TRAINING BELIEVABLE AGENTS IN 3D ELECTRONIC BUSINESS ENVIRONMENTS USING RECURSIVE-ARC GRAPHS

Anton Bogdanovych, Simeon Simoff
School of Computing and Mathematics, University of Western Sydney, NSW, Australia

Marc Esteva
Artificial Intelligence Research Institute (IIIA-CSIC), Campus UAB, Barcelona, Catalonia, Spain

Keywords: Autonomous Agents, Virtual Institutions, Implicit Training, Recursive-Arc Graphs.

Abstract: Using 3D Virtual Worlds for commercial activities on the Web and the development of human-like sales assistants operating in such environments are ongoing trends of E-Commerce. The majority of the existing approaches oriented towards the development of such assistants are agent-based and are focused on explicit programming of the agents' decision making apparatus. While effective in some very specific situations, these approaches often restrict agents' capabilities to adapt to the changes in the environment and learn new behaviors. In this paper we propose an implicit training method that can address the aforementioned drawbacks. In this method we formalize the virtual environment using Electronic Institutions and make the agent use these formalizations for observing a human principle and learning believable behaviors from the human. The training of the agent can be conducted implicitly using the specific data structures called recursive-arc graphs.

1 INTRODUCTION

Electronic marketplace is a space populated by computerised players that represent the variety of human and software traders, intermediaries, and information or infrastructure providers. Believable marketplaces are perceived as "marketplaces where people are", as "marketplaces that are alive and engaging". The majority of present electronic markets are focused on the backend transaction processing and catalogue-style interaction, and do not provide such perceptions.

Overall, the believability of a marketplace depends on the believability of the presence and interactions in it, including the players' behaviour and the narrative scenarios of the marketplace. A high degree of believability of the presence and interactions in an electronic marketplace can be achieved through its visualization as a 3D Virtual World (Bogdanovych, 2007). However, the believability of the players in this case becomes a serious concern, in particular when the marketplace accepts computerized autonomous agents as active participants.

The creation of such believable computerized agents, known as virtual humans, is an active area of computer science research, which attempts to model the full richness of human-like interactions including natural language communication, gestures, emotional expression, as well as the cognitive apparatus that underlies these capabilities (Huang et al., 2007). Most virtual human research has focused on the cognitive behaviour on the source side of the interaction (Rist et al., 2003), (Tomlinson et al., 2006) with a recent shift towards the "recipient" (Maatman et al., 2005).

The research in building models of different features that contribute to the believability of virtual humans (i.e. personality, social role awareness, self-motivation, illusion of life, etc.) utilizes contemporary developments in cognitive modeling (Prendinger and Ishizuka, 2001) and attempts to formalise those models in computational forms in order to implement them in virtual environments, in particular, in virtual worlds (Magnenat-Thalmann et al., 2005). As the complexity of such models increases, the complexity of their implementation increases too. However, passing the Turing test (different adaptations of which remain the only known research method for believability assessment) is still on the list of developers' goals (Livingstone, 2006).

In our research on believable electronic markets, we look at personalising the believability features,

Bogdanovych A., Simoff S. and Esteva M. (2008).
TRAINING BELIEVABLE AGENTS IN 3D ELECTRONIC BUSINESS ENVIRONMENTS USING RECURSIVE-ARC GRAPHS.
In *Proceedings of the Third International Conference on Software and Data Technologies - PL/DPS/KE*, pages 339-346
DOI: 10.5220/0001901103390346

adapting them towards the interacting player. In this sense *believable does not necessarily mean realistic*. We take a view of believability as providing an "anticipatory" feature of the technology (Pantic et al., 2006) underlying electronic markets, i. e. we are looking for (i) *components of human behaviour that can be integrated in electronic markets* and (ii) *the interpretation of such components by the environment*.

