Preliminary Study of Adaptive Decision-Making System for Vocal Command in Smart Home

Alexis Brenon, François Portet, Michel Vacher

Preliminary Study of Adaptive Decision-Making System for Vocal Command in Smart Home

Alexis Brenon
Univ. Grenoble Alpes, LIG
F-38000 Grenoble, France
alexis.brenon@imag.fr

François Portet
Univ. Grenoble Alpes, LIG
F-38000 Grenoble, France
francois.portet@imag.fr

Michel Vacher
CNRS, LIG
F-38000 Grenoble, France
michel.vacher@imag.fr

***Abstract*—In smart homes, prediction and decision are often defined a priori and require tuning from the user, which can be tedious, and complex. However, these smart homes have the ability to analyze the user behavior and to modify their decisions automatically. We present a preliminary study that tests a decision system from voice command and the user's context, which is modified by reinforcement learning. The system ran on a realistic corpus, which shows the interest of such an adaptation.**

I. Introduction

The smart home aims to improve the user control and experience. To provide this enhanced control, these systems perceive their environment and make decisions about it. This decision can take place reactively – after a user command – or proactively – to alert about a risky situation. This perception is useful not only to make a decision but also to adapt the decision to the context, characterized by the current conditions.

Last years, the community has shown a new keen interest in voice controlled smart homes [1]. This kind of control is based on a vocal user interface, which allows a 'natural' communication with the system and, which is well adapted for disabled people or emergency situation (hands-free and distant interaction) [2]. In this framework, vocal commands may be 'turn on the light', 'check the door', 'call my daughter', etc. and the context must be used to remove the ambiguity of the command. If the command 'turn on the light' is pronounced in a bedroom with many lamps, the user doesn't precise which lamp(s) to turn on because he expects that his counterpart would guess it. Moreover, it would not be natural to ask the user to specify any details of a command; it is up to the system to check the context to take the most relevant decision.

The state of the art shows some approaches deploying a decision-making system in a smart home. For example Moore *et al.* [3] has developed a system which uses a set of fuzzy rules to find the best decision based on the context. Kofler *et al.* [4] use description logic to define the behavior of a context-aware system while Leong *et al.* [5] model the behavior of a perceptive system with ECA rules (Event-Condition-Action). Other approaches based on Bayesian networks have been implemented [6] to reflect the uncertainty of the data. However, in these proposals, the system adapts to the current situation and does not take into account changes that may occur. Indeed, depending on the season, time of day, the arrival of a guest or a disability, patterns and contexts of use may change. In addition, to reflect the preferences of the user, these systems need to be configured, which can be very complex [7]. Moreover, some misinterpretations of a vocal command or of the context may be identified by the system that could automatically correct it to avoid the call for a fine and tedious parameterization. These constraints militate for a flexible, scalable system that can adapt to the changing behavior of individuals.

This was the case in the ACHE project (*Adaptive Control of Home Environment*) [8] in which a home automation control system was learned, observing how the user deals with the system decisions. Actions were predicted by a neural network and the usefulness of the action was modified using reinforcement learning techniques. However, this project does not seem to have been brought to completion and did not explicitly uses the context to make its decision.

The aim of the study presented in this article is to develop a system which adapts itself to the user and the environment in the long term. The article presents in section II the Q-learning method and how it is particularly well suited to the problem of decision-making following a voice command. Section III describes the testing of this method on a corpus acquired in realistic condition. The article ends with a discussion of the results and the perspectives for improvement.

II. Method

We make a brief introduction to Q-learning and how it was implemented for decision-making system in a smart home.

A. Reinforcement Learning & Q-Learning

Reinforcement learning is a machine learning technique that allows an agent to learn its behavior through a feedback from its environment. Thus, a reinforcement learning problem is based on three main components [9]: (1) the **environment** on which agent acts; (2) the **reinforcement function** which defines the objective of the agent, assigning a reward for each state-action pair; (3) the **value function** associating the expected reward until final state (where the system succeeds or fails), to each state of the agent.

The goal of the agent is to determine a strategy, based on the value function, by a series of trial-and-error interactions. The

first approaches were based on the principles of dynamic programming to modify the value function (usually represented as an association table) until convergence.

