Jacob - An Animated Instruction Agent in Virtual Reality

Marc Evers[1], Anton Nijholt[1]

[1] University of Twente, Department of Computer Science
P.O. Box 217, 7500 AE Enschede, The Netherlands
{evers, anijholt}@cs.utwente.nl

Abstract. This paper gives an overview of the Jacob project. This project involves the construction of a 3D virtual environment where an animated human-like agent called Jacob gives instruction to the user. The project investigates virtual reality techniques and focuses on three issues: the software engineering aspects of building a virtual reality system, the integration of natural language interaction and other interaction modalities, and the use of agent technology. The Jacob agent complies with the H-Anim standard. It has been given a task model and an instruction model in order to teach the user a particular task. The results of the project can be generalised so that the agent can be used to instruct other tasks in other virtual environments.

1. Introduction

The Jacob project investigates the application of virtual reality techniques and involves the design and construction of an animated agent in a 3-dimensional virtual environment. The agent is called Jacob and provides instruction and assistance for tasks that the user has to learn to perform in a virtual environment. The user interacts with Jacob by performing actions as well as by using natural language. The use of a lifelike agent in an interactive learning environment has a strong positive impact on students, which is shown by an empirical study performed by Lester et al. Such an agent can increase both the learning performance and the student's motivation [8].

The Jacob project involves an integration of knowledge from different disciplines, like intelligent tutoring systems, virtual reality, intelligent agent technology, natural language processing, and agent visualisation and animation techniques. It is a pilot project of the VR-Valley Twente Foundation, which aims at establishing a regional knowledge centre on virtual reality in the Netherlands. Two other pilot projects we would like to mention are AneuRx and Digimap. AneuRx aims at creating a medical training system in virtual reality for a specific type of surgery. Digimap aims at creating a 3D map of the region of Twente, which can be used to visualise and navigate through proposed changes in the environment. In the future, Jacob could play a role in these projects. More information about VR-Valley Twente can be found at http://www.vr-valley.com/.

Jacob is also closely related to the Virtual Music Centre (VMC), an ongoing project of our research group [12]. The VMC is a model of an existing music theatre in virtual reality. It contains agents that provide information about performances, that can handle theatre bookings, and that assist the user in navigating through the building. Eventually, the Jacob agent will be integrated in the VMC.

We will first describe the objectives and the approach of the Jacob project; then we will give an overview of the current state of the Jacob system. Related work and future research will be discussed. Last, we present some conclusions.

2.   Objectives

The research focus of the Jacob project is the use of virtual reality techniques and the design and implementation of virtual reality based systems. Important questions addrexssed in this project are:

- Can traditional software engineering and HCI technology be applied for designing and building a virtual reality system or does it require different technology?
- How can different interaction modalities like natural language, gestures, gaze, and manipulation of objects be integrated in a task oriented virtual reality system?
- How can agent technology be used in virtual reality systems?

To support the research activities, a prototype system is being developed that has to meet a number of requirements. First, the interaction between the user and Jacob should be multimodal. An important issue is how natural language dialogue and non-verbal actions are to be integrated and what knowledge of the virtual environment is needed for this purpose. Second, the Jacob agent should behave in an intelligent way, helping the user proactively and learning from the interaction. Third, visualisation of Jacob plays an important role, including natural animation of the body, generation of facial expressions, and synchronisation of lip movement and speech. Fourth, both the user and Jacob should be able to manipulate objects in the virtual environment, e.g. by using a dataglove. Fifth, there should be interaction between multiple agents in the virtual environment. Jacob should be able to communicate with other (artificial) agents and multiple users.

As the problem domain for the project, we have selected instruction of tasks in the virtual environment. This concerns tasks that consist of manipulating objects in the virtual environment, e.g. moving an object, pressing a button, or pulling a lever.

3.   Approach

We use existing knowledge, theories, and frameworks from different areas like intelligent tutoring systems [2] [15], computer graphics [11], and multi agent technology [17].

Software engineering plays a prominent role in the Jacob project. We apply object oriented techniques [3], design patterns [4], and software architecture knowledge [14]. We are investigating how to design and build such a virtual reality system in a maintainable and adaptable way.

For the current version of the prototype, we have applied the following technology: the basic virtual environment and the Jacob agent have been defined using VRML 2.0 (Virtual Reality Markup Language). The 'intelligent' part of the system has been written using the Java 1.1 programming language. The Java part is linked to the VRML world through the external authoring interface (EAI). The system runs in a web browser with a VRML plug-in. In this way, the Jacob system is highly portable and can be executed on a regular (powerful) PC or workstation.

