AUTOMATIC NAVIGATION AMONG MOBILE DTV SERVICES

Chengyuan Peng, Petri Vuorimaa
Telecommunications Software and Multimedia Laboratory,
Department of Compter Science and Engineering,
Helsinki University of Technology,
P.O. Box 5400, FIN-02015, Finland

Keywords: Intelligent user interface, button prediction, automatic navigation, reinforcement learning, exploration

Abstract: Limited number of input buttons on a mobile device, such as mobile phones and PDAs, restricts people's access to digital broadcast services. In this paper, we present a reinforcement learning approach to automatically navigating among services in mobile digital television systems. Our approach uses standard Q-learning algorithm as a theory basis to predict next button for the user by learning usage patterns from interaction experiences. We did the experiment using a modified algorithm in test system. The experimental results demonstrate that the performance is good and the method is feasible and appropriate in practice.

1 INTRODUCTION

Reception of TV programs in moving vehicles, such as in buses, trains or cars, is only practicable with digital television, not with the present analogue TV system. Digital television provides far more robust TV signals that are far less susceptible to interference than analogue TV, especially on the move.

Mobile reception, which is one of the advantages of the DVB-T (Digital Video Broadcasting-Terrestrial) broadcasting solution, is defined in DVB as being the reception of a DVB-T signal while in motion, using an omni directional antenna situated at no less than 1.5 meters above the ground level [Motivate, 2000] [EN 300 744, 2001]. Figure 1 shows one example, which is drawn from EyeTV [EyeTV 400, 2003].

Among all the parameters which characterize the service delivered to mobile receivers, the speed of the mobile, corresponding to a given Doppler frequency value, is considered as the main variable [Motivate, 2000].

For example, some measurements have confirmed that 8K mobile DVB-T reception of the 15 Mbit/s mode {8K, 16-QAM (quadrature amplitude modulation), code rate=1/2, Tg=224 ms (a guard interval of duration)} is possible in the entire UHF band with speeds in the range 130-235 km/h and a carrier-to-noise (C/N) ratio requirement of less than 18 dB [Pogrzeba, 1999] [Reimers, 1998]. Data rate for DVB-MHP (DVB-Multimedia Home Platform) applications provided by DVB-T is capable of 256 kbps [EN 300 744, 2001]. Therefore, there is a realistic possibility for high-speed mobile reception of both television and data services in cars,

Figure 1: Mobile reception of DTV.

Peng C. and Vuorimaa P. (2004).
AUTOMATIC NAVIGATION AMONG MOBILE DTV SERVICES.
In *Proceedings of the Sixth International Conference on Enterprise Information Systems*, pages 140-145
DOI: 10.5220/0002629501400145

Digital television will offer a large amount of digital broadcast data services as well as TV channels, such as Electronic Program Guide (EPG) and digital Teletext. People can access digital television services on the move using their mobile devices, such as mobile phones and PDAs.

Users are often frustrated in their efforts to access these services with limited number of input buttons on their mobile devices. Also some mobile devices, such as mobile digital television set in a car, demand less user input. Thus, designing a user interface with minimum or without user's interference is an advantage. It could save user time, efforts, and frustration.

Our goal is to find an intelligent solution to create "zero-input" for browsing services. In further detail, whenever a user switches his or her mobile digital television sets or is at any state of a TV service, the system would be able to recommend next button to be pressed and execute the function represented by this button automatically for the user instead of manual browsing if the user does not want to give his or her own input. And therefore, the number of buttons pressed in this way can be significantly reduced.

Here we clear some confusion: the paper is not the issue of helping novice users or compensating the original poor user interface, but of resolving the design by novel learning algorithm. We also argue that multiple users may not access the same device to interrupt the agent.

It is very difficult to predict real intentions of a user because there are no examples to guide agent's learning, and also most of the times the interactions made are stochastic. Interactions are dynamic and parallel with learning, and demand real-time reaction. Unexpected button-press recommendation is unacceptable.