1.1 Learning Believable Behaviour

When the goal is to personalize the believability features, instead of trying to discover and explicitly program various believability characteristics some researchers rely on the simulation theory. The key hypothesis behind this theory can be best summarized by the cliché "to know a man is to walk a mile in his shoes" (Breazeal, 1999). It is assumed that simulation and imitation are the key technologies for achieving believability. In particular, using these techniques to produce more human-like behavior is quite popular in cognitive systems research (Schaal, 1999).

The motivation for scholars to rely on the simulation theory comes from observing human beings. Almost everything that constitutes humans' personality had to be learned at some point of their lives. The newborns are initially supplied with some very basic knowledge (reflexes) and have no knowledge about how to walk, how to talk or how to behave in public. All these behaviors are learned from observing and simulating other humans (Bauckhage et al., 2007).

Applying simulation theory to the development of autonomous agents is known as *imitation learning*. Up until recently most of the imitation learning research was focused on autonomous machines intended for deployment in physical world (Bauckhage et al., 2007). This focus led to a situation where research aimed at behavior representation and learning still first and foremost struggles with issues arising from embodiment dissimilarities (Alissandrakis et al., 2001), uncontrollable environmental dynamics (Aleotti et al., 2003), perception and recognition problems (Schaal, 1999) and noisy sensors (Schaal, 1999).

1.2 Imitation Learning in VWs

The aforementioned problems do not exist in Virtual Worlds. The sensors available there are not noisy, all the participants normally share similar embodiment (in terms of avatars), the environment is controllable and easily observable. Thus, using imitation learning for virtual agents represented as avatars within Virtual Worlds ought to be more successful than applying imitation learning to robots situated in physical world. Despite this fact, up until recently not many researchers from the imitation learning community were concerned with Virtual Worlds and Virtual Agents. Only a few scholars have taken this direction and most of them are concerned with gaming environments, where virtual agents are used as computer controlled enemies fighting with human players (Gorman et al., 2006), (Le Hy et al., 2004).

Having a focus on video games made it possible to introduce a number of limitations and simplifications, which are not acceptable in non-gaming Virtual Worlds (like 3D electronic markets). The algorithms described in (Gorman et al., 2006) seem to be quite successful in teaching the agent reactive behaviors, where next state an agent should switch into is predicted on the basis of the previous state and the set of parameters observed in the environment. These algorithms also prove to be quite useful in learning strategic behavior inside a particular video game (Quake II). The main limitation of this approach is that the long term goals of the players are assumed to be quite simple, namely to collect as many items as possible and to defeat their opponents (Thurau et al., 2004).

Provided the human only has simple goals as described above this method is quite sufficient and can be successfully used for training autonomous agents to execute human-like reactive behaviors while fighting the opponents in the selected video game. In many non-gaming Virtual Worlds, however, the situation is not that simple. Not only are the goals more complex, but there is also a need to be able to recognize the goals, desires and intentions of the human. Understanding the goals and subgoals is required to be able to assign the context to the training data and sort it into different logical clusters. Recognizing the desires and intentions is particularly important in situations when the agent is to replace the human in doing a particular task. For example, a human may wish to train a virtual agent to answer customer enquiries about the product or to participate in an auction on the human's behalf. One of the reasons why such tasks are impossible to achieve using the algorithms presented in (Gorman et al., 2006) and (Le Hy et al., 2004) is that there is no mechanism provided there to communicate human requests, as well as there is no method for the agent to infer human's desires and intentions.

In respect to making agent understand the desires and intentions of the human, existing approaches fall under one of the following two extreme cases. First case is to purely rely on explicit communication between agents and humans, when every goal, belief, desire, intention of the human and action the human trains the agent to perform is formalized for the agent. Second case is the fully implicit communication between humans and agents, when any explicit form of communication is considered unacceptable. In the first case it often becomes easier to program the agents than to train them and in the second case only simple reactive behaviors can be learned and strategical or tactical behaviors are mostly left out (as it is not possible to recognize complex human desires or intentions, like a human wanting to leave the agent to participate in an auction on his/her behalf).

