8-2010

Learning personal agents with adaptive player modeling in virtual worlds

Yilin KANG

Ah-hwee TAN
Singapore Management University, ahtan@smu.edu.sg

2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology

Learning Personal Agents with Adaptive Player Modeling in Virtual Worlds

Yilin Kang
School of Computer Engineering
Nanyang Technological University
Nanyang Avenue, Singapore, 639798
Email: kang0028@ntu.edu.sg

Ah-Hwee Tan
School of Computer Engineering
Nanyang Technological University
Nanyang Avenue, Singapore, 639798
Email: asahtan@ntu.edu.sg

***Abstract*—There has been growing interest in creating intelligent agents in virtual worlds that do not follow fixed scripts predefined by the developers, but react accordingly based on actions performed by human players during their interaction. In order to achieve this objective, previous approaches have attempted to model the environment and the user's context directly. However, a critical component for enabling personalized virtual world experience is missing, namely the capability to adapt over time to the habits and eccentricity of a particular player. To address the above issue, this paper presents a cognitive agent with learning player model capability for personalized recommendations. Specifically, a self-organizing neural model, named FALCON (Fusion Architecture for Learning and Cognition), is deployed, which enables an autonomous agent to adapt and function during the players' interaction. We have developed personal agents with adaptive player models as tour guides in a virtual world environment. Our experimental results show that we are able to learn user models that evolve and adapt with players in real time. Furthermore, the virtual tour guides with player models outperform those without adaptive player modeling in terms of recommendation accuracy.**

***Keywords*-player modeling; virtual world; learning agents**

I. Introduction

Virtual world has become a popular platform used in a variety of contexts, including teaching in classrooms, informal learning, distance learning, business, and e-commerce [1]. Studies in South Korea have recently shown that users prefer virtual world to television [2]. Gartner even predicted that 80 percent of the Internet users will be actively participating in non-gaming virtual world by the end of 2011.

A typical research problem in virtual world is to incorporate intelligent learning agents to improve its interactivity and playability. Indeed, learning in a virtual world, just like in the real world, poses many challenges not addressed by traditional machine learning algorithms. In particular, learning in virtual world is typically unsupervised, without an explicit teacher to guide the agent in learning. Furthermore, it requires an interplay of a myriad of learning paradigms. However, most virtual worlds tend to constrain agents' actions to a very coarse level, dictated by hard coded rules [3], [4], [5]. In the recent few years, there has been growing interest in creating intelligent agents in virtual worlds that do not follow fixed scripts predefined by the developers, but react accordingly based on actions performed by the players during their interaction. In order to achieve this objective, previous approaches have attempted to model the dynamic environments and user's immediate context [6], [7], [9], [10]. However, they typically ignored a significant component of making the virtual world experience much more intense and personalized for players, namely the capability to adapt over time to the habits as well as eccentricity of a particular player.

To address the inadequacy of previous work, in this paper, we present a self-organizing neural model, named TD-FALCON (Temporal Difference - Fusion Architecture for Learning and Cognition) [11], for creating intelligent learning agents in virtual worlds. By incorporating TD-FALCON, an agent is able to learn from sensory and evaluative feedback signals received from the virtual environment. In this way, the agent needs neither an explicit teacher nor a perfect model to learn from. Performing reinforcement learning in real time, it is also able to adapt itself to the variations in the virtual environment and changes in the user behavior patterns.

More importantly, to enable more dynamic and personalized experience, we propose to incorporate adaptive user modeling with our learning personal agents. Specifically, a two-channel fusion ART [12], [13], is used for learning player models during the players' interaction. By formulating cognitive codes associating agent's recommendation to user feedback, fusion ART learns direct player models of the users' likes and dislikes.

We have developed learning personal agents using TD-FALCON with adaptive user modeling in a 3-D virtual world called Youth Olympic Village (YOV) Co-Space. In this application, the learning personal agents are designed to befriend human users and proactively offer them personalized functions and services in the virtual environment [14]. Our experimental results show that we are able to learn player models that evolve and adapt with player during run time. Furthermore, the virtual tour guides with adaptive player models outperform those without player models. Although the idea of player modeling has been proposed before, most of these works do not work in conjunction with a reinforcement learning agent. Furthermore, even few have

978-0-7695-4191-4/10 $26.00 © 2010 IEEE
DOI 10.1109/WI-IAT.2010.201

attempted to provide any quantitative assessment.

The rest of this paper is organized as follows. In section II, we give a brief review of the related work. In section III, we describe the case study on the YOV Co-Space. In section IV, we provide the algorithm of fusion ART. In section V, we present the procedure and methods of using TD-FALCON for learning personal agents in real time. We provide the methods for using two-channel fusion ART for adaptive user modeling in section VI. In section VII, we describe how we integrate player models with personal agents. In section VIII, we report the results of the empirical experiments. The final section concludes and highlights future work.

II. Related Work

Player modeling in computer games and virtual worlds have recently attracted the interest in both the academia and the computer games industry.

In the domain of interactive storytelling, many researchers employed player modeling to learn the player's preferred style of play, and then uses that model to select the content of an interactive story [15], [16]. Similarly, Charles and McNeill proposed to use player modeling to facilitate player-centered game design, in the form of providing a more appropriate level of challenge, smoothening the learning curve, and enhancing the gameplay experience [17].

