Available at www.ComputerScienceWeb.com
POWERED BY SCIENCE @ DIRECT®

Robotics and Autonomous Systems 44 (2003) 273–282

Robotics and Autonomous Systems

www.elsevier.com/locate/robot

Autominder: an intelligent cognitive orthotic system for people with memory impairment

Martha E. Pollack[a,*], Laura Brown[b], Dirk Colbry[c], Colleen E. McCarthy[d], Cheryl Orosz[a], Bart Peintner[a], Sailesh Ramakrishnan[e], Ioannis Tsamardinos[b]

[a] *Computer Science and Engineering, University of Michigan, Ann Arbor, MI 48109, USA*
[b] *Biomedical Informatics, Vanderbilt University, Nashville, TN 37232, USA*
[c] *Computer Science and Engineering, Michigan State University, East Lansing, MI 48824, USA*
[d] *Computer Engineering and Computer Science, California State University at Long Beach, Long Beach, CA 90840, USA*
[e] *QSS Group Inc./NASA Ames Research Center, Moffett Field, CA 94035, USA*

Abstract

The world’s population is aging at a phenomenal rate. Certain types of cognitive decline, in particular some forms of memory impairment, occur much more frequently in the elderly. This paper describes Autominder, a cognitive orthotic system intended to help older adults adapt to cognitive decline and continue the satisfactory performance of routine activities, thereby potentially enabling them to remain in their own homes longer. Autominder achieves this goal by providing adaptive, personalized reminders of (basic, instrumental, and extended) activities of daily living. Cognitive orthotic systems on the market today mainly provide alarms for prescribed activities at fixed times that are specified in advance. In contrast, Autominder uses a range of AI techniques to model an individual’s daily plans, observe and reason about the execution of those plans, and make decisions about whether and when it is most appropriate to issue reminders. Autominder is currently deployed on a mobile robot, and is being developed as part of the Initiative on Personal Robotic Assistants for the Elderly (the Nursebot project).

Keywords: Autominder; Cognitive orthotic systems; Reminder systems

1. Introduction

The world’s population is aging at a phenomenal rate. According to the United Nations Population Division, in 2000 about 606 million people, constituting approximately 10% of the world’s population, were over 60; by 2050, this percentage is expected to double to 2 billion people, or 21.4% of the population. Even more dramatic will be the increase in the percentage of people over 80, often called the “oldest old”. Today there are 69 million people in this category, constituting 1.1% of the world’s population; by 2050 the percentage will nearly quadruple to 4%, with 379 million people over the age of 80 alive [29].

It has been shown that the quality of life for people remaining in their own homes is generally better than for those who are institutionalized [23]; moreover, the cost for institutional care can be much higher than the cost of care for a patient at home. This paper describes Autominder, a system aimed at helping older adults with mild to moderate memory impairment remain in their homes longer. Many forms of memory impairment are strongly correlated with age

* Corresponding author.
E-mail address: pollackm@umich.edu (M.E. Pollack).

and can make it difficult for someone to organize and regularly perform their necessary daily activities, such as taking medicine correctly, eating, drinking water, toileting, performing routine hygiene, engaging in social and family activities, keeping medical appointments, and so on.[1] Autominder serves as a *cognitive orthotic*, providing its users (or "clients") with reminders about their daily activities. Most existing cognitive orthotics mainly issue alarms for prescribed activities at fixed times that are specified in advance. In contrast, Autominder is capable of much more flexible, adaptive behavior. It models its client's daily plans, tracks their execution by reasoning about the client's observable behavior, and makes decisions about whether and when it is most appropriate to issue reminders. The current version of Autominder is deployed on a mobile robot, and is being developed as part of the Initiative on Personal Robotic Assistants for the Elderly (the Nursebot project). We are also exploring alternative platforms for Autominder, such as distributed sensors and/or wearable devices.

In the next section, we provide a description of Autominder's architecture and its current platform. This is followed by three sections in which we discuss Autominder's main components: the Plan Manager, the Client Modeler, and the Personal Cognitive Orthotic (its reminder generation module). We then present a brief overview of other cognitive orthotic systems, and conclude with a description of our ongoing work on the system.

2. Autominder's architecture

To motivate Autominder's architecture, it is useful to provide a simple example of its interaction with a client. Consider a forgetful, elderly person with urinary incontinence who is supposed to be reminded to use the toilet every three hours, and whose next reminder is scheduled for 11:00. Suppose that, using on-board sensors, the robot on which Autominder is deployed observes the person enter the bathroom at 10:40 and stay there for a period of a few minutes.