1.3 The Scope of the Paper

In this paper we suggest that to be able for the agent to handle the complexity of the human actions and goals, the agent should not purely rely on its own intelligence but should expect some help from the environment it is situated in. As an example of such environment we consider the concept of Virtual Institutions (Bogdanovych, 2007), which are Virtual Worlds with normative regulation of participants' interactions.

In order for the agent to make use of the environment's formalization provided by Virtual Institutions the data structures that can map these formalisms onto the logical states of the agents are required. Most of the popular methods used for modeling the representation of the states of the agents and the mechanism of progressing trough these (i.e. final state machines, neural networks, decision trees etc.) utilize graphs for this task. Furthermore, Virtual Institutions also use graphs in many parts of the formalization. Therefore, using a graph-like structure for our purpose was a natural choice. For these graphs to satisfy our needs and allow for a general kind of learning we created a new data structure called the recursive-arc graphs.

The remainder of the paper is structured around the recursive-arc graphs concept. Before going into details of the proposed solution, Section 2 provides a description of the Virtual Institutions concept. In Section 3 it is shown how using Virtual Institutions enables implicit training of virtual agents and the recursive-arc graphs that are used for modeling the agent's state space are throughly explained. Finally, Section 4 summarizes the contribution and outlines the directions of future work.

2 VIRTUAL INSTITUTIONS

Virtual Institutions (Bogdanovych, 2007) are a new class of normative Virtual Worlds, that combine the strengths of 3D Virtual Worlds and normative multiagent systems, in particular, Electronic Institutions (Esteva, 2003). In this ”symbiosis” the 3D Virtual World component spans the space for visual and audio presence, and the electronic institution component takes care for enabling the formal rules of interactions among participants.

The interaction rules include the roles the participants can play, the groups of activities each role can engage into, the interaction protocols associated with each group of the activities and a set of actions they can perform (see (Esteva, 2003) for more details). The Virtual World is separated into a number of logical spaces (scenes), connected with each other through corridors or teleports (also called transition). Only participants playing particular roles are admitted to a scene. Once admitted the participants should follow the interaction protocol specified for each of them.

The correct application of the institutional rules and the functioning of the 3D environment of the virtual institution is enabled by a three-layered architecture. Figure 1 presents a high-level overview of this architecture. It is presented in three conceptually (and technologically) independent layers, as follows.

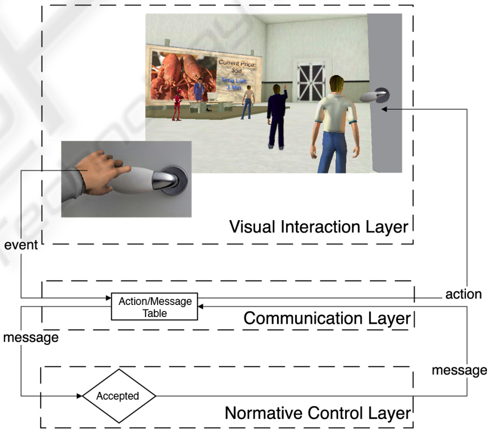

Figure 1: Three-layered architecture of Virtual Institutions.

Normative Control Layer. Its task is to regulate the interactions between participants by enforcing the institutional rules.

Communication Layer. Its task is to causally connect the above discussed institutional dimensions with the virtual world representation of the institution and transform the actions in the virtual world into messages, understandable by the institutional infrastructure and vice versa.

Visual Interaction Layer. Its task is to support the immersive interaction space of a virtual institution and indicate institutional actions, if such occur, to the inhabitants. Technologically, this layer includes a 3D virtual world and the interface that converts communication messages from the Communication Layer.

3 IMPLICIT TRAINING

Existing 3D Virtual Worlds are mostly human centered with very low agent involvement. Virtual institutions, in contrast, is an agent-centered technology, which treats humans as heterogenous, self-interested agents with unknown internal architecture. Every human participant (principal) is always supplied with a corresponding software agent, that communicates with the institutional infrastructure on human's behalf. The *couple agent/principal* is represented by an avatar. Each avatar is manipulated by either a human or an autonomous agent through an interface that translates all activities into terms of the institution machine understandable language. The autonomous agent is always active, and when the human is driving the avatar the agent observes the avatar actions and learns how to make the decisions on human's behalf. At any time a human may decide to let the agent control the avatar via ordering it to achieve some task. If trained to do so the agent will find the right sequence of actions and complete the task imitating the human.