Comparing with most other games, poker games can be seen as a pioneer of player modeling as it already has shown a great impact on success. Lockett and Miikkulainen made use of coarse approximations to game-theoretic player representations to improve the performance of software players in Limit Texas Hold 'Em poker [18]. Also, Baker et.al. investigated the impact of Bayesian opponent modeling upon the evolution of a player for a simplified poker game [19]. Unlike virtual world, player models are relatively simple in poker games which are divided into three or four categories. In contrast, virtual world presents a much more complex environment. As such, simple strategies may not work.

Houlette defined the concept of player models and discussed the strategies for design, implementation and integration for general games such as First Person Shooting [20]. However, he didn't present any experimental result to support his theory. Other researchers, such as Hladky and Bulitko, successfully predicted opponent positions by evaluating hidden semi-Markov models and particle filters in First Person Shooting games [21]. Heijden et.al. discussed an approach to organizing units by learning the effectiveness of a formation and directly applying learned formations according to the classification of the opponent player in Real-Time Strategy games [22]. However, all these work employ offline learning which can not deal with the changes in the players' behaviour during run-time.

All the work described above have involved the use of player models in games with specific motivations. However, to the best of our knowledge, there has been very few work, if any, integrating player models with reinforcement learning agents in virtual world that can adapt to the habits and eccentric of a particular player dynamically. Our work is motivated by such a consideration.

III. A Case Study On YOV Co-Space

Co-Spaces are virtual worlds developed for mirroring a real physical world in terms of look-and-feel, functions and services. The objective of the Youth Olympic Village (YOV) Co-Space is to introduce the YOV and the hosting country to visitors around the world in an interactive and playable manner. To achieve this objective, we are in the process of developing and populating human-like cognitive agents in the form of autonomous avatars that roam in the landscape of YOV Co-Space. The agents are designed to be aware of its surrounding and can interact with users through their human avatars. With the autonomous avatars befriending and providing personalized context-aware services to human avatars, we aim to make the content and services readily available to the users.

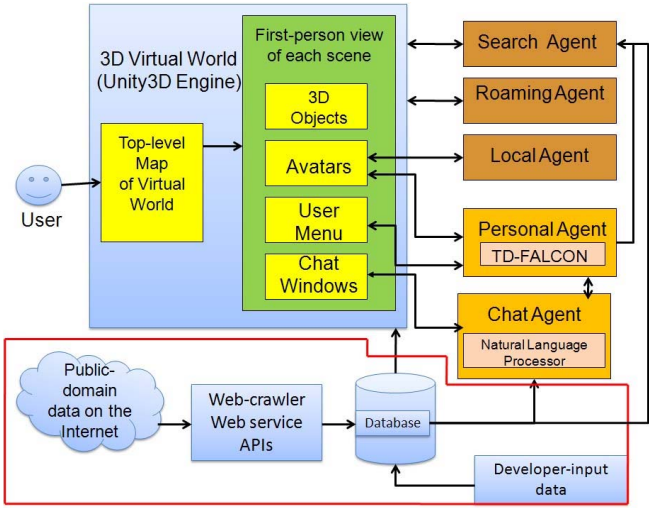

Figure 1. The architecture of Co-Space.

Figure 1 shows the architecture of Co-Space. As illustrated in this framework, the TD-FALCON based personal agent works in conjunction with the search agent in recommending functions and services to the users. Specifically, the personal agent determines the appropriate type of services to recommend whereas the search agent retrieves the specific services based on the environment situations as well as the users' context parameters. Figure 2 provides a screenshot of the virtual world, showing a personal agent serving the user.

IV. Fusion ART

Fusion ART [23] is based on multi-channel Adaptive Resonance Associative Map (multichannel ARAM) [24], an extension of predictive Adaptive Resonance Theory (ART) networks. Whereas predictive neural network models, such

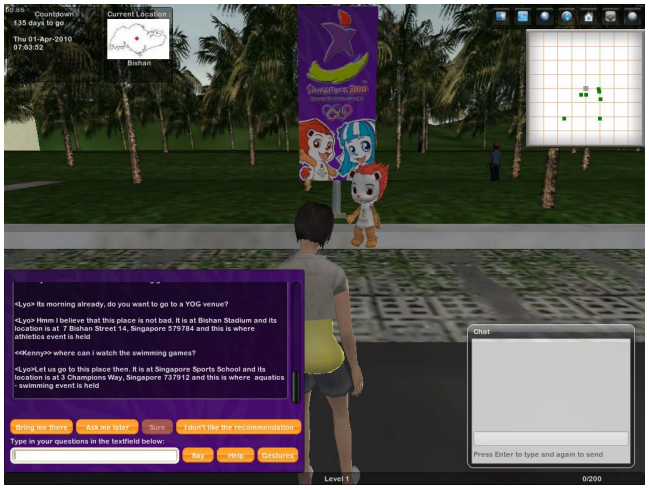

Figure 2. A screenshot of TD-FALCON based personal agent in Co-Space.

as ARTMAP network [25] and ARAM [8], learn multi-dimensional mappings between the input and output patterns, fusion ART formulates cognitive codes associating multi-modal patterns across multiple input channels.