In 1989, Watkins published an extension to the dynamic programming approach called the *Q-Learning* [10]. He introduced the notion of *Q-Value* not associated with a system state but with a state-action pair. Similarly, the notion of value function was replaced by the Q-function. It also defined the equation for updating the values of the Q-function, according to the Bellman equation used for dynamic programming.

$$Q^{n+1}_{s_t,a_t} = Q^{n}_{s_t,a_t} + \alpha \left(r(s_t,a_t) + \gamma \max_{a_{t+1}} Q^{n}_{s_{t+1},a_{t+1}} \right) \quad (1)$$

where Q^n denotes the Q-function at step n, s_t the state of the environment at time t, a_t an action at time t, α the learning factor used to vary the learning speed of the agent, $r(s_t,a_t)$ the reward received for action a_t, γ the discount factor and $\max_{a_{t+1}} Q(s_{t+1},a_{t+1})$ the value of Q for the best action a_{t+1} in the next state s_{t+1}.

This new approach limited the computing cost of reinforcement learning for non-deterministic environments. While the use of dynamic programming implies the computation of a sum of a theoretically infinite number of interactions, the Q-Learning only requires the values of the current state and of the next one to update the Q-function. However, the Q-value associated with a state is provided to converge to the optimal value after enough interactions, as in previous approaches.

B. Voice Controlled Smart Home

A smart home is a place that integrates sensors and actuators (from home automation) and is capable of *perceiving* the environment to *act* on it reactively or proactively [11].

A voice control application illustrates this need for perception and action. For example, if a user in a smart home pronounces 'Turn on the light', another person will easily interpret this request and take action. A computer system must infer alone the lamp that is implicit in the statement (if there are several lamps) and its intensity (in the case where it would be adjustable). This missing information must be retrieved from the knowledge of the context: the system must infer the user's location to turn on the light in the right room and must also deduct, from his activity, the location and the light intensity.

In our study, we consider that possible actions are: turn on/off the light/radio, open/close the blinds/curtains, tell the temperature/time, request a video/phone/emergency call.

These actions are a subset of possible actions defined after a user study [12]. Of course, this set of actions should be tailored to each user and housing, but this predefined list was useful for the evaluation of the system. This study also identified voice commands using a simple grammar as shown in Figure 1. Each command begins with a unique keyword, *'Nestor'*.

This kind of environment seems particularly suited for Q-learning as the sets of states and actions, though potentially great, are finite and discrete.

```
basicCmd         = key initiateCommand object | key emergencyCommand
key              = "Nestor"
initiateCommand  = "ouvre" | "ferme" | "teins" | "allume" | "appelle" " | "donne"
emergencyCommand = "au secours" | " l'aide"
object           = [determiner] ( device | person | organisation)
determiner       = "mon" | "ma" | "le" | "la" | "les" | "un" | "des" | "du"
device           = "lumire" | "store" | "rideau" | "radio" | "heure" | "temprature"
person           = "fille" | "fils" | "femme" | "mari" | "mdecin"
organisation     = "samu" | "secours" | "pompiers" | "suprette" | "supermarch"
```

Figure 1. Vocal commands grammar sample (in French).

III. Experiment

This section presents the realized scenarios and realistic data from a previous study, details how learning has been done and the evaluation of the experiment.

A. Scenario of the use of a voice controlled smart home

We consider in this study the DOMUS smart home designed by *Grenoble Informatics Laboratory* (LIG) [13]. This 30 square meters home, includes a bathroom, a kitchen, a bedroom, and a study. All these rooms are equipped with sensors and actuators such as infrared motion sensors, touch sensors, video cameras (only used for annotation purposes), etc. In addition, seven microphones were placed in the ceiling for audio capture. The apartment is fully usable and can accommodate a resident for several days. More than 150 sensors are managed in the apartment to provide different services (*e.g.* light, opening/closing shutters, media management, etc.).