The research conducted by (Bauckhage et al., 2007) suggests that in order to achieve believable agent behavior the agent should learn reactive behaviors, localized tactics and strategical behaviors. The authors, however, do not suggest an integrated solution for learning all these behaviors in a consistent manner and their method has a number of limitations.

To address these limitations in the implicit training method we rely on the concept of the pyramid of virtual collaboration proposed by (Biuk-Aghai, 2003).

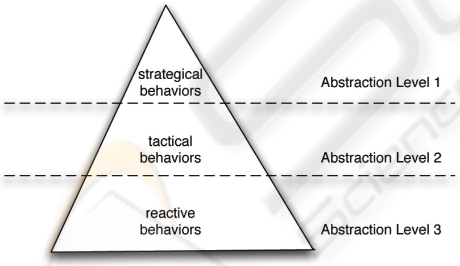

Figure 2: The Pyramid: Different Levels of Abstraction.

Figure 2 outlines the adaptation of this concept to implicit training of autonomous agents in Virtual Institutions. The pyramid shows the integration of different levels of abstraction of the training data and suggests that training should happen on each of the abstraction levels simultaneously. We distinguish between three abstraction levels, where the lowest level corresponds to pure reactive behaviors, middle levels represents tactical behaviors and the highest level stands for strategical behaviors. Next, we present the scenario that illustrates the capabilities of implicit training and explains each of the abstraction levels.

3.1 Training Scenario

To simplify the understanding of the implicit training concept we propose the scenario outlined in Figure 3. The virtual institution presented here consists of a building containing three rooms (scenes) connected by corridors (transition). The task an agent has to learn is walking into the last room and participating in an auction there. The human trains the agent by controlling the avatar while performing a task of buying fish in the TradeRoom.

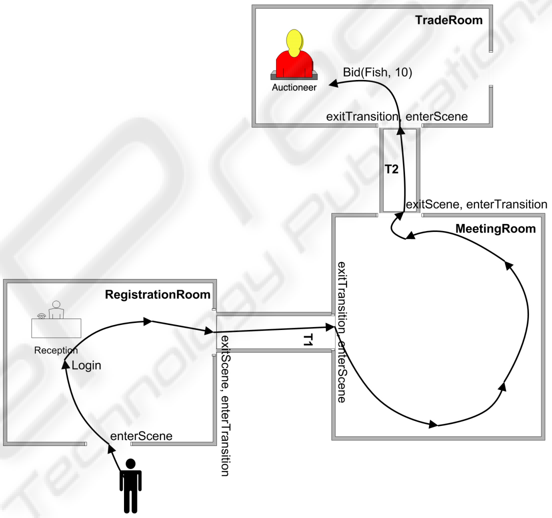
TradeRoom
Auctioneer
Bid(Fish, 10)
exitTransition, enterScene
T2
exitScene, enterTransition
MeetingRoom
exitTransition, enterScene
RegistrationRoom
Reception
Login
T1
exitScene, enterTransition
enterScene

Figure 3: Outline of a prototypical Virtual Institution.

The actions of the highest level of abstraction in this scenario are strategical behaviors. Such actions are strictly controlled by the institution and can be prohibited if a certain activity is not consistent with the institutional state or with the role a participant is playing. In our scenario those are "enterScene", "exitScene", "enterTransition", "exitTransition", "login" and "bid". The tactical behaviors from abstraction level 2 in this case are actions not controlled by the institution and actions independent from a particular Virtual World. These are approaching the receptionist, leaving the receptionist, approaching the auctioneer and leaving the auctioneer. The lowest level of abstraction deals with the actions tightly connected with the selected Virtual World. Learning such actions helps the agent to learn correct reactive behaviors influenced by static and dynamic objects located in the training environment.