4. Current State of the Project

In the current version of the prototype, Jacob teaches the user the Towers of Hanoi game, a classic toy problem from the field of artificial intelligence. In this game, the user has to move a stack of blocks from one peg to another using an auxiliary peg. The user can move single blocks from peg to peg; it is not allowed to place a larger block on top of a smaller one. Figure 1 shows a screenshot of the Jacob agent moving a block in the Towers of Hanoi.

Fig. 1. Screenshot of the Jacob prototype

Figure 2 depicts the software architecture of the system. We have applied a layered architecture to separate the concerns of the 3D visualisation from the basic functionality of the system. The concrete 3D world layer consists of a hierarchical structure of VRML nodes, like transformation nodes, geometry nodes, and sensor nodes. The example node objects in the figure depict this.

The abstract 3D world contains objects representing e.g. blocks, pegs, and Jacob's physical manifestation. Each user is represented by an Avatar object. This abstract 3D world layer also provides simulation of physical properties. For the

Towers of Hanoi, we have implemented basic collision avoidance so that objects cannot be moved through each other. Furthermore, we have implemented a simple gravity variant that makes objects fall at a constant speed.
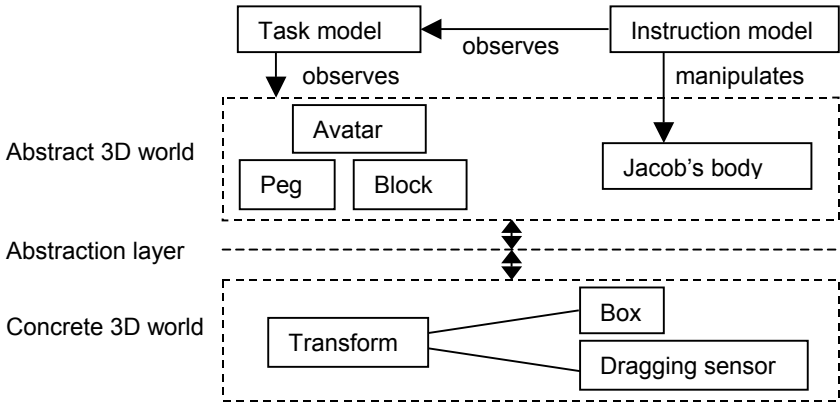

Fig. 2. Software architecture of the Jacob system

We have defined an interface layer between the abstract and concrete 3D world layers to make the system more robust to changes. This is necessary because standards like EAI are under development and are subject to change. This interface layer exposes only the essential properties of the nodes in the concrete 3D world, like the position of an object. All other details are hidden.

The task model and the instruction model together form Jacob's mind. They act as controllers that observe the abstract world and try to reach specific instruction objectives by manipulating Jacob's body.

The task model encapsulates knowledge about the task and the performance of the task: which objects are relevant for the task, how should the task be performed, and what errors can the user make. The task model also specifies why specific steps and actions should be taken. It observes the events and actions that happen in the world to keep track of the current state of the task.

The instruction model encapsulates instruction knowledge. This knowledge is generic and independent of the specific task. It includes adaptation of the instruction to the user, giving feedback on the user's actions, helping the user when he/she gets stuck, giving explanations, and showing the next step. An example of an instruction model is one where Jacob lets the user find his/her way in solving the Towers of Hanoi task while he occasionally gives feedback and answers questions from the user. For users that need more guidance, an instruction model can be defined where Jacob takes initiative and directs the user towards the solution of the task.

To keep track of the user's characteristics and his/her performance on the task, a student model is needed. Because of time limitations we have decided to leave this model out of the prototype.

The visualisation of Jacob's body has been created to comply with the H-Anim standard [10]. H-Anim is a standard for describing a humanoid body in terms of joints and segments. The use of this standard makes it relatively easy to plug in a

different body for Jacob. A small number of simple animations like walking and jumping have been defined for Jacob. These animations can work with any agent body that complies with the H-Anim standard. Note that there is not yet a standard for specifying animations for H-Anim agents.

5. Related Work

We will briefly describe the following projects: STEVE, WhizLow, PPP Persona, Jack, and AutoTutor.

In the STEVE project, an animated, pedagogical agent called STEVE gives instruction in procedural tasks in an immersive virtual environment. The STEVE environment can involve multiple users and multiple STEVE agents in the same task [13]. STEVE is similar to Jacob, but it has limited natural language interaction and focuses less on software engineering aspects.

