# MavHome: An Agent-Based Smart Home

Diane J. Cook, Michael Youngblood, Edwin O. Heierman, III,
Karthik Gopalratnam, Sira Rao, Andrey Litvin, and Farhan Khawaja
Department of Computer Science Engineering
University of Texas at Arlington
Email: cook@cse.uta.edu

## Abstract

*The goal of the MavHome (***M***anaging* **A***n Intelligent* **V***ersatile* **Home***) project is to create a home that acts as an intelligent agent.*[1] *In this paper, we introduce the MavHome architecture. The role of prediction algorithms within the architecture is discussed, and a meta-predictor is presented which combines the strengths of multiple approaches to inhabitant action prediction. We demonstrate the effectiveness of these algorithms on smart home data.*

## 1. Introduction

The MavHome smart home project focuses on the creation of an environment that acts as an intelligent agent, perceiving the state of the home through sensors and acting upon the environment through device controllers [2]. The agent's goal is a function that maximizes comfort and productivity of its inhabitants and minimizes operation cost. In order to achieve this goals, the house must be able to predict, reason about, and adapt to its inhabitants.

MavHome operations can be characterized by the following scenario. At 6:45am, MavHome turns up the heat because it has learned that the home needs 15 minutes to warm to optimal waking temperature. The alarm sounds at 7:00, after which the bedroom light and kitchen coffee maker turn on. Bob steps into the bathroom and turns on the light. MavHome records this interaction, displays the morning news on the bathroom video screen, and turns on the shower. When Bob finishes grooming, the bathroom light turns off while the kitchen light and display turn on, and the news program moves to the kitchen screen. During breakfast, Bob requests the janitor robot to clean the house. When Bob leaves for work, MavHome secures the home, and starts the lawn sprinklers despite knowing the 30% predicted chance of rain. Because the refrigerator is low on milk and cheese, MavHome places a grocery order. When Bob arrives home, his grocery order has arrived and the hot tub is waiting for him.

[1]This work was supported by National Science Foundation grants IIS-0121297 and EIA-9820440.

## 2. MavHome Architecture

A number of capabilities are needed for our smart home scenario, requiring integration of technologies from databases, robotics, machine learning, mobile computing, and multimedia computing. To scale to large environments, the MavHome agent can be decomposed into lower-level agents responsible for subtasks within the home, including robot and sensor agents, and MavHome can dynamically reorganize the hierarchy to maximize performance.

The desired smart home capabilities must be organized into an architecture that seamlessly connects these components while allowing improvement in any of the underlying technologies. Figure 1 shows the architecture of a MavHome agent. The technologies within each agent are separated into four cooperating layers. The **Decision** layer selects actions for the agent to execute based on information supplied from the other layers through the Information layer. The **Information** layer gathers, stores, and generates knowledge useful for decision making. The **Communication** layer facilitates the communication of information, requests, and queries between agents. The **Physical** layer contains the hardware within the house including individual devices, transducers, and network hardware. Because the architecture is hierarchical, the Physical layer may actually represent another agent in the hierarchy.

Perception is a bottom-up process. Sensors monitor the environment (e.g., lawn moisture level) and, if necessary, transmit the information to another agent through the Communication layer. The database records the information in the Information layer, updates its learned concepts and predictions accordingly, and alerts the Decision layer of the presence of new data. During action execution, information flows top down. The Decision layer selects an action (e.g.,

