Evaluating an Animated Pedagogical Agent

Antonija Mitrovic and Pramuditha Suraweera

Intelligent Computer Tutoring Group
Department of Computer Science, University of Canterbury
Private Bag 4800, Christchurch, New Zealand
tanja@cosc.canterbury.ac.nz, psu16@student.canterbury.ac.nz

Abstract. The paper presents SmartEgg, an animated pedagogical agent developed for SQLT-Web, an intelligent SQL tutor on the Web. It has been shown in previous studies that pedagogical agents have a significant motivational impact on students. Our hypothesis was that even a very simple and constrained agent, like SmartEgg, would enhance learning. We report on an evaluation study that confirmed our hypothesis.

1 Introduction

Computers and Internet access are available in most schools today and offer a wealth of information to students. However, the access to computers does not guarantee effective learning, as many students lack the abilities to find their way through a vast amount of accessible knowledge. Students need guidance, either from human or computerized tutors. Recently, there have been several research projects that concentrate on the development of animated pedagogical agents, lifelike creatures that inhabit learning environments. Experiments have shown that such agents significantly increase student motivation and perception of their learning. Here we present SmartEgg, an animated pedagogical agent for SQLT-Web, and the initial evaluation of it.

We have developed SQL-Tutor, a standalone system for the SQL database language [9,10]. The system has been used by senior computer science students and has been found easy to use, effective and enjoyable [11]. Recently, SQL-Tutor was extended into a Web-enabled system, named SQLT-Web, and our initial experiences show that students find it equally enjoyable and useful [12]. SQLT-Web has been used only by local students. We plan to have SQLT-Web widely accessible soon, in which case students outside our university may find some aspects of the system more difficult to grasp. Therefore, we have started exploring possibilities of providing more feedback, and providing it in a manner that would motivate students.

We discuss animated pedagogical agents in section 2. Section 3 introduces SQL-Tutor and the Web-enabled version of it. We present SmartEgg in section 4, focusing on its implementation, behaviour space and communication with SQLT-Web. Section 5 presents the results of the initial evaluation, followed by discussion and conclusions.

2 Animated Pedagogical Agents

Animated pedagogical agents are animated characters that support student learning. They broaden the communication channel by using emotive facial expressions and body movements, which are very appealing to students. Pedagogical agents are extremely important for student motivation, as they provide advice and encouragement, empathize with students, and increase the credibility and utility of a system. Several studies have investigated the affective impact of agents on student learning and revealed the *persona effect*, "which is that the presence of a lifelike character in an interactive learning environment - even one that is not expressive - can have a strong positive effect on student's perception of their learning experience" [6]. Experiments have shown that students are much more motivated when the agent is present, tend to interact more frequently and find agents very helpful, credible and entertaining.

Animated pedagogical agents may be presented as cartoon-style drawings, real video or 3D models. Most agents are fully bodied, and use facial expressions and body movements to communicate emotions. An agent may exist within the learning environment, i.e. be immersed into the learning environment, move through it and manipulate objects within. It is also possible for an agent to exist in a separate window. Agents may adhere to the laws of physics, or may be stylised to emphasize emotions. Agents' behaviour may be specified off-line, manually. Ideally, behaviour should be generated online, dynamically, so as to correspond to the changes in the learning environment.

Herman the Bug [7] is an animated pedagogical agent for the *Design-A-Plant* learning environment, in which children learn about plant anatomy and physiology by designing plants for specific environments. Herman is a 3D model, immersed into the learning environment, capable of performing engaging actions, such as diving into plant roots, bungee jumping, shrinking and expanding.

Adele (Agent for Distance Education – Light Edition) [5] is an autonomous agent that facilitates distance learning. The agent is used with a simulated environment in which students solve problems. Adele consists of three components: a reasoning engine, which monitors student's actions and generates appropriate pedagogical responses to them, an animated persona that runs in a separate window, and a session manager, which enables multiple students to use the system concurrently.

Steve (Soar Training Expert for Virtual Environments) [4] is a human-like animated agent that cohabits a virtual reality environment and helps students learn to perform procedures. Being a 3D model immersed in a simulation, Steve can perform not only the pedagogical functions common in intelligent educational systems, but also can demonstrate actions by manipulating objects in the simulated environment. Multiple Steve agents can inhabit the environment, thus giving a possibility to teach team tasks.