The generic network dynamics of fusion ART, based on fuzzy ART operations [25], is described as follows.

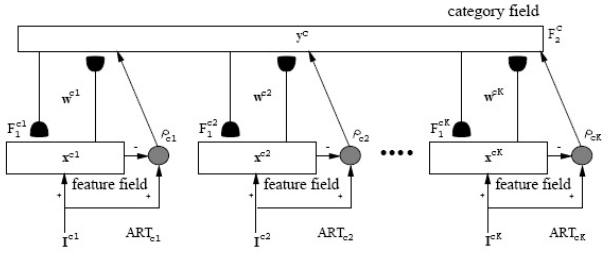

Figure 3. The architecture of fusion ART.

Input vectors: Let $\mathbf{S} = (s_1, s_2, \ldots, s_n)$ denote the state vector, where $s_i \in [0, 1]$ indicates the value of sensory input i. Let $\mathbf{A} = (a_1, a_2, \ldots, a_m)$ denote the action vector, where $a_i \in [0, 1]$ indicates the preference of a possible action i. Let $\mathbf{R} = (r, \overline{r})$ denote the reward vector, where $r \in [0, 1]$ is the reward signal value and $\overline{r}$ is given by $\overline{r} = 1 - r$. The whole input vectors are with complement coding.

Activity vectors: Let $\mathbf{x}^{ck}$ denote the F_1^{ck} activity vector for $k = 1, \ldots, 3$. Let $\mathbf{y}$ denote the F_2 activity vector.

Weight vectors: Let $\mathbf{w}_j^{ck}$ denote the weight vector associated with the jth node in F_2 for learning the input patterns in F_1^{ck} for $k = 1, \ldots, 3$. Initially, F_2 contains only one *uncommitted* node. An *uncommitted* node is one which has not been used to encode any pattern and its weight vector contains all 1s.

Parameters: The FALCON's dynamics is determined by choice parameters $\alpha^{ck} \geq 0$, learning rate parameters $\beta^{ck} \in [0, 1]$, contribution parameters $\gamma^{ck} \in [0, 1]$ and vigilance parameters $\rho^{ck} \in [0, 1]$ for $k = 1, \ldots, 3$.

The FALCON pattern processing cycle comprises of five key stages, namely code activation, code competition, activity readout, template matching, and template learning, as described below.

Code activation: A bottom-up propagation process first takes place in which the activities of the category nodes in the F_2 field are computed. Specifically, given the activity vectors $\mathbf{x}^{c1}, \mathbf{x}^{c2}, \mathbf{x}^{c3}$, for each F_2 node j, the choice function T_j is computed as follows:

$$T_j = \sum_{k=1}^{3} \gamma^{ck} \frac{|\mathbf{x}^{ck} \wedge \mathbf{w}_j^{ck}|}{\alpha^{ck} + |\mathbf{w}_j^{ck}|}, \tag{1}$$

where the fuzzy AND operation $\wedge$ is defined by $(\mathbf{p} \wedge \mathbf{q})_i \equiv min(p_i, q_i)$, and the norm $|.|$ is defined by $|\mathbf{p}| \equiv \sum_i p_i$ for vectors $\mathbf{p}$ and $\mathbf{q}$.

Code competition: A code competition process follows under which the F_2 node with the highest choice function value is identified. The system is said to make a choice when at most one F_2 node can become active after the code competition process. The winner is indexed at J where $T_J = \max\{T_J: \text{for all } F_2 \text{ node } j\}$.

When a category choice is made at node J, $y_J = 1$ and $y_j = 0$ for all $j \neq J$. This indicates a winner-take-all strategy.

Activity readout: The chosen F_2 node J performs a readout of its weight vectors into the input fields F_1^{ck} such that

$$x^{ck(new)} = x^{ck(old)} \wedge w_J^{ck}. \tag{2}$$

The resultant F_1^{ck} activity vectors are thus the fuzzy AND of their original values and their corresponding weight vectors.

Template matching: Before the node J can be used for learning, a template matching process checks that the weight templates of node J are sufficiently close to their respective input patterns. Specifically, resonance occurs if for each channel k, the *match function* m_J^{ck} of the chosen node J meets its vigilance criterion ρ^{ck}:

$$m_J^{ck} = \frac{|\mathbf{x}^{ck} \wedge \mathbf{w}_J^{ck}|}{|\mathbf{x}^{ck}|} \geq \rho^{ck}. \tag{3}$$

If any of the vigilance constraints is violated, mismatch reset occurs in which the value of the choice function T_J is set to 0 for the duration of the input presentation. The search process then selects another F_2 node J until a resonance is achieved.

Template learning: Once a resonance occurs, for each channel ck, the weight vector $\mathbf{w}_J^{ck}$ is modified by the following learning rule:

$$\mathbf{w}_J^{ck(\text{new})} = (1 - \beta^{ck})\mathbf{w}_J^{ck(\text{old})} + \beta^{ck}(\mathbf{x}^{ck} \wedge \mathbf{w}_J^{ck(\text{old})}). \tag{4}$$

When an uncommitted node is selected for learning, it becomes *committed* and a new uncommitted node is added to the F_2 field. Fusion ART thus expands its network architecture dynamically in response to the input patterns.