Let's assume that after participating in the fish auction the human has entered the building, registered himself at the reception desk, walked into the

TradeRoom and bought a box of fish for the price of $10 . In meanwhile the agent was observing the human and learning from him on each of the abstraction levels. Now the human decides to use a special command "Do:bid(fish, 10)" instructing the agent to buy fish. To satisfy this request the agent searches the prerecorded sequence of the actions of the highest abstraction level for the presence of bid(x,y) function. Once the function is found, the agent knows which sequence of institutional level actions will lead to achieving the goal. At the second level it knows that for doing so it will first need to approach the receptionist and then approach the auctioneer. Finally, at the lowest level the agent knows which actions it has to execute for a believable imitation of the human movement. The aforementioned reasoning will result in the following behavior: the avatar enters the Registration Room, the avatar reproduces the trajectory of the human and approaches the reception desk, the request for login information is received and the agent sends the login details. In the similar way the agent continues its movement to the Trade Room, where it offers $10 for the box of fish. If the agent wins the lot - the scenario is finished; if the price this time is higher - the agent will request the human intervention.

3.2 Technological Solution

The implicit training is implemented as a lazy learning method, based on graph representation. The Virtual Institution corresponds to the scenario outlined in Figure 3. The implicit training method is demonstrated on learning movement styles.

3.2.1 Training Data

The data provided by Virtual Institution consists of: institutional messages executed as a result of human actions in the Virtual World, the attributes of the objects located in the environments, the attributes of the avatars located in the environment and the information about the actions executed by the avatars. The aforementioned attributes are updated by the system every 50 Msc and contain information like the coordinates, transformation vectors and other parts of the mathematical model of the Virtual World.

This data is expressed using the coordinate space of the 3D Virtual World we are dealing with and, therefore, is highly environment dependent. Using it for agent training in such form would make it impossible to come up with a general solution that could be used in a different Virtual World or handle the situations the agent wasn't directly trained to deal with. Therefore, this data has to be further processed and filtered into the above described levels of abstractions.

In our scenario we use the parameters of the objects visible to the agent's avatar as the learning attributes and the action performed by the agent as the variable to be learned. To avoid operating with coordinates, on the lowest level of abstraction we can substitute coordinates of the visible objects by distances and space orientation. For example, instead of recording that the object "PineTree" is located at (10,30,158) we could say that it is located NE (north east from the avatar) and the distance to it is 10 m. The actions of the agent should also be abstracted. On this level, instead of using coordinates we approximate the current avatar position based on the previous position and describe the change as either of the following: "MF", "TR", "TL", or "J". Here "MF" corresponds to the avatar moving forward by one unit, "TR" corresponds to turning right by one unit, "TL" stands for turning left and "J" represents jumping up. This kind of representation helps to train the agent to perform localized reactive behaviors (like different ways of approaching an avatar or an object) on the lowest level of abstraction.

On the second level of abstraction the agent learns tactical movement behaviors like an avatar approaching an object or another avatar, leaving an object or avatar, following an avatar, etc. On the highest level of abstraction we use the data provided by the formalization of the institution as the source for teaching agent to perform the normative level actions. These include entering rooms, placing bids on auctions, initiating conversations with agents, etc.

Next, we describe the recursive-edge graph data type that can successfully integrate the data from all the aforementioned levels of abstraction.

3.2.2 Constructing the Learning Graph

To be able to collect and use the data on different levels of abstraction we introduce the recursive-arc graph data structure outlined in Figure 4. The figure is divided into three parts (a,b,c) each representing a different abstraction level.