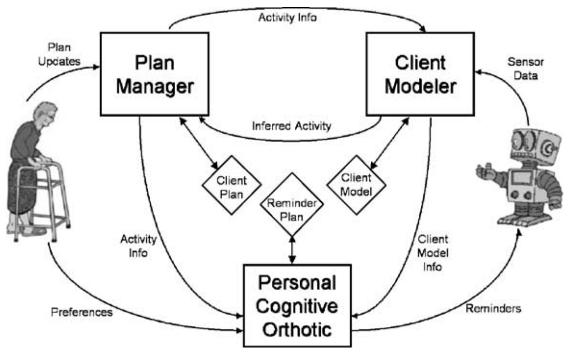

Fig. 1. Autominder architecture.

Autominder may conclude that toileting has occurred, and that, consequently, it should not issue a reminder at 11:00 as previously planned. Instead, the client's plan must be adjusted, so that the next scheduled toileting is to occur approximately three hours later, i.e., around 13:40. Flexibility is essential because a strict three-hour interval may not be optimal. For instance, if the client's favorite television program is aired from 13:30 to 14:00, it might be better to issue a reminder at 13:25, and provide a justification that mentions the television program: "Mrs. Smith, Why don't you use the toilet now? That way I won't have to interrupt you during your show."

To achieve this type of behavior, Autominder must maintain an accurate model of the client's daily plan, monitor its execution, and plan reminders accordingly. Autominder's architecture, depicted in Fig. 1, has three main components, one dedicated to each of these tasks. The Plan Manager stores the client's plan of daily activities in the *Client Plan*, and is responsible for updating it and identifying and resolving any potential conflicts in it. The Client Modeler uses information about the client's observable activities to track the execution of the plan, storing beliefs about the execution status in the *Client Model*. A reminder generation component called the Personal Cognitive Orthotic reasons about any disparities between what the client is supposed to do and what she is doing, and makes decisions about whether and when to issue reminders.

Autominder is currently embedded on a mobile robot named "Pearl", designed and built by researchers at Carnegie Mellon University. Pearl is constructed on a Nomadic Technologies Scout II robot, with a custom-designed and manufactured "head", and

[1] The medical community classifies such activities into three groups: Activities of Daily Living, Intermediate Activities of Daily Living, and Extended Activities of Daily Living. These distinctions do not matter for this paper.

includes a differential drive system, two on-board Pentium PCs, wireless Ethernet, SICK laser range finders, sonar sensors, microphones for speech recognition, speakers for speech synthesis, touch-sensitive graphical displays, and stereo camera systems. See [18] for details of Pearl's hardware and navigation algorithms.

3. The Plan Manager

The first of Autominder's three main components is the Plan Manager (PM). The technology in the PM grew out of our earlier work on plan management, in particular, the Plan Management Agent (PMA), a prototype intelligent calendar tool [20]. In Autominder, as in PMA, we found that it was essential that we be able to represent a rich set of temporal constraints in the plans: for example, we may need to express that the client should take a medication within 15 minutes of waking, and then eat breakfast between 1 and 2 hours later. We thus model plans as disjunctive temporal problems (DTPs) [19,24] and use a highly efficient algorithm that we developed for reasoning about them [25,28]. DTPs allow for both quantitative (metric) and qualitative (ordering) constraints, as well as conjunctive and disjunctive combinations of them. We have also recently developed an approach to handling conditional constraints [27], but we have not yet implemented these in the PM.

Formally, a DTP is defined to be a pair $\langle V, C\rangle$, where V is a set of variables (or nodes) whose domains are the real numbers, and C is a set of disjunctive constraints of the form: $C_i : x_1 - y_1 \leq b_1 \vee \cdots \vee x_n - y_n \leq b_n$ such that x_i and y_i are both members of V, and b_i is a real number. A solution to a DTP is an assignment to each variable in V such that all the constraints in C are satisfied. If a DTP has at least one solution, it is *consistent*. Within the PM, we assign a pair of DTP variables to each activity in the client's plan: one variable represents the start time of the activity, while the other represents its end time. We can easily encode a variety of constraints, including absolute times of events, relative times of events, and event durations, and can also express ranges for each of these. Fig. 2 gives some typical plan constraints encoded in the language of DTPs. The start (end) of a step A is denoted A_{S} (A_{E}). Note that to express a clock-time constraint, e.g., TV watching beginning at 18:00, we use a *temporal reference* (TR) *point*, a distinguished value representing some fixed clock time. In the figure, as well as in the Autominder system itself, the TR corresponds to midnight; the schedule is updated each day.

