimodal cues [26]. For example, if the problem inferred by the system is that the patient has not recognized the object key to the next action, the most effective prompt may be to have that object flash or buzz, rather than naming it in a verbal prompt. Future work will address this opportunity by using a wider range of prompting methods and testing them with people with dementia. Another important area for future work is to address the problem of identifying who is in the kitchen when it may be the patient, a friend or a carer.

While there are many challenges to be addressed before this method can be used in a practical service for people with dementia, we believe that utilizing the efficiency of an IU analysis of task and environment to automatically generate flexible and error tolerant POMDP prompting models has much promise.

References

[20] K. J. Åström, Optimal control of Markov decision processes with incomplete state estimation, Journal of Mathe-
[21] W. S. Lovejoy, A survey of algorithmic methods for partially observed Markov decision processes, Annals of
[22] P. Poupart, Exploiting structure to efficiently solve large scale partially observable Markov decision processes,
Intelligence Research 24 (2005) 195–220.
[25] A. Mihailidis, J. Roger, M. Candido, J. Hoey, The coach prompting system to assist older adults with dementia
through handwashing: An efficacy study, BMC Geriatrics 8 (28).
[26] J. P. Wherton, A. F. Monk, Problems people with dementia have with kitchen tasks: the challenge for pervasive
computing, Interacting with Computers 22 (4) (2009) 253–266.
[27] P. Olivier, A. Monk, G. Xu, J. Hoey, Ambient kitchen: designing situated services using a high fidelity prototyping
environment, in: Proceedings of the ACM 2nd International Conference on PErvasive Technologies Related to
Assistive Environments, Corfu, Greece, 2009.
[31] H. Ryu, A. Monk, Will it be a capital letter: signalling case mode in mobile phones, Interacting with Computers 17
1492–1508.
[34] C. Pham, P. Olivier, Slice&dice: Recognizing food preparation activities using embedded accelerometers, in: Eu-
ropean Conference on Ambient Intelligence, Springer-Verlag, Berlin: Salzburg, Austria, 2009, pp. 34–43.
1994.