The part a) of the figure corresponds to the highest level of abstraction represented by the institutional specification. On this level the agent learns which combinations of the institutional messages will lead to achieving which goals. When a human principal that conducts the training enters the institution, the corresponding autonomous agent begins recording principal's actions, storing them inside a learning graph. At this level of abstraction the nodes of this graph correspond to the institutional messages, executed in response to the actions of the human. Each of the nodes is associated with two variables: the message name together with parameters and the probabil-

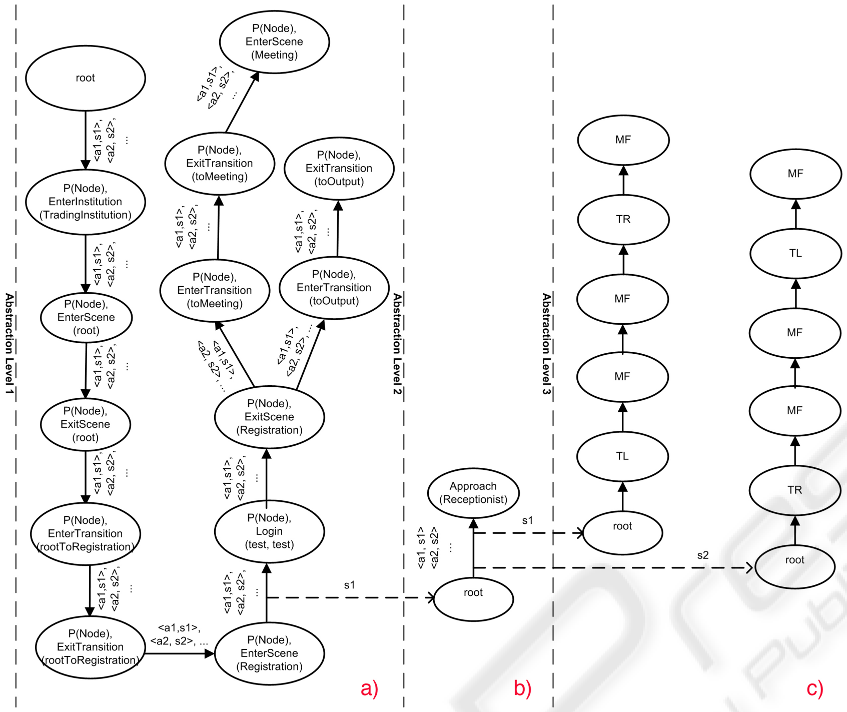
Abstraction Level 1
Abstraction Level 2
Abstraction Level 3
root
<a1,s1>, <a2, s2>, ...
P(Node), EnterInstitution (TradingInstitution)
P(Node), EnterScene (root)
P(Node), ExitScene (root)
P(Node), EnterTransition (rootToRegistration)
P(Node), ExitTransition (rootToRegistration)
P(Node), EnterScene (Registration)
P(Node), Login (test, test)
P(Node), ExitScene (Registration)
P(Node), EnterTransition (toMeeting)
P(Node), ExitTransition (toMeeting)
P(Node), EnterScene (Meeting)
P(Node), EnterTransition (toOutput)
P(Node), ExitTransition (toOutput)
s1
Approach (Receptionist)
<a1, s1> <a2, s2> ...
root
s1
s2
root
TL
MF
MF
TR
MF
root
TR
MF
MF
TL
MF
a)
b)
c)

Figure 4: A fragment of the learning graph.

ity $P(Node)$ of executing the message. The probability used for estimating the likelihood of executing a particular institutional message in a given state of the agent. This probability is continuously updated, and in the current implementation it is calculated as $P(Node) = \frac{n_a}{n_o}$. Here n_o is the number of times a principal had a chance to perform a particular action and n_a shows how times s/he actually did perform it.

The arcs connecting the nodes are associated with the data from the next level of abstraction. On the figure the arcs are marked with an array of pairs $\langle a_i, s_i \rangle$. Here $(a_1, \ldots, a_n)$ are the attribute vectors that the avatar associated with the agent was able to observe and $(s_1, \ldots, s_l)$ are the sequences of actions that were executed as the result. Each pair $\langle a_n, s_n \rangle$ is stored in a hashtable, where a_i is the key of the table and s_i is the value. Each a_i consists of the list of parameters: $a_i = \langle p_1, \ldots\ p_k \rangle$.