"Toileting should begin between 11:00 and 11:15." $660 \leq Toileting_S - TR \leq 675$
"Toileting takes between 1 and 3 minutes." $1 \leq Toileting_E - Toileting_S \leq 3$
"Watching the TV news can begin at 18:00 or 23:00." $1080 \leq WatchNews_S - TR \leq 1082 \vee$ $1380 \leq WatchNews_S - TR \leq 1382$
"The news takes exactly 30 minutes." $30 \leq WatchNews_E - WatchNews_S \leq 30$
"Medicine should be taken within 1 hour of finishing breakfast." $0 \leq TakeMeds_S - EatBreakfast_E \leq 60$
"Toileting and watching the news cannot overlap." $0 \leq WatchNews_S - Toileting_E \leq \infty \vee$ $0 \leq Toileting_S - WatchNews_E \leq \infty$

Fig. 2. Examples of the use of DTP constraints.

3.1. Plan initialization

The PM in Autominder is initialized in advance of its use with a specification of the client's daily plan, constructed by the client's caregiver, in consultation with the client. Different daily plans might be constructed, e.g., one for weekdays and one for weekends, with the appropriate plan loaded each morning, but here we will assume that there is just one daily plan.

We currently have a rather minimal GUI for specifying a daily plan.[2] It allows one to select pre-constructed plan fragments for routine activities from a library, and to then input specific temporal constraints on the steps in the selected fragments. Thus, a caregiver might begin construction of a typical daily plan by performing the following steps:

1. Select a pre-constructed plan fragment for breakfast. The fragment includes three steps—going to the kitchen, making breakfast, and eating breakfast—as well as temporal constraints that order these, causal links that capture their dependencies, and some default durations, e.g., that the eating step will take between 20 and 30 minutes.

[2] The same GUI can be used for modifying the plan once execution has begun.

2. Specify that the first step in the breakfast plan must begin by 7:00, and that the last step must be done by 8:30.
3. Select a pre-constructed plan fragment for taking medicine, which has only one step—take the medicine—with a default duration of 1 minute.
4. Specify an interstep constraint to ensure that the medicine taking occurs within one hour of finishing breakfast.

As each pre-constructed plan fragment or constraint is added, the PM performs step merging [26,30], that is, it checks to ensure the consistency of the daily plan being constructed and resolves any conflicts. To do this, it uses the same techniques for consistency checking that are used during plan execution; these techniques are described in the next subsection.

Although our current interface is sufficient for development and testing purposes, further work is required to develop more user-friendly interfaces to allow caregivers to specify plans.

3.2. Plan update

The primary role of the PM is to update the client's plan as the day progresses, ensuring its continued consistency. Update occurs in response to four types of events:

1. *The addition of a new activity to the plan*. The daily plan created at initialization provides a starting point for daily activities, but during the course of the day the client and/or her caregivers may want to make additions to the plan: for instance, to attend a bridge game or a newly scheduled doctor's appointment. At this point plan merging must be performed to ensure that the overall plan remains consistent. Suppose that the client plan initially specifies taking medicine sometime between 14:00 and 15:00, and that the client then adds a bridge game outside the apartment to begin at 14:30. The PM must update the plan so that the medicine-taking step precedes the client leaving for the bridge game. (We assume that the medicine must be taken at home.) If, in addition, the medicine-taking must occur at least two hours after each meal, the added restriction on when the medicine will be taken may also further restrict the time of lunch.
2. *The modification or deletion of an activity in the plan*. This is similar to the previous case: the bridge game might be cancelled, or the doctor's office may change the time of the appointment.[3] The type of required changes are like those needed when an activity is added. Note that the PM will add or tighten constraints if needed, but will not "roll back" (i.e., weaken) any constraints. Continuing the example above, if the bridge game were cancelled, the constraint that the medicine be taken between 14:00 and 14:30 would remain in the plan. More sophisticated plan retraction is an area of future research.
3. *The execution of an activity in the plan*. When the Client Modeler, discussed below, infers that an activity has been performed, it notifies the PM, which then updates the plan accordingly. Suppose again that medicine-taking is supposed to occur at least two hours after the completion of each meal. Upon learning that breakfast has been completed at 7:45, the PM can establish an earliest start time of 9:45 for taking the medicine.
4. *The passage of a time boundary in the plan*. Just as the execution of a plan step may necessitate plan update, so may the non-execution of a plan step. As a very simple example, suppose that the client wants to watch the news on television each day, either from 18:00 to 18:30 or from 23:00 to 23:30 p.m. At 18:00 (or a few minutes after), if the client has not begun watching the news, then the PM should update the plan to ensure that the 23:00–23:30 slot is reserved for that purpose. (To keep the example simple, assume that the client always wants to watch from the very beginning of the show.)