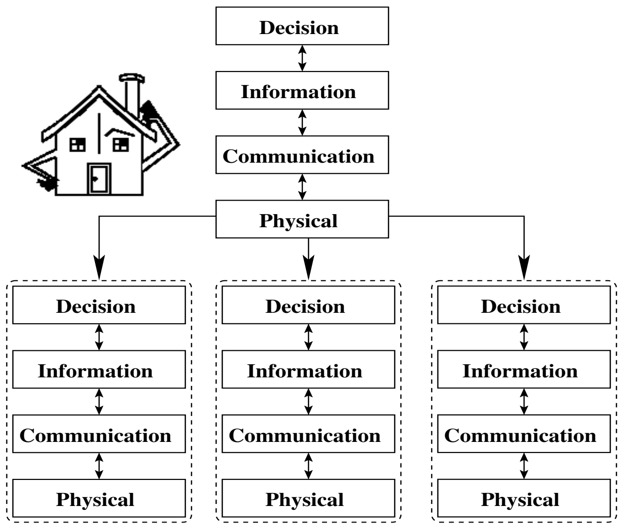

**Figure 1. MavHome agent architecture.**

run the sprinklers) and relates the decision to the Information layer. After updating the database, the Communication layer routes the action to the appropriate effector to execute. If the effector is actually another agent, the agent receives the command through its effector as perceived information and must decide upon the best method of executing the desired action. A specialized interface agent provides interaction capabilities with users and with external resources such as the Internet. Agents can communicate with other agents using the hierarchical flow of control and information shown in Figure 1.

Several smart home-related projects have been initiated by research labs. The Georgia Tech Aware Home [3] and the MIT Intelligent Room [6] include an impressive array of sensors to determine user locations and activities within an actual house. The Neural Network House at the University of Colorado Boulder [4] employs a neural network to control heating, lighting, ventilation, and water temperature in a manner that minimizes operating cost. The interest of industrial labs in smart home and networked appliance technologies is evidenced by the creation of Jini, Bluetooth, and SIP standards, and by supporting technologies developed by IBM, Xerox, Cisco, and Microsoft. MavHome is unique in combining technologies from artificial intelligence, machine learning, databases, mobile computing, robotics, and multimedia computing to create an entire smart home that acts as an intelligent agent.

The MavHome architecture has been implemented using a CORBA interface between software components and powerline control for most electric devices. In the current MavHome lab environment, students register their presence and MavHome begins collecting data on their activities. MavHome features include collection of activities in

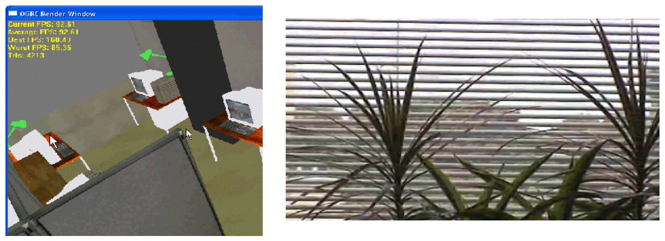

**Figure 2. ResiSim and automated blinds.**

a database, prediction of inhabitant actions, identification of inhabitants from observed activities, mobility prediction, robotic assistants, multimedia adaptability, and intelligent control and visualization of home activities through the ReSiSim 3D simulator. Because components are connected using CORBA, off-the-shelf devices can be controlled by MavHome as easily as customized components such as our automated mini-blinds, built using stepper motors from 5 1/4" floppy drives.

## 3. Inhabitant Action Prediction

An intelligent environment must be able to acquire and apply knowledge about its inhabitants in order to adapt to the inhabitants and meet the goals of comfort and efficiency. These capabilities rely upon effective prediction algorithms. Given a prediction of inhabitant activities, MavHome can decide whether or not to automate the activity or even find a way to improve the activity to meet the house goals.

Specifically, the MavHome intelligent agent needs to predict the inhabitant's next action in order to automate the repetitive tasks for the inhabitant. Patterns observed in past inhabitant activities can be used to determine how to control devices throughout the home. Here we present prediction algorithms suitable for that task.

### 3.1. Prediction Using Sequence Matching

Our SHIP algorithm matches the most recent sequence of events with sequences in collected histories. When the inhabitant issues a command to a device, it is recorded in the inhabitant *history*. A match identifies a sequence in history that matches the immediate event history. A match queue is maintained to ensure a near-linear run time.

SHIP consists of two steps. First, the match queue is updated when a new action is recorded. At time $t$ in state $s$ we compute $l_t(s, a)$, the length of the longest sequences that end with action $a$ in state $s$ and match the history sequence immediately prior to time $t$. In addition, we define a frequency measure $f(s, a)$ which represents the number of times the action $a$ has been taken from the current state.

In the second step, SHIP evaluates the matches based on a combination of match length and frequency, according to the equation

$$R_t(s,a) = \alpha \frac{l_t(s,a)}{\Sigma_t l_t(s,a_i)} + (1-\alpha)\frac{f(s,a)}{\Sigma_t f(s,a_i)}.$$

SHIP returns the action $a$ corresponding to the greatest $R_t(s,a)$ value as its prediction. The user can refine the algorithm by specifying a decay factor for old history or allowing an inexact match.