PPP Persona [3] guides the learner through Web-based material by pointing to important elements of Web pages, and providing additional auditory comments. There are five different characters, three of which are video-based, and the remaining two are cartoon characters. AlgeBrain [1] is a Web-based intelligent tutoring system that teaches students how to solve algebraic equations. The pedagogical agent used is a cartoon-like drawing that appears in a separate window.

Three architectures have emerged for online generation of agent behaviour [4]. The *behaviour sequencing approach* is based on a behaviour space, which is a library of predefined primitives (actions, speech elements etc). In an instructional session, the behaviour of an agent is assembled on-line from the primitives, by a behaviour sequencing engine. The behaviour space of Herman the Bug consists of 30 animated segments of the agent performing various actions, and of 160 audio clips and songs [6]. These actions are combined at runtime by the emotive-kinaesthetic behaviour sequencing engine [7].

The second architecture is the *layered generative approach*, where animations are generated in real time. This is the architecture Steve is based on, and it is especially suitable for immersive environments, but it requires a much higher rendering computation load. Finally, the *state machine compilation approach* composes behaviour out of primitives, but generates a state machine, so that the behaviour of an agent can adapt at run time to student actions. Andre, Rist and Muller [2] describe a presentation planner, which develops a navigation graph from given goals. A navigation graph contains all presentation units with associated durations and transitional information.

3 An Intelligent SQL Tutor

SQL-Tutor is an Intelligent Teaching System (ITS) that helps students to learn SQL [9,10]. It is designed as a problem-solving environment and as such is not intended to replace classroom instruction, but to complement it. We assume that students are already familiar with the database theory and fundamentals of SQL. Students work on their own as much as possible and the system intervenes when the student is stuck or asks for help.

The standalone version of the system consists of an interface, a pedagogical module that determines the timing and content of pedagogical actions, and a student modeller that analyses student answers. There is no domain module, as usual in ITSs, which can solve the problem being posed to a student. The system contains definitions of several databases, implemented on the RDBMS used in the lab. SQL-Tutor also contains a set of problems for specified databases and the ideal solutions to them. In order to be able to check the correctness of the student's solution, SQL-Tutor uses domain knowledge represented in form of constraints, as described in [11]. Student solutions are compared to the ideal solutions and the domain knowledge.

At the beginning of a session, SQL-Tutor selects a problem for the student to work on. When the student enters the solution, the pedagogical module (PM) sends it to the student modeller, which analyses the solution, identifies mistakes (if there are any) and updates the student model appropriately. On the basis of the student model, PM generates an appropriate pedagogical action (i.e. feedback). When the current problem is solved, or the student requires a new problem to work on, the pedagogical module selects an appropriate problem on the basis of the student model.

SQL-Tutor uses Constraint-Based Modelling (CBM) [13] to form models of its students. CBM is a computationally efficient student modelling approach, which reduces the complex task of inducing student models to simple pattern matching. The

strength of CBM lies in domain knowledge, represented in the form of state constraints, which contain the basic principles of a domain.

We have recently developed SQLT-Web, a Web-enabled version of SQL-Tutor [12]. The basic philosophy remains the same, but SQLT-Web is capable of dealing with multiple students. It has been developed in a programmable CL-HTTP Web server [8]. All pedagogical functions (student modelling, generation of feedback and selection of problems) are performed on the server side. The system communicates to the student's Web browser by generating HTML pages dynamically. The server stores all student models at the same place, thus allowing a student to access the system from any machine.

4 SmartEgg: an Animated Pedagogical Agent for SQLT-Web

SmartEgg is an animated pedagogical agent developed by our group for SQLT-Web. It is a cartoon-like character that gives feedback on student actions. As the agent was developed for a fully functional ITS, it was possible to have SQLT-Web to generate student models and appropriate feedback. Therefore, our agent has to perform much simpler tasks in comparison to agents discussed in the previous section.