V. Learning Personal Agent

Our personal agent is based on TD-FALCON [11] that employs a three-channel fusion ART (Figure 4) and incorporates Temporal Difference (TD) methods to estimate and learn value functions of action-state pairs $Q(s, a)$ that indicates the goodness for a learning system to take a certain action a in a given state s. Such value functions are then used in the action selection mechanism, also known as the *policy*, to select an action with the maximal payoff. TD-FALCON algorithm selects an action with the maximal Q-value in a state s by enumerating and evaluating each available action a by presenting the corresponding state and action vectors **S** and **A** to FALCON.

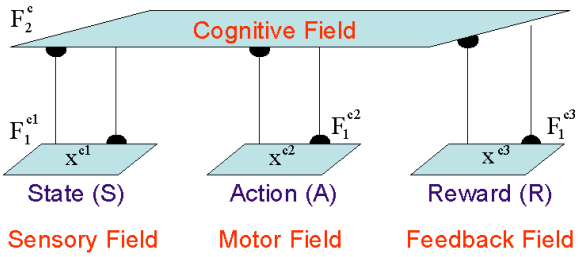

Figure 4. TD-FALCON architecture.

TD-FALCON has the properties such as self-adaptation, generalization, TD learning, fast and stable real-time learning. These make TD-FALCON a suitable candidate for building learning agents in virtual world. By incorporating TD-FALCON, an agent will be able to learn from sensory and evaluative feedback signals received from the virtual environment without involving human supervision and intervention. In this way, the agent needs neither an explicit teacher nor a perfect model to learn from. Performing reinforcement learning in real time, it is also able to adapt itself to the variations in the virtual environment and changes in the user behavior patterns. Furthermore, by incorporating temporal difference learning, TD-FALCON agents can overcome the issues, such as the absence of immediate reward (or penalty) signals in virtual world by estimating the credit of an action based on what it will lead to eventually.

The general sense-act-learn algorithm of TD-FALCON is summarized in Table I.

A. State Representation

The input state of the personal agent consists of five sets of attributes, namely time, location, player's request, player's interest and player's current activity. A summary of these attributes together with the possible values is given in Table II. All attributes adopt a binary encoding scheme in the state vector. Although we include user request as part of the state space, the agents are intended to work without explicit user request. In other words, they are supposed to proactively make recommendation based on user's interest, context and situation.

B. Action Space Representation

The personal agent is designed to determine the most appropriate service for recommendation to its user. The action field thus consists of recommendations of five types of services, namely hotel, dinning, YOV, place of interest and shopping. Using a binary encoding scheme, the action vector is represented as $\mathbf{a} = (a_1, a_2, \ldots, a_5)$, where $a_j = 1$ and $a_k = 0$ for all $k \neq j$ indicate that action j is recommended.

C. Computing Reward Signals

The reward function $r(t)$ is defined as a synthesis of the player's feedback and mood as $r(t) = (f(t) + m(t))/2$, where f is an explicit reward based on the user feedback through the dialog menu and f is an implicit reward value suggested by the gestures of the user. The dialog menu enables a player to choose among "wonderful", "thanks", "good", "fair", and "leave me alone", which correspond to a reward value of 1, 0.75, 0.25 and 0 respectively. Meanwhile, we obtain a player's mood by observing the gestures of the player. Gestures, such as waving, dancing, and jumping, indicate a positive reward of 1 since they indicate satisfaction with the service. In contrast, a reward of 0 is given to gestures, such as angry and walk away, which suggest unhappiness.

VI. Learning Player Models

We adopt a two-channel fusion ART (Figure 5) for learning player models. Essentially, the model performs supervised learning through the pairing of the input patterns and teaching signals received from the virtual environment without involving human supervision and intervention. If an initial user profile is available, the model first creates cognitive nodes that associate the attributes specified in the player's profile with positive reward signals. During play time, the player model learns the user's likes and dislikes by formulating cognitive codes associating the attributes of the agent's recommendations to users' feedbacks. For example, if the agent recommends a Japanese restaurant "Dozo" and the user likes it, the attribute corresponding to Japanese in the player's profile and the reward attribute are set to '1' while other attributes are set to '0'. Fusion ART then encodes these patterns into a cognitive code. Performing supervised learning in real time, it is also able to adapt itself to the changes in the user preference patterns.

During the learning mode, given the input space S and output space A, the system first performs code activation and code competition (as described in Section IV) to select a winner J. Before node J can be used for learning, a template matching process checks that the weight templates of node are sufficiently close to their respective input patterns. If the chosen node J violates its vigilance criterion, the search process then selects another F_2 node J under the revised vigilance criterion until a resonance is achieved. Once a node

Table I
GENERIC FLOW OF THE TD-FALCON ALGORITHM

1. Initialize the FALCON network
2. Given the current state s, for each available action a in the action set $\mathcal{A}$, predict the value of the action Q(s,a) by presenting the corresponding state and action vectors $\mathbf{S}$ and $\mathbf{A}$ to FALCON.
3. Based on the value functions computed, select an action a from $\mathcal{A}$ following an action selection policy.
4. Perform the action a, observe the next state s', and receive a reward r (if any) from the environment.
5. Estimate the value function $Q(s,a)$ following a temporal difference formula given by $\Delta Q(s,a) = \alpha TD_{err}$
6. Present the corresponding state, action, and reward (Q-value) vector, namely $\mathbf{S}$, $\mathbf{A}$ and $\mathbf{R}$, to FALCON for learning.
7. Update the current state by $s = s'$.
8. Repeat from Step2 until s is a terminal state.