To perform plan update in each of these cases, the PM formulates and solves a DTP that includes the constraints already in the client plan augmented with the constraints imposed by the condition triggering the current update. The details of the problem formulation and the processes used to solve it are outside the scope of the current paper, but see [22]. Here we note only that for the problems we have worked with so far in the Autominder domain, the process of update al-

[3] Currently, we allow arbitrary changes to be made to the plan. In subsequent versions of the system, we will need to implement security mechanisms that, for instance, allow the user to make changes to social engagements but not the medicine-taking actions.

most always takes less than one second, except when it detects a plan failure.

4. The Client Modeler

The second major component of Autominder is the Client Modeler (CM). The job of the CM is to monitor the execution of the client plan, using information about observable actions of the client, as well as knowledge of what time it is and whether and when any reminders were issued. In our current implementation, the observable information is relatively impoverished—basically, the robot's on-board sensors provide the CM only with reports of the current location of the client (what room she is presently in). However, one can imagine enhancing the system in a variety of ways, e.g., with sensors on pill-bottle caps, on kitchen drawers and the refrigerator, on the toilet flusher, etc.

The task performed by the CM is a particularly challenging one because it involves monitoring a non-Markovian system. The client plans, recall, may simultaneously contain qualitative and quantitative relationships such as "taking medicine must occur before breakfast", "the television is usually turned on 10–15 minutes after finishing lunch", and "lunch is eaten 3–4 hours after breakfast". Thus, inferring the current state of the client's plan requires using information from several different time points.

To infer the state of the client plan, we need to be able to perform temporal reasoning under uncertainty. Previous approaches tend to fall into two categories. Those in first category, which derive from Time Nets [10], augment the nodes and arcs in a standard Bayesian network to implicitly model time. Time is encoded in the values of the nodes while temporal relationships are encoded in the conditional probability tables (CPTs) of the nodes. In this way, the Bayes net can maintain and infer probability distributions over when events occur or when properties of the environment change values. The chief limitation of formalisms in this category, with respect to monitoring, is their inability to model the fluctuating values of fluents, an ability required by the CM.

The second category, exemplified by Dynamic Bayes Nets (DBNs) [4] uses a more explicit model of time. Instead of simply ascribing temporal semantics to otherwise non-temporal elements in a formalism, these approaches annotate elements of a model with time points or intervals. In general, these approaches take a non-temporal causal structure and instantiate it once for each time point or interval modeled. Most authors call each instance of this causal structure a time slice. Arcs connecting nodes in different time slices encode the temporal dependencies between elements.

In some ways, DBNs are ideal for monitoring. They allow beliefs to continually evolve over a possibly infinite timeline. Also, fluents fit nicely into DBNs, which allow variables to change their values at each time slice transition. In domains where the Markov assumption is easily applied, DBNs prove to be an intuitive, compact, and relatively tractable model. However, because they rely on the Markov assumption, their ability to express temporal relationships is limited.

Because the Markov property is violated in Autominder, we use a new reasoning formalism called a Quantitative Temporal Bayesian Networks (QTBNs) [3]. A QTBN maintains two Bayes nets: a DBN containing all causal relationships and a standard Bayes net (similar to a Time Net) that represents all temporal relationships. Interface functions carry information between the two networks.

The structure of the QTBN is generated automatically from the client plan, making two initial assumptions: first, that all activities in the plan will, with reasonable probability, be executed by the client within the time range specified in the plan, and second, that the actual time of an activity can be described by a uniform probability density function over the range associated with that activity. This naive initial model can be refined using information provided by the caregiver.

The CM updates the client model whenever new sensor data arrives or a time boundary in the Time Net component is passed. In the former case, the new evidence is recorded in the DBN component, and then standard DBN rollup occurs. In the latter case, interface functions first transfer information between the DBN and the time net, working in both directions, and then rollup occurs (for details, see [3]). Whenever the result of CM update changes the value of a node representing an action so that it rises over a threshold belief value, the CM notifies the rest of the system. When the value enters an intermediate range, the system may ask for verification from the client, e.g., “Mrs. Smith, Did you just drink your water?”