The agent explains system's functions, provides feedback on student's actions and informs students about additional ways of getting help or background information. The project is still in its initial phases, and so far the agent presents all information in textual form. In the later phases, we plan to broaden the types of available feedback, including audio, and to extend agent's functionality.

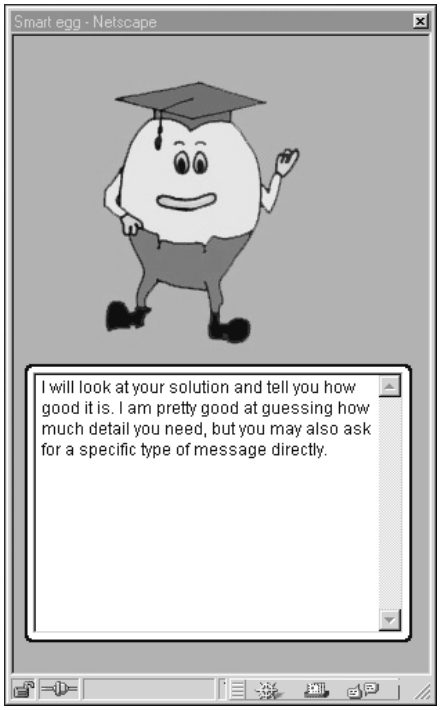

Fig. 1: Introduction to SmartEgg

SmartEgg is implemented as a Java applet, by using the animation toolkit of Adele [5]. An appropriate character was developed (illustrated in figure 1), and thirty-eight frames were sketched to define the gestures. The animation toolkit swaps frames and uses techniques such as morphing to perform animations. Currently, there are 14 gestures that SmartEgg can perform, requiring two to five frames each. The library of gestures consists of presentation gestures (e.g. pointing), reactive gestures (used to present feedback) and idle-time gestures (e.g. waiting for a solution).

The required behaviours were developed next. Behaviour is a sequence of several gestures. The behaviours of our agent are pre-

specified, and not dynamically generated. The SmartEggs's behaviour space consists of three main categories of behaviours: introductory, explanatory and congratulatory. *Introductory behaviours* accompany initial interactions, introducing the system's functions and describing levels of feedback to new users. Feedback messages from SQLT-Web are delivered to students using *explanatory behaviours.* For each type of feedback, there is a set of behaviours the pedagogical agent can perform. *Congratulatory behaviours* are an attempt to motivate users. SmartEgg congratulates the student when a correct answer is submitted and displays disappointment after an incorrect submission.

SmartEgg follows a predefined set of rules when selecting an appropriate behaviour from its behaviour space. This procedure is based on the student's interactions with SQLT-Web. Each distinct state (e.g. login, solving a problem, logout) is assigned three different behaviours to ensure variation in the agent's appearance.

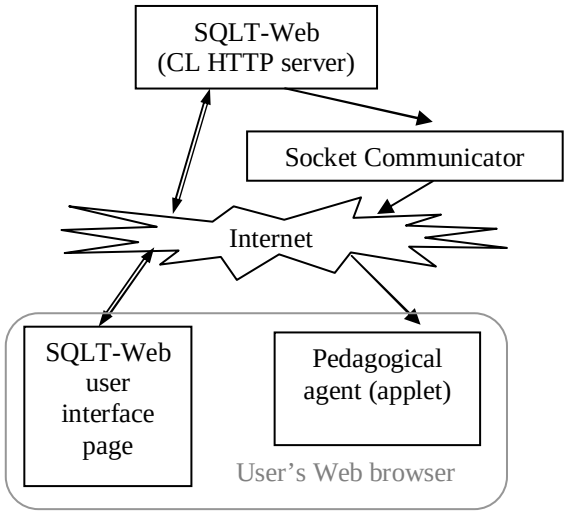

Fig. 2: Architecture of SQLT-Web with pedagogical agent

Finally, the applet persona was incorporated with SQLT-Web. The pedagogical agent's Java applet and the server are required to exchange messages in order for the agent to receive the feedback text and know the actions performed by the user. This was achieved by implementing a Java socket connection between the server and the applet. The agent consists of a dedicated thread of execution that waits to receive messages from the server. For each received message, the agent selects an appropriate behaviour by using the behaviour selection rules, which is then carried out by the animated persona. Figure 2 illustrates the architecture of SQLT-Web and the pedagogical agent.