The agent's learning in this problem heavily depends on user's interactions or experiences with the environment and the changes of the broadcast environment itself as well. Consequently, we use experience-based and reinforcement learning techniques (especially the standard Q-learning algorithm) in machine learning. In section 3, we will describe reinforcement learning technique used in this paper and our approaches.

2 DESIGN OF LEARNING AGENT

The mobile digital television will consist of estimated tens of TV channels and 800 digital Teletext pages [Peng, 2002]. Navigation agent in a mobile device is more personalized toward individual users' interests and dynamic user behavior [Lieberman, 2001]. Every time user presses a button on a mobile device, that's an expression of interest in the subject of the services.

The design goal of learning agent in this paper was to be able to learn and infer user's intensions and interests by tracking interactions (i.e., history information of user's behavior) between the user and the device over the long term and provide continuous, real-time display of recommendations for the user. Agent keeps any significant histories of interaction. Browsing history, after all, is a rich source of input about the user's interests [Lieberman, 2001].

Further goal on the learning agent is concerned with tracking and understanding users' patterns of services browsing. In this paper, a reward function in Q-learning algorithm is used to match the behavior of the user in current situation with the past behaviors whose browsing pattern fits most closely, and return its predictions.

The agent performs reconnaissance in tracking user-browsing history to recommend new actions. This concept is not new in user interface design [Lieberman, 2001]. Given enough time, the agent becomes pretty good at predicting which button the user would be likely to choose next.

This paper presents another concept: What if there is no or few history information or services are changing? How the agent deals with this kind of situation? How to deal with the recommendations that user might not be interested in? Users might have many interests and changes over time. Also, users have a rich browsing history over potentially many months that can be exploited to better understand their true interests. Agent finds functions on a service of interest that the user might overlook because of the complexity of the service.

The agent designed runs simultaneously with mobile digital television services. The agent automates interactions between user and mobile device. Over time, it learns a pattern of the user's interests by recording and analyzing the user's browsing activity in real time, and provides a continuous stream of recommendations.

When the user is actively browsing and attention is focused on the current service or function, the user need not to pay any attention to the agent. However, if the user is unsure where to go next, or dissatisfied with the current offerings, he or she can glance over to the recommendation window to look at agent's suggestions.

3 REINFORCEMENT LEARNING

In this section, we will give a short description of reinforcement learning, which is adopted in this paper. Reinforcement learning serves as a computational approach to automating goal-directed learning and decision-making for an agent [Sutton, 1998]. Its distinguished feature from other approaches is learning from direct interaction with the environment. It has been used by a number of researchers as a practical computational tool for constructing autonomous systems that improve themselves with experiences [Papudesi, 2003].

Reinforcement learning is a kind of unsupervised learning. Reinforcement learning, especially Q-learning agent is learning to act, i.e., learning how to map agent's states to its actions in order to maximize a numerical reward signal rather than giving training information in the form of instructions like in supervised learning [Sutton, 1998]. A reward indicates the instinct of an agent and it is usually programmed by a designer in advance. An agent learns to associate a value (i.e. value function) with the execution of actions in different states. A value function specifies what is good in the long run (i.e. total expected reward).

Reinforcement learning is also a kind of trial-and-error learning, in which the central problem is balancing exploration and exploitation [Dearden, 1998]. That is, in order to obtain a lot of reward, an agent has to exploit what it already knows in order to obtain reward, but it has also to explore in order to make better action selections in the future [Sutton, 1998]. Both model-based and model-free methods for balancing exploration and exploitation have been introduced in [Dearden, 1998] [Wiering, 1998].

Our focus in this paper is however not on the trade-off between exploration and exploitation, but on exploration, in which we show how the system recommends actions and how unexpected actions (or errors) can be reduced by an efficient action selection strategy. By reasonable exploration, we can shorten agent's learning time and solve the problems in the concept mentioned in section 2.

We also study the problems of collecting useful experiences and environment changes via exploration in stochastic environment using reward functions in Q-learning algorithm.