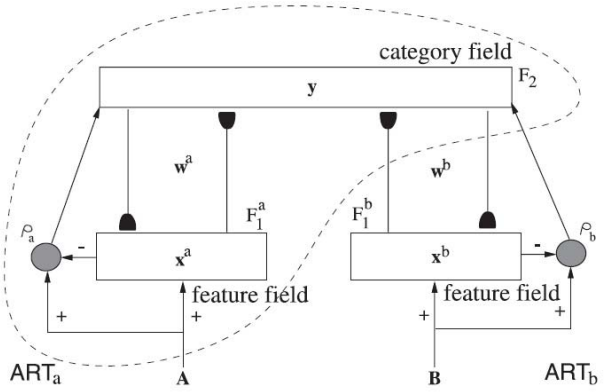

Figure 5. A two channel fusion ART architecture for supervised learning of user models.

Table II
ATTRIBUTES AND POSSIBLE VALUES IN THE STATE SPACE FOR PERSONAL AGENT

Attributes	Values
Time	Morning, lunch Time, Afternoon Dinner Time, Evening, Bed Time
Player's Location	Central, North, South, East West
Player's Request	Hotel, Dining, YOV Venue, Place of Interest(POI), Shopping, Unknown
Player's Interest	Dining, YOV, POI, Shopping unknown
Player's Activity	Hotel, Dining, YOV, POI Shopping, Unknown

J is selected for firing, for each channel k, the weight vector is modified by the template learning rule.

In a predicting mode, fusion ART receives input patterns from one or more input fields and predicts the patterns in the remaining fields. The predicting process follows three key steps, namely, code activation, code competition, and activity readout. In case of more than one winners during the code competition process, these nodes perform a combined activity readout. Furthermore, the players' general interest could be inferred based on the assumption that the greater the number of player's preferences in one general category, the more the player is interested in this category. In this way, player's interest in the state space of personal agent (shown in Table II) will be updated over run-time.

A. Input Space Representation

The input space of the player model consists of six sets of attributes, namely Preferred Hotel Class, Favorite Type of Food, Consumption Level, Favorite Type of Sports, Favorite Type of POI and Favorite Type of Shopping. A summary of these attributes together with the possible values is given in Table III. All attributes adopt a binary encoding scheme in the state vector.

Table III
ATTRIBUTES AND POSSIBLE VALUES IN THE STATE SPACE OF PLAYER MODEL

Attributes	Values
Preferred Hotel Class	5 star, 4 star, 3 star and below
Favorite Type of Food	Chinese, Western, International, Taiwanese, Seafood, Indian, Mexican, Korean, Thai, Halal, Italian, Singapore, Japanese, French, Steamboat, Nonya and Peranakan, Vegetarian, Vietnamese
Consumption Level	Luxury, Medium, Budget
Favorite Type of Sports	Aquatics, Archery, Athletics, Badminton, Basketball, Boxing, Canoe-Kayak, Cycling, Fencing, Football, Gymnastics, Modern Pentathlon, Rowing, Sailing, Shooting, Table tennis, Triathlon, Volleyball, Weightlifting, Judo, Hockey, Equestrian, Handball, Taekwondo, Wrestling, Tennis
Favorite Type of POI	Church, Memorial, Mosque, Palace, Temple, Zoo
Favorite Type of Shopping	Children and Maternity, Electrical and Electronic, IT and Telecommunication, Art and Craft, Jewellery and Watch, Fashion and Accessory, Entertainment and Music, Leather good and Footwear, Health and Pharmacy and Toilet

B. Output Space Representation

The player model is defined to simulate players with different background, personalities and interests. As a start, our action field only consists of two attributes: Like and dislike, represented using the binary encoding scheme.

VII. INTEGRATING PLAYER MODEL WITH PERSONAL AGENT

The architecture of the overall recommendation agent, incorporating the personal agent, the search agent, and the player model, is shown in Figure 6. As illustrated in this framework, we incorporate the two-channel fusion ART based player model together with the personal agent and the search agent in recommending functions and services

to the users. Specifically, the personal agent determines the appropriate type of services to recommend, such as hotel, restaurant, YOV, POI and shopping according to the current environment situation and user's context. Fusion ART then infers the player model especially player special preference and updates the personal agent with the players' current general interests during the interplay. By incorporating the personal agent with the adaptive user model, the system would be more adaptive to the players' habits and eccentricity. Based on the output of the personal agent and the player model's inferred result, the search agent works like a search engine that handles the retrieving of requested information from the database.

A typical example of how the various components work together is described as follows: When it is near dinner time, the personal agent suggests the service of providing the restaurant information to the player. Meanwhile, the player model indicates that this player prefers Japanese cuisine. Hence, the query of Restaurant and Japanese will be passed to the search agent, and Japanese restaurants with the highest matching scores will be recommended to the player. As the player requests for different restaurants, other cuisine types will be learnt by player model and updated as the player's interest after competing with other categories of general interest. Hence, during the following play time, the personal agent will give updated recommendations accordingly.