5 Evaluation of SmartEgg

Our goal when developing SmartEgg was to increase the motivation of students by presenting feedback in an engaging way. We started with a hypothesis that the existence of a simple animated pedagogical agent would enhance students' perception of the system (as reflected in the students' subjective ratings of the system), and would support learning, resulting in better understanding and application of the underlying knowledge. Both gains would come from the motivational impact of the agent. Earlier studies [1,3,4,7] have shown that pedagogical agents have such effects on students; however, in these cases, the agents were much more sophisticated than SmartEgg. Here we set to determine whether even a very simple and constrained agent would enhance learning.

5.1 Experimental Setting

In October 1999 we performed an evaluation study, which involved second year students enrolled in an introductory database course. The students used the system in a 2-hour lab session and were randomly assigned to a version of the system with and without the agent (the agent and the control group respectively). SQLT-Web and SmartEgg conveyed exactly the same information to the students, as we wanted to determine the impact of the agent's existence on students' learning.

The study started with a pre-test, consisting of three multi-choice questions. After that, students interacted with the system. The problems and the order in which they were presented were not identical, as students were allowed to select problems by themselves, or let the system to select appropriate problems based on their student models. After working with the system, students completed a post-test consisting of three multi-choice questions of the same difficulty as the ones in the pre-test. They also filled a user questionnaire, the purpose of which was to evaluate the students' perception of SmartEgg and SQLT-Web.

5.2 System/Agent Assessment

The questionnaire consisted of 16 questions based on the Likert scale with five responses ranging from *very good* (5) to *very poor* (1). Students were also allowed to put free-form responses. Out of 26 students who participated in the study, 22 completed questionnaires.

The analysis of the responses revealed that the students liked SmartEgg. When asked to rate how much they enjoyed the system, the average rating for the agent group was 4.5 and for the control group 3.83 (Table 1). The majority (60%) of the agent group students chose option 5, compared to only 33% of the control group. The difference is significant ($t=1.79$, $p=.03$).

Both groups were equally comfortable with the interface, in the terms of how much time it took to learn it, and the ease of using the interface. The students were also asked to rate the amount learnt from the system. Both groups chose similar values, the means being 3.8 for the agent group and 3.92 for the control group. This result was expected as both groups received identical feedback.

However, when asked to rate the usefulness of feedback, the mean for the agent group was 4.8 and for the control group was 4.09. The majority (80%) of the students who used the agent rated the system as very useful (option 5), and only 42% of the control group chose the same option. As both versions of the system presented the same problem-based messages, it is clear from the findings that the students who used the agent found it easier to comprehend the feedback from the system. The difference in rating the usefulness of feedback is significant (t=2.15, p=.015). The written comments were also very positive.

	Mean		Standard deviation	
	Agent group	Control group	Agent group	Control group
Enjoyment rating	4.50	3.83	0.71	1.03
Time to learn interface (min)	11.00	10.83	10.22	9.25
Ease of using the interface	4.10	3.73	0.74	1.01
Amount learnt	3.80	3.92	0.79	0.67
Usefulness of feedback	4.80	4.09	0.42	1.04

Table 1: Mean responses for system/agent assessment

5.3 Learning Efficiency and Effectiveness

All actions students performed in the study were logged, and later used to analyse the effect of the agent on learning (Table 2). The students in the agent group spent 55.9 minutes interacting with the system, and the control group subjects averaged 49.6 minutes. As the agent group spent more time with the system, they attempted and solved more problems.

The agent group took fewer attempts to solve problems (30.9 compared to 32.56 attempt needed by the control group). In order to establish whether the knowledge level of the students may have affected this, we looked at the proportion of problems that were solved in the first attempt and found them to be similar for both the groups (5.1 for the agent group and 4.56 for the control group). This finding was consistent with our expectations, as the students did not get any direct help from the system before submitting initial solutions. Therefore, the students in both groups have comparable knowledge of SQL (this is also justified by the pre-test performance, discussed in section 5.4). Furthermore, students in both groups required a similar number of attempts to solve problems that could not be solved in the first attempt (when problem-specific hints were provided). The number of problems successfully solved per unit of time was similar for both groups. Students who used the agent recorded on average 0.27 correct answers per minute and the control group managed 0.22.