To collect decision data in context, several participants were recruited to play scenarios of the daily life in the apartment. Participants were asked to issue voice commands to activate the actuators in the smart home. The objective of this experiment was to test a smart controller [14] in real situations corresponding to voice commands spoken by the user. The situations that we considered in the study were: (1) to clean up the apartment, (2) prepare and eat a meal, (3) converse via video conference (4) do leisure activities (reading), (5) take a nap. To guide the participants in the realization of the experiment, the grammar voice command was provided (see Figure 1) with a scenario of everyday scenes (Figure 2). This scenario was designed to last about 45 minutes, however, there was no constraint on the execution time. The first four parts put the user in situations of daily activities while speaking a voice command. Each participant received a list of actions to make and voice commands to pronounce. Each participant had to use a voice command, repeating up to 3 times in case

Go to the kitchen, ask for the temperature:
Nestor donne la température
Take a snack, put the dish in the sink, ask for the time:
Nestor donne moi l'heure
You realized it's late, you need to go shopping
Before leaving, you want to turn off the lights:
Nestor éteins la lumière
You also want to close the blinds:
Nestor baisse les stores

Figure 2. A sample of an example scenario that a participant had to do.

of system failure. A Wizard of Oz was used in the case of persistent failure.

A total of 15 people (9 women, 6 men) participated in the experiment to record 11 hours of data. The average age of participants was 38 ± 13.6 (19–62). All experiments were video recorded but only for annotation purposes. In the study presented in this article, the home automation control corpus is composed of a set of situations (human activity, location, condition of the apartment), voice control statements and the corresponding home automation commands. This data set is extracted from the Sweet-Home corpus [15].

B. Learning

Our model is learned on a first corpus, called *train*, before to be evaluated using cross-validation technique on a *test* corpus.

1) Corpus description: Each of the data sets is one or more text files following this scheme:

```
  usr_cmd usr_loc usr_act -> expctd_cmd expctd_loc

Examples:
  blind - open kitchen none
         ->blind - open kitchen
  light - on kitchen cook
         ->light - on kitchen - sink
```

The focus is on taking a decision when the user calls the system through a vocal command. Thus, each file entries necessarily contains a valid *usr_cmd* field as the *usr_loc* field which corresponds to the user location when he interacts with the system. However, the *usr_act* field, which symbolizes the activity of the user, can take a null value that means that we do not know what activity is being performed.

Each state of the system (which here corresponds to the current context) is then associated with an expected output consisting of two pieces of information: the desired command – turn on the lights – and the place of the action – the ceiling or bedside lamp. This information is defined, for the *train* corpus, by a set of rules and in the case of the *test* corpus by a domain expert.

The *train* corpus is a corpus which should enable us to learn a first model of our environment. For this, it must be relatively large. As there is little decision data corpus, the training corpus was generated automatically by following the method used in [14]. For this, we used a script that runs through all possible states of our system and deducts the expected decision from a set of logical rules. From the 300 states of our system, about 400 samples are generated.

The *test* corpus is based on real data extracted and annotated from experiments performed during the Sweet-Home project [15]. During cross-validation, we leave data from one subject for evaluation, tuning our model using data from fifteen subjects. Each subject did 25 interactions on average leading to a total of 407 interactions covering 37 states.

2) Experiment: The experiment is conducted in 2 phases: training, and cross-validation. Cross-validation is executed on a sixteen folds corpus. Each phase is broken down into 10 steps at the end of which the awards received are integrated for learning. This allow us to track the performances all along the experiment.

For the training phase, we use the *train* corpus to simulate many interactions. We try to ensure that all states have been met several times allowing the system to explore different actions outcomes. Then, we get a Q-values table representing a first model of the environment.

We use this model as a base for the evaluation phase. For each fold of the *test* corpus, the system adapts itself using data from fifteen subjects before being evaluated on the left out one. The results of the experiment are saved to extract metrics to evaluate the performance of our system.

In both cases, learning process in the same. We provide a state to the system which returns an action. If expected and returned actions are the same, the system is rewarded and we provide a new state. Else, the system is punished and it stays in the same state. With the *train* corpus, the system can try infinitely to find the right action whereas with the *test* corpus, to simulate the user giving up, it has only three tries. For the moment, we use a *Minimum time to goal* [9] reward function, but more custom functions will be used to take into account the apartment layout or to severely punish the system if it is too long to find the right action.

3) Evaluation and metrics: During the experiment, we save data allowing us to compute different metrics like the F1-score and the mean reward.