4 EXPLORATION VIA Q-LEARNING ALGORITHM

This section gives a short introduction to Q-learning algorithm and presents our approach based on model-free method. Model-Free approaches only require sufficient exploration of the environment to solve the problem. This is very useful when the number of states is very large. An outline of Q-learning algorithm using ε-greedy action selection method follows [Sutton, 1998]:

```
(1) Initialize value function
    Q(s,a);
(2) Choose an action a from the
    state s using a policy derived
    from Q using ε-greedy method;
(3) Take the action a, observe reward
    R and next state s';
(4) Calculate value function Q by
    equation:
$Q(s,a) = Q(s,a) + \alpha\{R + \gamma \max_{a'} Q(s',a')$
        $- Q(s,a)\}$
(5) s = s';
(6) Go to step (2);

where
  Q  : a table of Q-values
  a  : previous action
  s  : previous state
  s´ : new state that resulted from
       the previous action
  a´ : action that will produce the
       maximum Q value
  α  : learning rate (0 < α ≤1)
  R  : reward
  γ  : discount rate ( 0≤ γ ≤1)
```

The ε-greedy action selection rule in above algorithm uses a small probability ε to select an action with the highest *Q* value, and to select a random action otherwise [Wiering, 1998]. In order for an agent to select an action, it needs to explore it. Random actions are necessary for exploration; however, actions selected by random process are sometimes unexpected in practical system where the problems are. We change ε-greedy rule at step (2) in above Q-learning algorithm as follows:

given a random number *rand* in each loop in above algorithm, action a is selected as following:

$$a = \begin{cases} a_m & \text{if} \quad rand < \varepsilon \\ a_p & \text{if} \quad rand \geq \varepsilon \end{cases}$$

where,

$$a_m = \begin{cases} a_x & \text{action with highest Q-value} \\ a_r & \text{randomly selected action} \end{cases}$$

$$a_p = \begin{cases} a_u & \text{human control} \\ a_t & s_t \in \text{set of (s,a) pairs tried} \\ a_s & s_t \notin \text{set of (s,a) pairs tried} \end{cases}$$

where,

a_t indicates experienced action

a_s indicates random action

In order to decide action a_m, we select an action with maximum Q-value in the Q-table; in case of no action with maximum Q-value in the Q-table, we choose an randomly selected action a_r, otherwise.

Action a_p is chosen from one of the followings: user controlled action a_u; reactive controlled action a_t which is used as learning intentions-usage patterns; or approaching the unknown action in a_s. S_t indicates the current state and the set of (*s*, *a*) pairs mean the experienced states and actions, i.e., the set:

$$\{ (s_0, a_0), (s_1, t_1), \dots, (s_{t-2}, a_{t-2}), (s_{t-1}, a_{t-1}) \} \quad (1)$$

The exploration strategy in this way would lead to free button predictions. Through the exploration, the system is going towards predicted states and actions.

There is still a question of whether or not this approach would scale. For example, user wishes to deviate or start operating with new behaviors. This situation can be handled well if we use multi-sensed description of the agent's world state and avoid very impoverished state.

5 EXPERIMENTS

We assume that Markov property does hold in this problem. Markov property means that transition probability only depends on the last state and action, not on the whole history of states and actions [Sutton, 1998].

The overall learning procedure is that every time the agent is in a state, it then tries out an action using the action selection algorithm designed in last section; after the action is performed, the system evaluates how successful it was by using a reward function defined in formula (2).

We also need to define a set of states and actions together with the reward function and other parameters, which are key elements of reinforcement learning, as in the rest of following paragraphs in this section:

We simplify functions represented by a series of buttons on a remote control in a mobile device as individual letters, which are shown in Table 1.

A state is a current summary of what can be sensed by the agent from the environment. In this experiment, we represent a state as two parts (there should have more bits in real system): one as current TV channel number and another as last action taken. For example, state *6 N* indicates the current state of the environment is in channel 6 and within EPG service. Also the actions of the agent taken determine the next state.

Table 1. A simplified representation of a remote control buttons.