We have not yet implemented learning capabilities in the CM, but hope to do so in the near future so that the CM can continually update its model of the client’s expected behavior. Additionally, we are currently developing a more theoretically sound inference mechanism for non-Markovian execution monitoring in general. Although the QTBN framework has so far worked in practice, the correctness of its inferences has not been formally demonstrated. We are thus developing an approach that has the same behavior as the QTBN, while being provably sound.

5. Reminder generation

The final component in Autominder is called the Personalized Cognitive Orthotic (PCO), and is responsible for deciding what reminders to issue and when [15]. In making its decisions, the PCO aims to balance four criteria: (i) ensuring that the client is aware of planned activities; (ii) achieving a high level of client and caregiver satisfaction; (iii) avoiding introducing inefficiency into the client activities and (iv) avoiding making the client overly reliant on the reminder system, which would have the detrimental effect of decreasing, rather than increasing client independence.

It would be straightforward to generate reminders if only the first criterion were of concern: one could simply issue a reminder for every activity at its earliest possible start time, perhaps repeating the reminder at regular intervals if the activity is not performed. However, such a policy might do a potentially poor job of satisfying the other criteria. It might do poorly on the second criterion because it ignore the preferences of the caregiver and the client, and so, for instance, might issue a reminder to use the toilet in the middle of the client’s favorite television program. It might do poorly on the third criterion because it fails to reason about the interactions between activities, and so might issue a reminder to the client to (get up from her chair and) go take her medicine, and then just when she has returned and sat down again, issue a reminder to (get up from her chair and) go use the toilet. And it might do poorly on the fourth criterion by never allowing the client to initiate activities on her own, instead always pre-emptively issuing reminders. (Arguably, the third and fourth criteria are special instances of the second.)

To achieve plans that have high quality with respect to all four of these criteria, the PCO adopts a local-search approach called Planning-by-Rewriting (PbR) [1]. It begins by creating an initial reminder plan that includes a reminder for each activity in the client plan at its earliest possible start time, and then performs local search, using a set of plan-rewrite rules to generate alternative candidate reminding plans. For example, one rewrite rule deletes reminders for activities that have low importance and that are seldom forgotten by the client. Another rule shifts the time of a reminder for an activity to its expected time, i.e. the time by which the client usually performs the activity. Yet another rule spaces out reminders for activities for the same type of action: for instance, instead of issuing eight reminders in a row to drink water, application of this rule would result in them being spaced out through the day.

The rewrite rules do not always result in valid reminder plans, i.e., plans that are consistent with respect to the constraints in the underlying client plan. Invalid reminder plans are detected and deleted, as are plans that omit reminders for critical activities such as taking medicine. The remaining newly generated reminder plans are evaluated using a heuristic function that takes into account factors such as the number of reminders, their timing, and their relative spacing. The reminder plan with the highest ranking is selected, and the process iterates, with rewrite rules now being applied to the selected plan. Iteration continues until either some reminder plan is judged to have quality exceeding some threshold, or there is an interrupt indicating that there has been a change to the client plan or to the client model. In the latter case, the entire reminder-plan generation process is restarted.

The PCO has also been designed to enable the generation of justifications for reminders, although this feature has not yet been fully implemented. Justifications are motivated by the hypothesis that client adherence to plans may be improved when the reasoning behind the existence and timing of a reminder is provided. For example, a reminder of the form “If you take your medicine now, you will not have to do it in the middle of your show”, may be more compelling than the simple message “Time for medicine”. In generating a justification for a reminder, the PCO can make use of the underlying client plan, the preferences of the caregiver and the client, and the particular rewrite rules used in creating the current reminder plan.

To perform a preliminary evaluation of the PCO, and in particular, to assess the suitability of the ranking heuristic used during local search, we constructed a client simulator that could be tuned for a set of stereotypical client execution patterns. The simulator could be tuned both for the client's reliability (how likely she was to perform activities without prompting) and her responsiveness (how likely she was to perform activities after a prompting). In all, we modeled eight different client behavior patterns. We considered four different reminder strategies: issuing reminders for all activities at the earliest possible start times; at the latest possible start times; making random selections of activities and times of reminders; and using the PCO's strategy. We conducted an experiment with cross-factorial design, simulating daily behavior in 32 (8×4) conditions, running two cases of each condition. We produced a graphical display of each outcome, which we provided to faculty from the School of Nursing at the University of Pittsburgh who are involved in the Nursebot project, and we asked them to complete questionnaires ranking the resulting behavior along various dimensions that correspond to the four evaluation criteria listed at the beginning of this subsection. Although the results are mixed for the simpler of the two cases (an extremely simple case where there are no interactions between activities), the PCO strategy was clearly superior in the more interesting and more realistic second case [16].