	Mean		Standard dev.	
	Agent	Control	Agent	Control
Total interaction time (mins)	55.90	49.63	17.30	26.70
No. of attempted problems	14.00	11.56	5.27	6.49
No. of solved problems	11.60	10.94	4.35	6.36
Total no. of attempts to solve the problems	30.90	32.56	14.13	23.97
Problems solved in the first attempt	5.10	4.56	2.60	2.73
Problems solved per time (problem/min)	0.22	0.27	0.07	0.21
Attempts to solve problems that could not be solved in the first attempt (attempts/problem)	2.90	2.91	1.61	1.34

Table 2: Means of interaction analyses

The average number of attempts taken to solve problems that were not solved in the first attempt was very similar: the agent group required 2.90 and the control group 2.91 attempts. As both versions of the system offered the same feedback, students from both groups required the same number of attempts.

In order to establish the effect of the agent on the student's learning over time, we plotted the average number of attempts taken to solve the i^{th} problem for each group. To reduce individual bias, the problems solved by less than 50% of the participating population were discarded (Fig. 3). Although no substantial trends can be seen, the agent group required 0.2 fewer attempts to solve each problem than the control group.

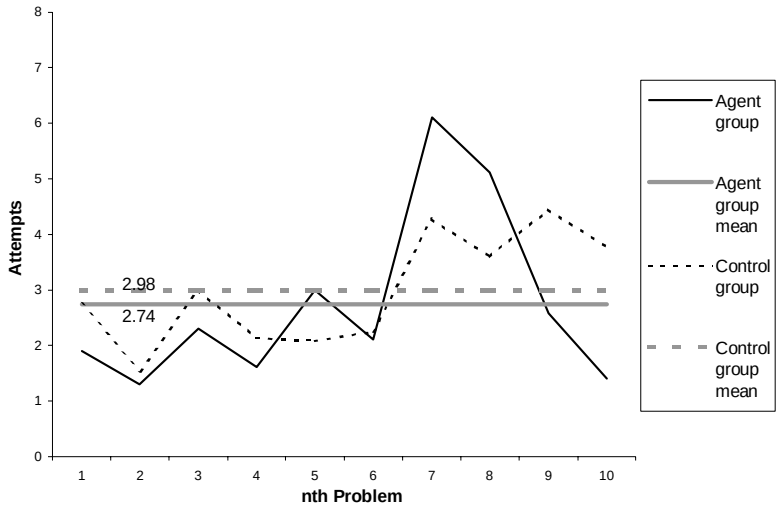

Fig. 3: The mean number of attempts taken to solve the i^{th} problem

5.4 Pre- and Post-Tests

Pre- and post-tests consisted of three multi-choice questions each, of comparable complexity. The marks allocated to questions were 1, 5 and 1 respectively. Nine out of ten students in the agent group and fourteen out of sixteen in the control group submitted valid pre-tests, the results of which are given in Table 3. The mean scores in the pre-test for the two groups are very close, suggesting that the two groups contained students of comparable knowledge.

Although participation in the pre-test was high, only four students from both groups sat the post-test[1]. Three of these students had used the agent, and a definite increase in their performance and confidence can be seen from the results of the post-test (4.33 and 2 for the agent and control group respectively). However, as the numbers involved are small, unbiased comparisons on the mean performances cannot be made.

Question	Agent group	Control group
1	0.33	0.14
2	2.56	2.50
3	0.67	0.71
Total	3.56	3.36

Table 3. Means for the pre-test

6 Discussion and Future Work

This paper presented SmartEgg, an animated pedagogical agent for SQLT-Web, an intelligent SQL tutor on the Web. Previous works on pedagogical agents have shown that they significantly increase motivation, resulting in longer interaction times and higher quality of learning.