The first one is a classical classification metrics. Each decision making is seen as a classification of the state. We can then compute the precision and the recall to compute the F1-score. This reflects the ability of the system to choose the right decision based on the current state.

The mean reward is a more custom metrics which gives the mean reward obtained between two reward integrations for learning (the 10 steps explained above). Depending on the reward function used, this metric, with the F1-score, allow us to determine if the system fails often but with a small mistake (it lights the ceiling instead of the bedside lamp) or rarely but with a big one (it turns on the radio instead of closing the store).

C. Results

We ran the protocol explained above, and summarized the classification results in the confusion matrix shown in Figure 3. The fact that the matrix diagonal is well identified, as well as the good overall accuracy of about 70%, show that the learning process associate the right actions to the right state.

Nevertheless, you can see two erroneous behaviors. The first one is a confusion between actions 19 and 20, ‘turning on the kitchen ceiling light’ and ‘turning on the kitchen sink light’. This is an understandable confusion as some states are very ambiguous even for a human annotator. The second error appears on actions 11 and 12, ‘turn off all the lights of the kitchen’ and ‘turn off the kitchen ceiling light’, classified as ‘tell time in the kitchen’. We are still investigating to understand what is going on, but this seems to be cause by the lack of representative samples for both actions.

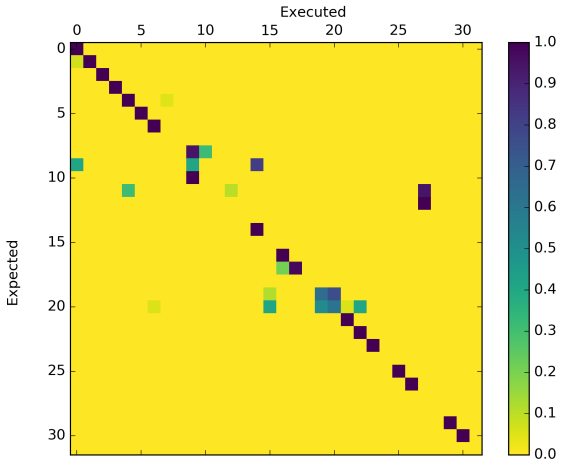

Figure 3. Confusion matrix for the decision-making system

IV. Discussion and Outlook

The experiment reported in this article shows that adapting a system to a user are possible even with a simple reinforcement approach as the Q-learning. However, a number of points have not been taken into account. The information generated by the home automation system is often used by decision systems. But in this work the decision is based on the information of location and activity that are the result of an inference. So, it is necessary to work on the transfer of negative reward that can either be due to a bad decision or a wrong inference. For this, we have to check whether the theoretical frameworks taking uncertainty into account can be applied to this double problem of uncertainty and reinforcement, as in the work of Hagras *et. al.* [16] in which a fuzzy reasoning based system adapts to the person while taking into account the uncertainty. The Partially Observable Markov Decision Process (POMDP) also seems particularly suited to this type of problem [17]. Finally, it is worth checking whether an MLN-based system [14] or a DNN-based [18] one can be adapted to include a strengthening mechanism.

Another important point not considered in the study is the identification of persons in the home. Indeed, a behavior is associated with a user and so rewards may be different. This problem can be included in the previous problem, characterizing a limited range of users that should be identified together with the rest of the environment.

Finally, we would like to validate our approach. A long-term adaptation requires a large amount of data that is extremely expensive to implement and raises verification problems (*e.g.* differentiate essential versus minor adjustments). Our strategy first is to test the system against radical adaptations like a person or habitat change that will provide information on system ease of 'installation' in a new environment. Moreover, we will test a long-term adaptation in less expensive environments, potentially less intrusive than the habitat, and with little identification problem, through smartphone applications and actions in an intelligent office.

References

[1] L. Cristoforetti, M. Ravanelli, M. Omologo, A. Sosi, A. Abad, M. Hagmueller, and P. Maragos, "The DIRHA simulated corpus," in *The 9th edition of the Language Resources and Evaluation Conference (LREC)*, Reykjavik, Iceland, 2014, pp. 2629–2634.