Code	Function	Code	Function
0-9	Channels	Z	Red button
T	Teletext	X	Green button
N	Navigator/ EPG	C	Yellow button
I	Interactive program	V	Blue button
U	Up arrow	B	Back/Exit button
D	Down arrow	Q	Enter button
L	Left arrow	R	Right arrow

Navigation in this problem means taking actions. We differentiate between interference actions (user pressed) and automatically learned actions. Actions are presented as buttons, which include both channel keys (0-9) and non-channel keys (the rest of keys in Table 1).

Q-Learning values are built on a reward scheme. We need to design a reward algorithm that will motivate the agent to perform a goal-oriented behavior. A reward function is quite related to usability heuristics in this problem. For each action, a reward is determined according to the current situation and action taken. For example, it receives a reward of maximum value if the action is the some, which the user pressed.

Reward function depends on the tasks. In this paper, we give the reward function as follows:

$$R(s,a)=\sum_{k=1}^{N}\omega_k\varphi_k(s,a) \qquad (2)$$

Where N indicates the number of tried (state, action) pairs (i.e. set (1)), ω_k is a constant and depends on action (e.g., user controlled action a_u has the biggest reward 0.8, random action a_s has a reward 0.6, random action a_p has a reward 0.3, etc.), and $\varphi_k(s, a)$ is either 1 or 0 and it depends on the state and action.

The system also keeps a so called "footage" of interactions (i.e., the history of interaction) that user ever made (cf. Table 2 and Table 3). These experiences are used for giving reward evaluation and more importantly, they are used for learning usage patterns of the user. During interaction, the system needs to update history file (cf. Table 2) online. For example, if there is no channel button in the history file then the system chooses random action and the reward would be 0.1.

In order to learn which actions are useful, the actual Q-values should be calculated and the entire Q-value table will be saved and used later on. Basically, we just need to calculate the equation and give the parameters values used in this experiment, i.e., explore-rate ε (0.1), discount-rate γ (0.65), and learning-rate α (0.3).

6 RESULTS

The system keeps a history file of state-action pairs interacted in the past. Table 2 and Table 3 list a part of interaction history, which includes both user pressed and agent learned actions using the exploration algorithm designed in this paper.

Table 2: A sample of the history of interaction.

9	
1	
2	
3	
4	
5	
6	
9	nqddddddbdqdddddb
1	**tddddduuuuqd**
2	
3	
4	
5	
6	
7	
8	
9	ndqzdqzddqxqxdqxddqb
9	idrqbbdddqdbb
9	tdddduuuuqqrrrrrlllldddd
9	tqddddddrddddrdddrddddddrdddddxdqqq ddddrdd

Table 2 lists some of past interactions. For example, text stream **1tddddduuuuqd** at the ninth line of Table 2 means that TV channel 1 and its digital Teletext service were once selected, then five down and four up buttons were pressed, finally one enter and up buttons were pressed.

Table 3 lists some actions learned by the agent from the exploration algorithm using past actions. For example, text stream **1nddddddddz** at the fifth line of Table 3 means that TV channel one was selected and agent recommended buttons are choosing Navigator service, then selecting eight down buttons, finally selecting red color button on a remote control.

The results of this evaluation were given by averaging of ten test runs. The error action rate is 3%, which is less than the expected error rate 5%. The results were based on the real services used in our previous experiments [Peng, 2002] in which we used twelve TV channels and three services, i.e. digital Teletext, Navigator or EPG, and interactive services.

Table 3: A sample of interactions and learned actions.

9	ndqzdqzdqzdqzdqzdqzdqzdqz
2	
1	tdddddd
8	
1	**nddddddddz**
9	tqzdqz
9	tqzdqzdrbqzdqzdqzd
6	x
7	ubz
2	
4	
9	tqzdqzdqzdqzdqzdqztqzdqzdqzdqz
1	tddddddddddddqu
5	
2	
3	nr
1	tdddddddl
1	tddddddddqudddddddddd
9	idq

The two averaged exploration rate curves are shown in Figure 2 in which the horizontal axis shows the steps of test running and the vertical one shows the average reward.