The simplifying assumption behind the training is that the behaviour of the principle is only influenced by what is currently visible to the avatar. We limit the visible items to the objects located in the environments and other avatars. So, the parameters used for learning are recorded in the following form: $p_i = \langle V_o, V_{av} \rangle$, where V_o is the list of currently visible objects; V_{av} is the list of currently visible avatars. The list of the visible objects is represented by the following set: $V_o = \{\langle O_1, D_1, P_1 \rangle, \ldots, \langle O_m, D_m, P_m \rangle$. Each O_j here is an object that visible to the agent. D_j are the distances from the current location of the agent to the centers of mass of such objects and P_j are the textual labesl assigned to the direction vector pointing from the center of mass of the avatar towards an object (i.e. “NE” – North-East).

The list of visible avatars is specified as follows: $V_{av} = \{\langle N_1, R_1, DAv_1, PAv_1 \rangle, \ldots, \langle N_p, R_k, DAv_p, PAv_p \rangle$. Here, N_k correspond to the names of the avatars that are visible to the user, R_k are the roles played by those avatars, DAv_k are the distances to those avatars and PAv_k are again the direction vectors.

Each of the sequences (s_i) is represented as a new graph from the next level of abstraction as shown in Figure 4 b). In these graphs the lower level actions (“Approach” in our example) mark the nodes and the arcs have similar format as in Figure 4 a), except for the fact that an even lower level of abstraction is used there. In our example of this lowest level we are dealing with moving, turning, jumping, etc.

The training is continuously conducted on each level of abstraction as new actions are observed there.

3.2.3 Applying the Learning Graph

When the construction of the learning graph is completed an agent is ready to accept commands from the principal. We have specified a list of textual commands that are typed into the chat window of the simulation engine. Each command includes a special keyword “Do:” and a valid institutional level message, e.g.“Do:EnterScene(Meeting)”.

The nodes of the learning graph are seen as internal states of the agent, the arcs determine the mechanism of switching between states and $P(Node)$ determines the probability of changing the agent’s current state to the state determined by the next node. Once the agent reaches a state $S(Node_i)$ it considers all the nodes connected to $Node_i$ that lead to the goal node and conducts a probability driven selection of the next node ($Node_k$). If $Node_k$ is found: the agent changes its current state to $S(Node_k)$ by executing the best matching sequence of the lower abstraction level stored on the arc that connects $Node_i$ and $Node_k$. If there are no such actions present on the arc - the agent sends the message associated to $Node_k$ and updates it’s internal state accordingly. This process is continued recursively for all the abstraction levels.

The parameters currently observed by the agent must match the parameters of the selected sequence as close as possible. To do so the agent creates the list of parameters it can currently observe and passes this list to a classifier (currently, a nearest neighbor classifier (Hastie and Tibshirani, 1996)). The later returns the best matching sequence and the agent executes each of its actions. The same procedure continues until the desired node is reached.

4 CONCLUSIONS

We have developed our argument for the need of implicit training of virtual agents participating in 3D Electronic Business Environments and highlighted the role of the environment itself in the feasibility of implicit training. Formalizing the environment with Virtual Institutions can significantly simplify the learning task. However, for the agent to use these formalization successfully it requires specific data structures to operate with. The paper has presented an example of the data structure called recursive-arc graphs that could be used by the agents participating in Virtual Institutions. Future work includes the development of the prototype that would confirm that such data structures are indeed suitable for training believable agents in 3D electronic business environments.

ACKNOWLEDGEMENTS

This research is partially supported by an ARC Discovery Grant DP0879789, the e-Markets Research Program (http://e-markets.org.au), projects AT (CON-SOLIDER CSD2007-0022), IEA (TIN2006-15662-C02-01), EU-FEDER funds, and by the Generalitat de Catalunya under the grant 2005-SGR-00093.

REFERENCES

Aleotti, J., Caselli, S., and Reggiani, M. (2003). Toward Programming of Assembly Tasks by Demonstration in Virtual Environments. In *IEEE Workshop Robot and Human Interactive Communication.*, pages 309 – 314.