[2] M. Vacher, S. Caffiau, F. Portet, B. Meillon, C. Roux, E. Elias, B. Lecouteux, and P. Chahuara, "Evaluation of a context-aware voice interface for Ambient Assisted Living: qualitative user study vs. quantitative system evaluation," *ACM Transactions on Accessible Computing*, vol. 7, no. issue 2, pp. 5:1–5:36, May 2015.

[3] P. Moore, B. Hu, and M. Jackson, "Rule strategies for intelligent context-aware systems: The application of conditional relationships in decision-support," in *International Conference on Complex, Intelligent and Software Intensive Systems, CISIS 2011*, Seoul, Korea, 2011, pp. 9–16.

[4] M. J. Kofler, C. Reinisch, and W. Kastner, "A semantic representation of energy-related information in future smart homes," *Energy and Buildings*, vol. 47, pp. 169–179, apr 2012.

[5] C. Y. Leong, A. Ramli, and T. Perumal, "A rule-based framework for heterogeneous subsystems management in smart home environment," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 3, pp. 1208–1213, 2009.

[6] S.-H. Lee and S.-B. Cho, "Fusion of modular bayesian networks for context-aware decision making," in *Hybrid Artificial Intelligent Systems*, ser. Lecture Notes in Computer Science, E. Corchado, V. Snel, A. Abraham, M. Wozniak, M. Graa, and S.-B. Cho, Eds. Springer Berlin / Heidelberg, 2012, vol. 7208, pp. 375–384.

[7] L. S. Shafti, P. A. Haya, M. Garca-Herranz, and E. Prez, "Inferring eca-based rules for ambient intelligence using evolutionary feature extraction," *Journal of Ambient Intelligence and Smart Environments*, vol. 5, no. 6, pp. 563–587, 2013.

[8] M. C. Mozer, "The Neural Network House: An Environment hat Adapts to its Inhabitants," in *Proc. AAAI Spring Symp. Intelligent Environments*, 1998, pp. 110–114.

[9] M. E. Harmon and S. S. Harmon, "Reinforcement learning: A tutorial," 1996.

[10] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, Cambridge, UK, 1989.

[11] M. Chan, E. Campo, D. Estève, and J.-Y. Fourniols, "Smart homes — current features and future perspectives," *Maturitas*, vol. 64, no. 2, pp. 90–97, 2009.

[12] F. Portet, M. Vacher, C. Golanski, C. Roux, and B. Meillon, "Design and evaluation of a smart home voice interface for the elderly: acceptability and objection aspects," *Personal and Ubiquitous Computing*, vol. 17, pp. 127–144, 2013. [Online]. Available: http://dx.doi.org/10.1007/s00779-011-0470-5

[13] M. Gallissot, J. Caelen, F. Jambon, and B. Meillon, "Une plateforme usage pour l'intgration de linformatique ambiante dans l'habitat. l'appartement domus," *Technique et Science Informatiques (TSI)*, vol. 32, no. 5, pp. 547–574, 2013.

[14] P. Chahuara, F. Portet, and M. Vacher, "Making Context Aware Decision from Uncertain Information in a Smart Home: A Markov Logic Network Approach," in *Ambient Intelligence*, ser. Lecture Notes in Computer Science, vol. 8309. Dublin, Ireland: Springer, Dec. 2013, pp. 78–93. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00953262

[15] M. Vacher, B. Lecouteux, P. Chahuara, F. Portet, B. Meillon, and N. Bonnefond, "The Sweet-Home speech and multimodal corpus for home automation interaction," in *The 9th edition of the Language Resources and Evaluation Conference (LREC)*, Reykjavik, Iceland, 2014, pp. 4499–4506. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00953006

[16] H. Hagras, F. Doctor, A. Lopez, and V. Callaghan, "An incremental adaptive life long learning approach for type-2 fuzzy embedded agents in ambient intelligent environments," *IEEE Transactions on Fuzzy Systems*, vol. 15, no. 1, pp. 41–55, 2007.

[17] S. Zaidenberg and P. Reignier, "Reinforcement Learning of User Preferences for a Ubiquitous Personal Assistant," in *Advances in Reinforcement Learning*, A. Mellouk, Ed. Intech, 2011, pp. 59–80.

[18] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.