SHIP has been tested using synthetic smart home data and real data collected by students using X10 controllers in their homes. In these experiments, SHIP yields a predictive accuracy as high as 53.4% on the real data and 94.4% on the synthetic data. SHIP's performance on the real data climbs to over 80% if we consider SHIP's top three matches. The advantages of SHIP are its design simplicity and ability to incrementally use all collected history. A key disadvantage is the fact that the entire action history must be stored and processed off line, which is not practical for large prediction tasks over a long period of time.

## 3.2. Compression-Based Prediction

Our second prediction algorithm, Active LeZi (ALZ), uses information theory principles to process historical action sequences. Because we characterize inhabitant-device interaction as a Markov chain of events, we can utilize a sequential prediction scheme that has been shown to be optimal in terms of predictive accuracy for this type of prediction problem. Because ALZ is based on the incremental LZ78 compression technique, it is also an online algorithm.

The LZ78 text compression algorithm parses an input string $s_1, s_2, \ldots, s_i$ into substrings $w_1, w_2, w_{c(i)}$ such that for all $j > 0$, the prefix of the substring $w_j$ is equal to some $w_i$ for $1 < i < j$. The algorithm maintains statistics for all contexts of each phrase [7]. This information is used to compress and reconstruct text strings in an online fashion.

Active LeZi enhances the original LZ78 algorithm by recapturing information that would be lost across phrase boundaries. To do this, we maintain a variable-length window of processed symbols. The length of this window is equal to the length of the longest phrase seen so far. We gather statistics on all of the possible contexts seen based on this information. The resulting algorithm also gains a better rate of convergence to optimal predictive accuracy.

To perform prediction, ALZ calculates the probability of each action occurring in the parsed sequence, and predicts the action with the highest probability. To enhance prediction, ALZ incorporates ideas from the Prediction by Partial Match (PPM) family of predictors. PPM algorithms consider weighted Markov models of different orders to build a probability distribution, already successfully explored for mobility prediction [1]. ALZ starts by building an order-$k$ Markov model, then employs the PPM strategy to gather information from all lower-order models to determine the probability value of the next symbol.

ALZ has been tested on synthetic data generated for 30 days representing action sequences for weekday and weekend scenarios. The algorithm yielded 87% accuracy over this test data.

The ability of ALZ to use information from different order models is a strength of this approach, as is the ability to process information in an incremental manner. However, this algorithm does not incorporate actual event times into the prediction. For predictions that require this information TMM is useful.

## 3.3. Prediction Using a Task-based Markov Model

The TMM algorithm identifies high-level tasks in action sequences to help direct the creation of a Markov Model for action prediction. A simple Markov Model can be generated from collected action sequences and used to predict the next action given the current state of the agent. The state information captures the individual device states as well as the time of day and action date. However, additional information about the context in which activities were performed may be useful in further directing the prediction.

The sequence of events, or actions, is first partitioned into individual tasks. A change in tasks is identified by gaps in activities and changes in location of the actions being performed. Second, a k-means clustering algorithm is used to cluster the partitioned task sequences into sets of similar tasks. Task meta-features are supplied to the clusterer that include the length in time and length in number of actions of the task sequence, the number of devices used in the task, and the number of locations covered by the task. The output of the clustering algorithm is a collection of task sets, each of which can be labeled as a separate task type. Task information seeds the transition probabilities for the Markov Model based on their co-membership in a task cluster.

The TMM algorithm was tested on synthetic data generated for 30 days, using separate scenarios for weekdays and weekends. The algorithm generated 74% predictive accuracy on this data. Note that while the predictive accuracy is lower than for SHIP and ALZ, the complexity of the problem is increased because of the reasoning about time as well as action sequences.

## 3.4. Episode Discovery

The SHIP, ALZ, and TMM algorithms are useful in identifying likely activities of a smart home inhabitant. This information can be used to automate interactions with the home, removing the need for manual control of devices. A wrong prediction, however, can be annoying or detrimental if the inhabitant must reverse the action executed by the house or repair damage caused by a faulty decision.