Alissandrakis, A., Nehaniv, C. L., and Dautenhahn, K. (2001). Through the Looking-Glass with ALICE: Trying to Imitate using Correspondences. In *Proceedings of the First International Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*, pages 115–122, Lurid, Sweden.

Bauckhage, C., Gorman, B., Thurau, C., and Humphrys, M. (2007). Learning Human Behavior from Analyzing Activities in Virtual Environments. *MMI-Interaktiv*, 12(April):3–17.

Biuk-Aghai, R. P. (2003). *Patterns of Virtual Collaboration*. PhD thesis, University of Technology Sydney, Australia.

Bogdanovych, A. (2007). *Virtual Institutions*. PhD thesis, University of Technology, Sydney, Australia.

Breazeal, C. (1999). Imitation as social exchange between humans and robots. In *Proceedings of the AISB Symposium on Imitation in Animals and Artifacts*, pages 96–104.

Esteva, M. (2003). *Electronic Institutions: From Specification to Development*. PhD thesis, Institut d’Investigació en Intelligència Artificial (IIIA), Spain.

Gorman, B., Thurau, C., Bauckhage, C., and Humphrys, M. (2006). Believability Testing and Bayesian Imitation in Interactive Computer Games. In *Proceedings of the 9th International Conference on the Simulation of Adaptive Behavior (SAB’06)*, volume LNAI 4095, pages 655–666. Springer.

Hastie, T. and Tibshirani, R. (1996). Discriminant adaptive nearest neighbor classification and regression. In Touretzky, D. S., Mozer, M. C., and Hasselmo, M. E., editors, *Advances in Neural Information Processing Systems*, volume 8, pages 409–415. The MIT Press.

Huang, T. S., Nijholt, A., Pantic, M., and Pentland, A., editors (2007). *Artifical Intelligence for Human Computing, ICMI 2006 and IJCAI 2007 International Workshops*, volume 4451 of *Lecture Notes in Computer Science*. Springer.

Le Hy, R., Arrigony, A., Bessiere, P., and Lebeltel, O. (2004). Teaching bayesian behaviors to video game characters. *Robotics and Autonomous Systems*, 47:177–185.

Livingstone, D. (2006). Turing’s test and believable AI in games. *Computers in Entertainment*, 4(1):6–18.

Maatman, R. M., Gratch, J., and Marsella, S. (2005). Natural behavior of a listening agent. *Lecture Notes in Computer Science*, pages 25–36.

Magnenat-Thalmann, N., Kim, H., Egges, A., and Garchery, S. (2005). Believability and Interaction in Virtual Worlds. In *Proceedings of the Multimedia Modelling Conference (MMM'05)*, pages 2–9, Washington, DC, USA. IEEE Computer Society.

Pantic, M., Pentland, A., Nijholt, A., and Huang, T. (2006). Human computing and machine understanding of human behavior: a survey. In *ICMI '06: Proceedings of the 8th international conference on Multimodal interfaces*, pages 239–248, New York, NY, USA. ACM.

Prendinger, H. and Ishizuka, M. (2001). Let's talk! socially intelligent agents for language conversation training. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 31(5):465–471.

Rist, T., Andre, E., and Baldes, S. (2003). A flexible platform for building applications with life-like characters. In *IUI '03: Proceedings of the 8th international conference on Intelligent user interfaces*, pages 158–165, New York, NY, USA. ACM.

Schaal, S. (1999). Is imitation learning the route to humanoid robots? *Trends in cognitive sciences*, (6):233–242.

Thurau, C., Bauckhage, C., and Sagerer, G. (2004). Learning Human-Like Movement Behavior for Computer Games. In *Proceedings of the 8-th Simulation of Adaptive Behavior Conference (SAB'04)*, pages 315–323. MIT Press.

Tomlinson, B., Yau, M. L., and Baumer, E. (2006). Embodied mobile agents. In *Proceedings of AAMAS '06 conference*, pages 969–976, New York, NY, USA. ACM.