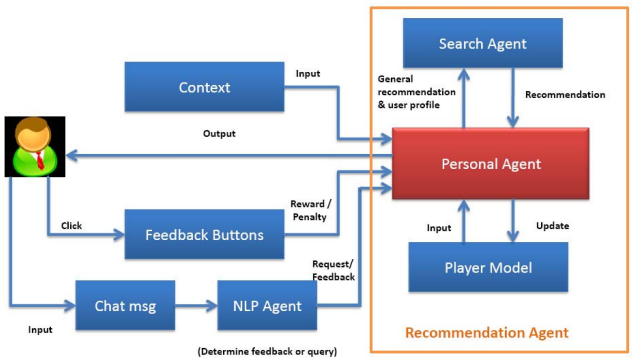

Figure 6. The architecture of recommendation agent.

VIII. Empirical Experiments

Our previous study [14] showed that the TD-FALCON based personal agents can adapt to the environment and improve their performance in real time. In this paper, we investigate whether we are able to learn player models that evolve and adapt with players during run time using fusion ART. Furthermore, we wish to evaluate the performance of the personal agents with adaptive player models in comparison with those without player models. Based on these considerations, we conduct three sets of experiments, namely experiments with static player profile, experiments with evolving player profile and experiments without player registered profile. Player registered profiles are the personal information captured by the system during user registration. Learning user models with and without initial profiles provide different challenges to fusion ART.

For the personal agents, we build a TD-FALCON network comprising 33 nodes in the sensory field, five nodes in the action field, and two nodes in the reward field. The parameter setting of TD-FALCON is shown in Table IV which has been proven to perform well in YOV Co-Space.

Table IV
TD-FALCON PARAMETER SETTING FOR PERSONAL AGENTS.

FALCON Parameters	Values
Choice parameter(α^{c1},α^{c2},α^{c3})	0.1, 0.1, 0.1
Learning rates(β^{c1},β^{c2},β^{c3})	1.0, 1.0, 1.0
Contribution parameter(γ^{c1},γ^{c2},γ^{c3})	1/3, 1/3, 1/3
Baseline vigilance parameter(ρ^{c1},ρ^{c2},ρ^{c3})	1.0, 1.0, 1.0
TD Learning Parameters	
TD learning rate α	1
Discount factor γ	0
Initial Q-value	0
ϵ-greedy Action Policy Parameters	
Initial ϵ value	0.5
ϵ decay rate	0.0002

For learning user models, we build a fusion ART network comprising 130 nodes in the input space field and two nodes in the output field. Moreover, in order to evaluate our system performance empirically, we design an automatic test procedure using synthetic player models, simulating the player's feedback to the agent's service based on a set of player logic. By matching the agent's recommendations with the player's expectation, we are able to compute reward signals that can be used to guide the agent in learning. A sample set of player logic is given in Table V as an illustration, where x is a value of Player's Interest in Table II and P is a set of the player's preference corresponding to the values in Table III.

Table V
LOGIC FOR GENERATING REWARDS BASED ON A SYNTHETIC PLAYER MODEL P.

IF	Time = Lunch or Dinner Time, Activity != Dinning, Request = Unknown,
EXPECT	Recommend restaurant with attributes $a_i \in$ P
IF	Time = Lunch or Dinner Time, Activity = Dinning, Request = Unknown, Interest contains x
EXPECT	Recommend x with attributes $a_i \in$ P
IF	Time = Bed Time, Request = Unknown
EXPECT	Recommend Hotel with attributes $a_i \in$ P

As the main role of the personal agents is to provide services to the players, we evaluate their performance by the accuracy of the service provided to the players over a period of time.

1) Experimenting with Static Player Profile: In this set of experiment, the player profile will not evolve during run-time. Fusion ART uses a default set of parameter values as listed in Table VI. The recommendation accuracy with static user profile averaged at 100-trial intervals over 3000 trials are shown in Figure 7. The results, obtained after averaging across five sets of experiments, generally indicate that the agents are able to gradually improve their performance through user feedback continuously over time. We can observe that when there is no change in the player profile, the performance is largely similar at all interaction cycles. This is because player registered profiles are beneficial to player model. They can be considered as the prior knowledge to initialize the player model.

Table VI
FUSION ART PARAMETER SETTING FOR PLAYER MODELS.

Fusion ART Parameters	Values
Choice parameter(α^{c1},α^{c2})	0.1, 0.1
Learning rates(β^{c1},β^{c2})	1.0, 1.0
Contribution parameter(γ^{c1},γ^{c2})	0.5, 0.5
Baseline vigilance parameter(ρ^{c1},ρ^{c2})	1.0, 0.0

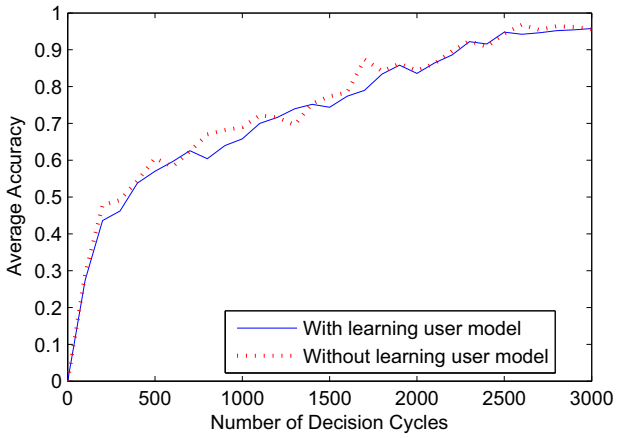

Figure 7. Recommendation accuracy of recommendation agents with registered player profile.