6. Related systems

The idea of using computer technology to enhance the performance of cognitively disabled people dates back nearly 40 years [6], and a number of systems exist to help people with cognitive impairment perform routine activities satisfactorily. Most of these systems can be classified as either scheduling aids, which help a person manage a number of distinct activities over an extended period of time, or as instructional cueing aids, which help a person navigate the generally consecutive steps of a single activity. Autominder is an example of a scheduling aid, while COACH [14] provides an example of an instructional cueing device. COACH assists a severely demented person with hand washing, by using a set of sensors to "watch" each step of the process, issuing a reminder if steps such as using soap are skipped or performed out of turn.

Early technology for scheduling aids included talking clocks, calendar systems, and similar devices, while more recent systems have provided reminders using the telephone [7], personal digital assistants [5,9] and pagers [8]. However, with the exception of PEAT [13], these systems generally function in a manner similar to alarm clocks: they provide alarms for prescribed activities at fixed times that are specified in advance by a client and/or her caregiver. PEAT was the first, and to the best of our knowledge, the only marketed cognitive orthotic system that relies on automated planning technology. PEAT, which is marketed primarily to patients with traumatic brain injury, is deployed on a handheld device, and provides visible and audible clues about plan execution. Like Autominder, PEAT maintains a detailed model of the client's plan and tracks its execution, propagating temporal constraints when the client inputs information specifying that an action has been performed. Also, upon the addition of a new activity, PEAT simulates the plan to uncover any conflicts, using the PROPEL planning and execution system [12] for this purpose. However, PEAT uses a less expressive planning language than Autominder; it does not attempt to infer the plan execution status; and it does not perform principled reasoning about what reminders to issue when, instead automatically providing a reminder for each planned activity.

Within the past year or two, several new projects aimed at designing intelligent cognitive orthotics have begun to emerge [2,11,17], and we look forward to increased activity in this area in the near future.

7. Conclusion

The Autominder system as described has been fully implemented, except where noted in the text. The system is written in Java and Lisp for a Wintel platform; we also have a Web-based interface for plan initialization and update. The current version of the system has been tested in the laboratory; an earlier version was integrated on the mobile robot Pearl and included in a preliminary field test conducted at the Longwood Retirement Community in Oakmont, PA, in June 2001. The goals of that test were, first, to ensure that the robot control software and the cognitive orthotic would work together, and second, to get an initial sense of the acceptability of such a system to older individuals. On both accounts, the test was successful. Admittedly, the older adults who enrolled in the studies were volunteers, and people likely to be intimidated or put off by this type of technology would not have volunteered. However, the people who did participate were uniformly excited about the system, as were the staff at Longwood, who made a number of suggestions to us about how this type of technology could also be used to assist them in their caregiving tasks. In the near future, we hope to conduct much more extensive field tests to exercise Autominder's reminding capabilities.

We have a number of plans for the continued development of Autominder, some of which were already mentioned in this paper. We have planned extensions to the individual reasoning modules, for example, adding the ability to handle conditional constraints to the PM, supplementing the PM with full-fledged planning capabilities to support replanning, and enabling the CM to learn the patterns of client activity over time, in order to better interpret observed behavior. As mentioned, we are also conducting work to develop more principled foundations for the reasoning done by the CM. An additional major area of current work involves the client-system interaction: we are presently investigating the use of reinforcement learning techniques to develop adaptive reminder policies. Additionally, we are actively exploring the implications of deploying Autominder on alternative hardware platforms. Although there are many advantages of using a robot, including the ability to piggyback on other capabilities, there are clearly also reasons to explore handheld and/or wearable devices as well as ubiquitous sensors to support cognitive orthotics. Finally, after our experiences with the staff at Longwood, we are interested in exploring the use of systems like ours within the facility-based setting. In that context, the system would coordinate the daily plans not only of a single person, but of multiple people, including both the residents and the staff that takes care of them.

Acknowledgements

This work is supported primarily by the National Science Foundation Grant IIS-0085796. Some of the supporting technology was developed with funding from the Air Force Office of Scientific Research (F49620-01-1-0066). The authors wish to thank the many members of the Nursebot team for their invaluable contributions: Donald Chiarulli, Jacqueline Dunbar-Jacob, Sandra Engberg, Jennifer Goetz, Sara Kiesler, Michael Montemerlo, Joelle Pineau, Joan Rogers, Nicholas Roy and Sebastian Thrun. Earlier versions of work described portions of this paper have presented in conference papers [3,15,21,22].