Instead of identifying and automating each inhabitant pattern, we describe here a data mining algorithm, called Episode Discovery (ED), that identifies *significant episodes* within an inhabitant event history. A significant episode can be viewed as a related set of device events that may be ordered, partially ordered, or unordered. A significant episode occurs at some regular interval or in response to other significant episodes called *triggers*. Actions can then be automated based on the significance of mined patterns as well as the predictive accuracy of the next event.

Our approach is based on the work of Srikant and Agrawal [5] for mining sequential patterns from time-ordered transactions, modified to process ordered or unordered home interactions. In addition, many of the episodes in our environment will occur daily or weekly, and need to be recognized for this regularity. In our MavHome scenario, ED detects regular sequences including {{HeatOn (daily)}, {AlarmOn, AlarmOff, BedroomLightOn, CoffeeMakerOn, BathRoomLightOn, BathRoomVideoOn, ShowerOn, HeatOff (daily)}, {SprinklerOn (weekly)}. Other activities, such as the robot activation, would not be identified by ED as significant because they do not occur with any predictable regularity.

To mine the data, a window is slid over the event history. ED evaluates the unordered or ordered set of events within the window using the minimum description length (MDL) principle, which favors sequences that minimize the description length of the input sequence once it is compressed by replacing each instance of the discovered pattern with a pointer to the pattern definition. The regularity factor (daily, weekly, monthly) helps compress the data and thus increases the value of a pattern.

To improve the quality of predictions, action sequences are first filtered using ED. If a sequence is considered significant, then predictions can be made for events within the sequence window. Using ED as a filter for a frequency-based prediction algorithm, the predictive accuracy increases from 47% to 100%. These results show that ED can be used to aid in the automation of device interactions, as described by our MavHome scenario.

### 3.5. Meta-Prediction

As is evident from our discussions, there are many approaches to smart home prediction. Because we want to draw from the strengths of the alternative approaches, our main MavHome prediction algorithm is actually a meta-predictor, called Predict$^2$.

The Predict$^2$ algorithm uses a backpropagation neural network to learn a confidence value for each prediction algorithm based on the gathered data and accuracy of the algorithm on this data, along with meta features such as the amount of training data, the number of devices in the environment, the number of inhabitants in the home, and the significance of the event as determined by ED. A weighted voting scheme using each individual prediction algorithm generates the final prediction. A CORBA interface between Predict$^2$ and the MavHome architecture allows the home to draw upon the prediction when needed.

## 4. Conclusions

In this paper we present the MavHome smart home architecture, which allows a home to act as an intelligent agent. As part of the MavHome architecture, several prediction algorithms are introduced that play critical roles in an adaptive and automated environment such as MavHome. Results from synthetic and real collected smart home data indicate that the predictive accuracy is high even in the presence of many possible activities. The Predict$^2$ algorithm allows each of these prediction approaches to play a role in predicting inhabitant activity within MavHome.

We have demonstrated the effectiveness of these algorithms on smart home data. Our next step will be to test MavHome on increasingly complex environments with multiple inhabitants.

## References

[1] A. Bhattacharya and S. Das. Lezi-update: An information-theoretic framework for personal mobility tracking in PCS networks. *Wireless Networks Journal*, 8(2-3):121–135, 2002.

[2] S. Das, D. J. Cook, A. Bhattacharya, I. E O Heierman, and T.-Y. Lin. The role of prediction algorithms in the MavHome smart home architecture. *IEEE Wireless Communications*, 2003.

[3] V. Lesser, M. Atighetchi, B. Benyo, B. Horling, A. Raja, R. Vincent, T. Wagner, X. Ping, and S. X. Zhang. The intelligent home testbed. In *Proceedings of the Autonomy Control Software Workshop*, 1999.

[4] M. Mozer. The neural network house: An environment that adapts to its inhabitants. In *Proceedings of the AAAI Spring Symposium on Intelligent Environments*, pages 110–114, 1998.

[5] R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. In *Proceedings of the Fifth International Conference on Extending Database Technology (EDBT)*, 1996.

[6] M. C. Torrance. Advances in human-computer interaction: The intelligent room. In *Working Notes of the CHI 95 Research Symposium*, 1995.

[7] J. Ziv and A. Lempel. Compression of individual sequences via variable rate coding. *IEEE Transactions on Information Theory*, IT-24:530–536, 1978.