2) Experimenting with Evolving Player Profile: In this set of the experiments, we evolve the player profile during run-time to investigate how the agents may cope with a shift in player's interests. After 1500 learning trials, we changed 20% of the interest in synthetic player model by shifting the player's preference from Japanese to seafood and steamboat and from Aquatics to Sailing and Table Tennis. Moreover, we add in new preferences after 1500 learning trials to shift the user's general interest from dinning to shopping.

As shown in Figure 8, the recommendation accuracy of both systems are largely similar over the first 1500 learning trials, which are consistent with the previous set of experiments. As the target player profile evolves after 1500 learning trials, both of them suffer a critical drop in performance at the next 100 trials. However, they all manage to improve the performance afterwards. Even without the learning player model, due to the learning ability of the personal agent, it can adapt and give general recommendations according to the feedbacks from the players. However, as the player profile has changed, the recommendation agent without a player model does not have the capability of adapting to the new interest and preferences. Hence, the recommendation agent with learning player model significantly outperform its counterpart after the player profile evolves.

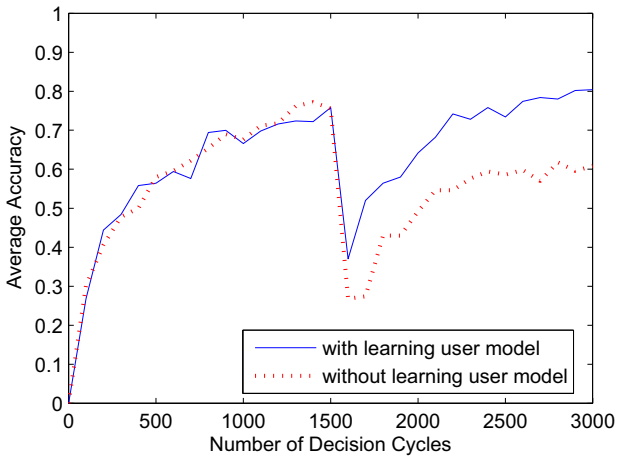

Figure 8. Recommendation accuracy of recommendation agent with evolving player profile.

3) Experimenting without Player Registered Profile: All the above mentioned experiments rely on the player's registered profile. However, in a realistic environment, players may not always fill in their information. They may log into the Co-Space by using a "guest" account. We note that this is more challenging because it requires the capability of learning from zero knowledge. In this set of experiments, we evaluate such a scenario. During the experiment, the player's preference exactly follows the setting of last experiment. The performance of learning player model, in terms of prediction accuracy, can be found in Figure 9.

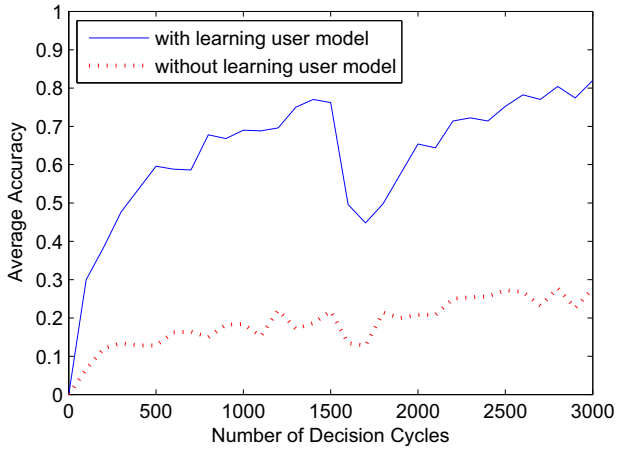

Figure 9. Recommendation accuracy of recommendation agent without registered player profile.

Since there is no more "pre-inserted hints", recommendation agent without a learning player model suffers a more significant drop of more than 50% comparing with that in the previous set of experiments. In contrast, fusion ART, despite without the player registered profile, quickly learns from zero knowledge and adapts to the player. Even after significant changes of player's interest, it manages to regain an accuracy of more than 80% eventually.

IX. Conclusions

The main objective of our work is to make the players' experience in virtual worlds more intense and personalized by employing learning player model technologies. In this paper, we have focused on the capability of agents to adapt over time to the habits and eccentricity of a particular player in virtual worlds. To this end, we present a personal agent based on FALCON, which enables an autonomous agent to adapt and function in a dynamic environment, and augmented it with an adaptive player model for personalized services in virtual worlds.

Our experimental results have thus far supported the validity of our multi-agent approach. While the study presented in this paper uses player models more tightly with the search agent, moving forward, we wish to expand the input state of personal agents by incorporating the user model directly. This will give rise to a richer user's context to enhance the decision making of personal agents. We also want to extend the player models to incorporate personal traits and other complex attributes so as to serve the players better.