In contrast to other discussed pedagogical agents, which required large teams of animators, pedagogues and programmers, SmartEgg was developed by a team of two people in a short period of time. Our initial hypothesis was that even a very simple agent would reveal the persona effect. In order to test the hypothesis, we performed an initial evaluation study in which two groups of students interacted with SQLT-Web and SmartEgg in a two-hour session. The students sat pre- and post-tests; all their actions were logged and finally the students filled a user questionnaire. Various analyses of the data collected in the evaluation study were performed, which showed a significant increase of motivation in the agent group. The students who interacted with the agent spent more time with the system, and solved more problems in fewer attempts than the students in the control group. We acknowledge the low number of students involved in the study, and will perform a much wider study to confirm the results from this initial evaluation.

[1] Some students did not log off properly, and have not even seen the post-test, which was administered on a separate Web page.

At the moment, SmartEgg provides textual information only. We plan to add verbal comments in the next phase, as it has been shown that more expressive agents are perceived to have greater utility and clarity [6]. Also, we plan to develop dynamic generation of behaviours. The behaviours would depend on the context of the feedback message, thus enabling SmartEgg to make a higher impact on students. Another future plan includes using the agent to provide support for self-explanation. This support would be in terms of dialogues with a student, where the agent prompts questions to guide the student.

Acknowledgements

This work was supported partly by the University of Canterbury research grant U6242. We are grateful to the Centre for Advanced Research in Technology for Education (CARTE) for providing the source code for the animation toolkit of Adele. We appreciated the stimulating environment in ICTG and the comments of its members. Our thanks go to Nenad Govedarovic for the initial drawing of SmartEgg, and the COSC205 students for their time and suggestions.

References

1. Alpert, S., Singley, M., Fairweather, P. Deploying Intelligent Tutors on the Web: an Architecture and an Example. *Int. J. AI in Education*, 10 (1999) 183-197.
2. Andre, E., Rist, T., Muller, J. WebPersona: a Life-Like Presentation Agent for Educational Applications on the WWW (1997). P. Brusilovsky, K. Nakabayashi, S. Ritter (eds) *Proceedings of workshop on Intelligent Educational Systems on the WWW, AI-ED'97.*
3. Andre, E., Rist, T., Muller, J. WebPersona: a Life-Like Presentation Agent for the World-Wide Web. (1998). *Knowledge-based Systems*, 11(1) (1998), 25-36.
4. Johnson, W.L. Pedagogical Agents. Invited paper, *ICCE'99* (1999).
5. Johnson, W.L., Shaw, E., Ganeshan, R. Pedagogical Agents on the Web. *Workshop on WWW-based Tutoring, ITS'98* (1998).
6. Lester, J., Converse, S., Kahler, S., Barlow, S., Stone, B., Bhogal, R. The persona effect: Affective Impact of Animated Pedagogical Agents, *Proc. CHI'97* (1997) 359-366.
7. Lester, J., Towns, S., FitzGerald, P. Achieving Affective Impact: Visual Emotive Communication in Lifelike Pedagogical Agents (1999). *Int. J. AI in Education.* 10 (1999).
8. Mallery, J.C. A Common LISP Hypermedia Server. *Proc. 1st Int. Conf. On the World Wide Web* (1994).
9. Mitrovic, A. A Knowledge-Based Teaching System for SQL. *Proc. ED-MEDIA'98,* T. Ottmann, I. Tomek (eds.) (1998) 1027-1032.
10. Mitrovic, A. Experiences in Implementing Constraint-Based Modeling in SQL-Tutor. *Proc. ITS'98* (1998) 414-423.
11. Mitrovic, A., Ohlsson, S. Evaluation of a constraint-based tutor for a database language, *Int. J. Artificial Intelligence in Education*, 10 (3-4) (1999).
12. Mitrovic, A., Hausler, K. An Intelligent SQL Tutor on the Web. Tech. Report TR-COSC 04/99, Computer Science Department, University of Canterbury (1999).
13. Ohlsson, S.: Constraint--based Student Modeling. In: Greer, J.E., McCalla, G.I. (eds.): *Student Modeling: the Key to Individualized Knowledge--based Instruction*. NATO ASI Series, Vol. 125. Springer-Verlag, (1994) 167-189.