The ε-greedy exploration rate curve in Figure 2 used the standard Q-learning algorithm, while experiment exploration rate curve in Figure 2 used the algorithm proposed in this paper. The curves show that the learning rate of proposed exploration algorithm converges faster than ε-greedy algorithm in this action selection problem. That is the experiment algorithm can provide meaningful and correct actions earlier than the standard ε-greedy algorithm.

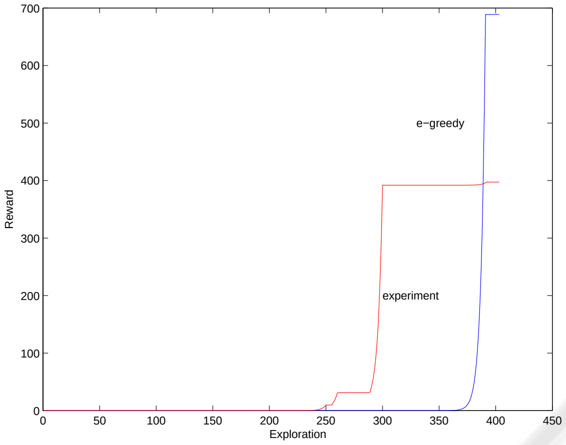

Figure 2: Exploration rate curves.

7 CONCLUSIONS

Through this paper, we highlighted an exploration algorithm used for automatic button selections in a mobile digital television system. The results have shown that this algorithm is feasible and could be a promising solution to this problem. It can be of practical values in such a system because the speed needed to process is in real-time and very little programming is required.

Also, if the environment changes, e.g. remote control manufacturer changes, it doesn't need to be reprogrammed as mentioned earlier. Even if the services are altered, it does not need to redesign the learning algorithm. If the learning algorithm is properly designed, the agent is guaranteed to be able to find the most efficient policy. These points have advantages over supervised learning approaches.

The reward function is very important, however choosing the reward function is hard and may not be obvious for any given task. Defining a reward depends on the task and prior knowledge built in the system.

The learning rate constant α in the Q-learning equation determines the rate of convergence on the optimal algorithm. It's important to strike the right balance between speedy learning and giving the Q-value history more weight.

Finally, Q-Learning algorithm would really perform best if the agent could keep track of more bits in a state. However, this would increase the number of states that the Q-table would need to keep track of. This will demand more system memory, and the memory available in a mobile device is an important constraint of mobile systems. Another problem in the algorithm is how to handle errors when the recommended action is not the expected one.

REFERENCES

Dearden, R., Friedman N., and Russel, S., 1998. Bayesian Q-learning. *American Association for Artificial Intelligence (AAAI)*, pp. 761-768.

EyeTV 400, 2003. Digital video recorder: for digital terrestrial television (DVB-T), *data sheet,* available at http://www.elgato.com.

Lieberman, H., Fry, C., and Weitzman, L., 2001. Exploring the Web with Reconnaissance Agents. *Communications of the ACM*, 44(8), pp. 70-75.

Motivate, 2000. Using DVB-T standard to deliver broadcast Services to mobile receivers. *Report TM2310*, available at http://www.dvb.org/.

Papudesi, V. N., and Huber, M., 2003. Learning from Reinforcement and Advice Using Composite Reward Functions. *In Proceedings of the 16th International FLAIRS Conference,* St. Augustine, FL, pp. 361-365.

Peng, C., 2002. *Digital Television Applications*, Doctoral Dissertation, Helsinki University of Technology Press, Espoo.

Pogrzeba, P., Burow, R., Faria, G., and Oliphant, A., 1999. Lab and field tests of mobile applications of DVB-T. *Montreux Symposium '99 Records*, pp. 649-656.

Sutton, R. S., and Barto, A.G., 1998. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge.

Wiering, M., and Schmidhuber, J., 1998. Efficient Model-based Exploration. In *Proceedings of the 5th International Conference on Simulation of Adaptive Behavior: from Animals to Animats 5*, Switzerland, pp. 223-228.

EN 300 744, 2001. Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television. Specification, *European Telecommunications Standards Institute.*

Reimers, U., 1998. Digital Video Broadcasting, *IEEE communications Magazine*, pp. 104-110.