Acknowledgment

This work was supported by the Singapore National Research Foundation Interactive Digital Media R&D Program, under research Grant NRF2008IDM-IDM004-037.

References

[1] D. Thomas and J. S. Brown, "Why virtual worlds can matter," *International Journal of Media and Learning*, vol. 1(1), pp. 37–49, 2009.

[2] J. Weinstein and J. Myers, "Same principles apply to virtual world expansion as to china and other new markets," *Media Village*, vol. 11, 2006.

[3] D. Rousseau and B. Hayes-Roth, "A social-psychological model for synthetic actors," *Stanford Knowledge Systems Laboratory Report KSL-97-07*, 1997.

[4] J. Weizenbaum, "ELIZA: a computer program for the study of natural language communication between men and machines," *Communications of the ACM*, vol. 9, 1996.

[5] S. Kopp, L. Gesellensetter, N. C. Krmer, and I. Wachsmuth, "A conversational agent as museum guide - design and evaluation of a real-world application," *Intelligent Virtual Agents*, pp. 329–343, 2005.

[6] S. Yoon, R. Burke, B. Blumberg, and G. Schneider, "Interactive training for synthetic characters," in *AAAI*, 2000, pp. 249–254.

[7] M. Gerhard, D. Moore, and D. Hobbs, "Embodiment and copresence in collaborative interfaces," *Int. J. Hum.-Comput. Stud.*, vol. 64(4), pp. 453–480, 2004.

[8] A.-H. Tan, "Adaptive resonance associative map," *Neural Networks*, vol. 8, pp. 437–446, 1995.

[9] D. Jan, A. Roque, A. Leuski, J. Morie, and D. Traum, "A virtual tour guide for virtual worlds," in *Intelligent Virtual Agents Conference (IVA)*, 2009, pp. 372–378.

[10] A. Leuski and D. Traum, "A statistical approach for text processing in virtual humans." in *26th Army Science Conference*, 2008.

[11] A.-H. Tan, N. Lu, and D. Xiao, "Integrating Temporal Difference Methods and Self-Organizing Neural Networks for Reinforcement Learning with Delayed Evaluative Feedback," *IEEE Transactions on Neural Networks*, vol. 9, no. 2, pp. 230–244, 2008.

[12] A.-H. Tan, "Self-organizing Neural Architecture for Reinforcement Learning," in *International Symposium on Neural Networks*, 2006, pp. 470–475.

[13] A.-H. Tan, G. A. Carpenter, and S. Grossberg, "Intelligence through Interaction: Towards a Unified Theory for Learning," in *Proceedings of International Symposium on Neural Networks*, 2007, pp. 1094–1103.

[14] Y.-L. Kang and A.-H. Tan, "Self-organizing agents for reinforcement learning in virtual worlds," to appear in *proceedings, International Joint Conference on Neural Networks*, 2010.

[15] D. Thue, V. Bulitko, M. Spetch, and E. Wasylishen, "Interactive storytelling: A player modelling approach," in *Proceedings of the third Artificial Intelligence and Interactive Digital Entertainment conference*, 2007.

[16] M. Sharma, S. Ontan, C. Strong, M. Mehta, and A. Ram, "Towards player preference modeling for drama management in interactive stories," in *In Proceedings of the Twentieth International FLAIRS Conference (FLAIRS07*. AAAI Press, 2007.

[17] D. Charles and M. Black, "Dynamic player modeling: A framework for player-centred digital games," in *International Conference on Computer Games: Artificial Intelligence, Design and Education*, 2004.

[18] A. J. Lockett, C. L. Chen, and R. Miikkulainen, "Evolving explicit opponent models in game playing," in *in: Proceeedings of the Genetic and Evolutionary Computation Conference*, 2007.

[19] R.J.S.Baker, P.I.Cowling, T.W.G.Randall, and P.Jiang, "Can opponent models aid poker player evolution?" in *IEEE Symposium on Computational Intelligence and Games*, 2008.

[20] R. Houlette, *Player Modelling for Adaptive Games*. Charles River Media, 2004.

[21] S. Hladky and V. Bulitko, "An evaluation of models for predicting opponent locations in first-person shooter video games," in *IEEE Symposium on Computational Intelligence and Games*, 2008.

[22] M. van der Heijden, S. Bakkes, and P. Spronck, "Dynamic formations in real-time strategy games," in *IEEE Symposium on Computational Intelligence and Games*, 2008.

[23] A.-H. Tan, "FALCON: A Fusion Architecture for Learning, COgnition, and Navigation," in *Proceedings of International Joint Conference on Neual Networks*, 2004, pp. 3297–3302.

[24] A.-H. Tan and H.-S. V. Soon, "Concept Hierarchy Memory Model: A neural architecture for conceptual knowledge representation, learning, and commonsense reasoning," *International Journal of Neural Systems*, vol. 7, no. 3, pp. 305–319, 1996.

[25] G. A. Carpenter, S. Grossberg, and D. B. Rosen, "Fuzzy ART: Fast Stable Learning and Categorization of Analog Patterns by an Adaptive Resonance System," *Neural Networks*, vol. 4, pp. 759–771, 1991.