References

[1] J.L. Ambite, C.A. Knoblock, Planning by rewriting, Journal of Artificial Intelligence Research 15 (2001) 207–261.

[2] S. Carmien, MAPS: PDA scaffolding for independence for persons with cognitive impairment, 2002. http://www.cs.colorado.edu/~l3d/clever/projects/maps/.

[3] D. Colbry, B. Peintner, M.E. Pollack, Execution monitoring with quantitative temporal Bayesian networks, in: Proceedings of the Sixth International Conference on AI Planning and Scheduling, 2002, pp. 229–238.

[4] T. Dean, K. Kanazawa, A model for reasoning about persistence and causation, Computational Intelligence 5 (1989) 142–150.

[5] M.M. Dowds, K. Robinson, Assistive technology for memory impairment: Palmtop computers and TBI, in: Proceedings of the Braintree Hospital Traumatic Brain Injury Neurorehabilitation Conference, 1996.

[6] D.C. Englebart, A conceptual framework for the augmentation of man's intellect, in: Vistas in Information Handling, Spartan Books, 1963, pp. 1–13.

[7] R.H. Friedman, Automated telephone conversation to assess health behavior and deliver behavioral interventions, Journal of Medical Systems 22 (1998) 95–101.

[8] N.A. Hersh, L. Treadgold, Neuropage: the rehabilitation of memory dysfunction by prosthetic memory and cueing, NeuroRehabilitation 4 (1994) 187–197.

[9] B. Jonsson, A. Svensk, Isaac: a personal digital assistant for the differently abled, in: Proceedings of the Second TIDE Congress, 1995, pp. 356–361.

[10] K. Kanazawa, A logic and time nets for probabilistic inference, in: Proceedings of the Ninth National Conference on Artificial Intelligence, 1991, pp. 360–365.

[11] H. Kautz, L. Arnstein, G. Borriello, O. Etzioni, D. Fox, An overview of the assisted cognition project, in: Proceedings of the AAAI-02 Workshop on Automation as Caregiver: The Role of Intelligent Technology in Elder Care, 2002, pp. 60–65.

[12] R. Levinson, A general programming language for unified planning and control, Artificial Intelligence 76 (1995) 319–375.
[13] R. Levinson, PEAT—The planning and execution assistant and trainer, Journal of Head Trauma Rehabilitation 12 (1997) 769.
[14] E.F. LoPresti, A. Mihailidis, N. Kirsch, Assistive technology for cognitive rehabilitation: State of the art, 2002, Manuscript. alex_mihailidis@sfu.ca.
[15] C.E. McCarthy, M.E. Pollack, A plan-based personalized cognitive orthotic, in: Proceedings of the Sixth International Conference on AI Planning and Scheduling, 2000, pp. 213–222.
[16] C.E. McCarthy, A plan-based cognitive orthotic for reminding, Ph.D. Dissertation, Department of Computer Science, University of Pittsburgh, 2002.
[17] C. Miller, V. Riley, The independent lifestyle assistant, in: Proceedings of the XVII World Congress of the International Association of Gerontology, 2001.
[18] M. Montemerlo, J. Pineau, N. Roy, S. Thrun, V. Verma, Experiences with a mobile robotic guide for the elderly, in: Proceedings of the 18th National Conference on Artificial Intelligence (AAAI-02), Edmonton, AB, 2002, pp. 587–592.
[19] A. Oddi, A. Cesta, Incremental forward checking for the disjunctive temporal problem, in: Proceedings of the 14th European Conference on Artificial Intelligence, IOS Press, 2000, pp. 108–112.
[20] M.E. Pollack, J.F. Horty, There's more to life than making plans: plan management in dynamic environments, AI Magazine 20 (4) (1999) 71–84.
[21] M.E. Pollack, C. McCarthy et al., Autominder: a planning, monitoring, and reminding assistive agent, in: Proceedings of the Seventh International Conference on Intelligent Autonomous Systems, IOS Press, 2002, pp. 265–272.
[22] M.E. Pollack, Planning technology for intelligent cognitive orthotics, in: Proceedings of the Sixth International Conference on Automated Planning and Scheduling, AAAI Press, Menlo Park, CA, 2002, pp. 322–331.
[23] A. Rivlin, J. Wiener, Caring for the Disabled Elderly: Who Will Pay, The Brookings Institute, 1988.
[24] K. Stergiou, M. Koubarakis, Backtracking algorithms for disjunctions of temporal constraints, Artificial Intelligence 120 (2000) 81–117.
[25] I. Tsamardinos, M.E. Pollack, Efficient solution techniques for disjunctive temporal reasoning problems, Artificial Intelligence, to appear. doi:10.1016/S0004-3702(03)00113-9.
[26] I. Tsamardinos, M.E. Pollack, J.F. Horty, Merging plans with quantitative temporal constraints, temporally extended actions, and conditional branches, in: Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling, 2000, pp. 264–272.
[27] I. Tsamardinos, T. Vidal, M.E. Pollack, CTP: a new constraint-based formalism for conditional, temporal planning. Constraints 8 (4) (2003), to appear.
[28] I. Tsamardinos, Constraint-based temporal reasoning algorithms with applications to planning, Ph.D. Dissertation, University of Pittsburgh Intelligent Systems Program, Pittsburgh, PA, 2001.
[29] United Nations, World population prospects: the 2000 revision highlights. United Nations Population Division, 2001. http://www.un.org/esa/population/publications/wpp2000/wpp2000h.pdf.
[30] Q. Yang, Intelligent Planning: A Decomposition and Abstraction Based Approach, Springer, New York, 1997.

Martha E. Pollack is Professor of Computer Science and Engineering at the University of Michigan. She was previously a professor at the University of Pittsburgh and, before that, a Senior Computer Scientist at the AI Center, SRI International. Pollack received a bachelor's degree from Dartmouth College and M.S.E. and Ph.D. degrees from the University of Pennsylvania. She has conducted research and published in the areas of automated planning, plan management, agent-based systems, natural-language discourse, and computational models of rationality; most recently, she has been focusing on applied work on cognitive orthotic systems. Recipient of several major awards including the Computers and Thought Award (1991), and the University of Pittsburgh Chancellor's Distinguished Research Award (2000), Pollack is also an elected fellow of American Association for Artificial Intelligence (1996), and currently serves as executive editor of the *Journal of Artificial Intelligence Research*.

Laura Brown completed her undergraduate education at Swarthmore College in 2000 with a degree in Engineering and a concentration in Computer Science. She then received the M.S. from the University of Michigan. She is currently attending the Ph.D. program in Biomedical Informatics, Vanderbilt University. Her research interests includes artificial intelligence, machine learning, and bioinformatics.

Dirk Colbry is working on his Ph.D. in Computer Science at Michigan State University, where his research focuses on artificial intelligence, machine vision and 3D modeling. He received the M.S.E. in Computer Science from the University of Michigan in 2002, where he worked on Autominder's client modeler. He also received the B.S.E. in mechanical engineering from The Georgia Institute of Technology in 1997.

Colleen E. McCarthy received her Ph.D. in Computer Science from the University of Pittsburgh in 2002; her dissertation research focused on Autominder's reminder-generation component. Her bachelor's degree is from Kent State University. McCarthy is currently an assistant professor in the Computer Engineering and Computer Science Department at California State University at Long Beach.

Cheryl Orosz is currently pursuing M.S. and Ph.D. degrees in Computer Science and Engineering at The University of Michigan. She received the A.B. degree in linguistics from The University of Michigan in 1992. Her research interests include plan representation, temporal reasoning, plan monitoring and recognition, multi-agent systems, and probabilistic reasoning.

Bart Peintner is currently a Ph.D. candidate in Computer Science at the University of Michigan, where he received the M.S.E. in Computer Science in 2001. He has been involved in the design and development of Autominder's client modeler, and his dissertation research focuses on advanced techniques for execution monitoring. Peintner also holds a B.S. in Electrical Engineering from Kansas State University.

Sailesh Ramakrishnan received his B.Tech. from IIT Madras and M.S. from the University of Pittsburgh. He is pursuing the Ph.D. at the University of Michigan while working at NASA Ames Research Center. His current research interests include temporal planning under uncertainty.

Ioannis Tsamardinos received his bachelor's degree in Computer Science from the University of Crete, and his M.S. and Ph.D. degrees from the Intelligent Systems Program of the University of Pittsburgh. His Ph.D. dissertation research improved both the efficiency and expressiveness of constrained-based temporal reasoning. Tsamardinos is currently working as an assistant professor in the Department of Biomedical Informatics at Vanderbilt University, pursuing research on machine learning, feature selection, and causal induction in biomedical domains.