For the IntelliMedia Initiative of the North Carolina State University three animated pedagogical agents have been developed, of which the WhizLow project involves a 3D virtual environment [9]. The WhizLow agent is an animated, lifelike agent in a virtual environment that provides instruction about computer architecture. The virtual environment consists of a 3D representation of concepts like CPU and buses. The student specifies a program and the agent shows and explains how the program is carried out by the different components in the world. There is no direct interaction with or manipulation of objects in the virtual environment.

PPP Persona (Personalized Plan-based Presenter) is a project of the German Research Center for Artificial Intelligence (DFKI), where an animated agent (called PPP Persona) provides instruction and presents information to users using the PPP system [1]. An agent creates a hierarchical plan how to present certain multimedia information to the user, taking into account e.g. characteristics of the user. A PPP Persona is represented in 2D and does not offer multimodal interaction.

Jack is a software system for controlling and animating human-like figures in an interactive, 3D environment. It has been developed at the Center for Human Modeling and Simulation at the University of Pennsylvania. It provides collision detection and avoidance, realistic grasping of objects, realistic graphics and visualisation. A commercial version of the Jack software is currently available from Transom Technologies Inc. (part of Engineering Animation Inc.) This commercial version of Jack aims at helping product designers to test their products in a virtual environment [16]. It enables users to define a virtual environment populated by virtual humans. Constraints can be used to specify the interaction between the virtual humans and the environment. Tasks can be defined for the virtual humans. Finally, the performance of the virtual humans can be analysed. It is for example possible to determine whether the virtual human can see or reach certain objects.

A system based on Jack is SodaJack, an architecture for animated agents that search for objects and manipulate these objects [5]. This architecture uses three hierarchical planners to locate objects and manipulate them. These planners produce low-level motion directives that are executed by the underlying Jack system. Both

Jack and SodaJack have realistically animated agents in a 3D virtual environment, but they are not interactive like Jacob.

AutoTutor is an intelligent tutoring system developed by the Tutoring Research Group of the University of Memphis. AutoTutor uses natural language dialogues for tutoring. The dialogue is delivered using an animated agent, specifically a talking head [6].

6. Future Research

We are currently working on the natural language interaction part of the Jacob system. We will restrict ourselves to keyboard input, but the intention is to add speech recognition and speech synthesis later on in the project. Natural language interaction involves parsing and interpretation of the user's utterances using an appropriate knowledge representation of the virtual environment. Jacob will respond by producing utterances or performing actions. For generation of Jacob's utterances, annotated templates will be used. The natural language dialogue manager and the instruction model are closely related and will be integrated.

To determine the lexicon and grammar for the user utterances, we will perform Wizard of Oz experiments [7]. In a Wizard of Oz experiment, users work with the Jacob system in an experimental setting where Jacob and Jacob's responses are controlled by a human. The dialogues are logged and analysed. We have already constructed a distributed version of the Jacob system where Jacob can be controlled from a remote system. Wizard of Oz experiments can also be useful to find out what kind of actions and nonverbal behaviour to expect from users.

The animation for Jacob should be made more natural. Jacob should also show facial expressions, like anger and surprise, as a response to utterances or actions of the user.

The Towers of Hanoi game is quite restricted as an example task. As a result, the task and instruction models are currently restricted as well. We will extend and generalise the Jacob system to other tasks and analyse the impact on the task and instruction models. We are investigating formalisms for representing these models. We will integrate Jacob in the Virtual Music Centre mentioned in the introduction. There, Jacob can give instruction for tasks like navigating through the VMC (to teach the user what to do and to find where) and operating the lights of the theatre.

We will also investigate the use of a dataglove. A dataglove allows the user to manipulate objects in the virtual environment directly. It can also be used for pointing at objects and for making gestures.

7. Conclusions

Although the current version of the Jacob system is restricted, we think that the results can be generalised very well. The concept of instruction and assistance is very generic and can be applied to various tasks and situations.

Furthermore, the use of stable knowledge from several fields like intelligent tutoring systems makes the components of the system quite generic.

The use of software engineering knowledge and techniques improves the flexibility of the Jacob system. The concepts of such a virtual reality system map well onto an object oriented model. We do not yet have results on the applicability of agent technology and the issues of multimodal interaction.

More information about the project, including the latest results and the prototype, can be found at: http://www.cs.utwente.nl/~evers/jacob

References

1. André, E., Rist, T., Müller, J.: WebPersona: A Life-Like Presentation Agent for the World-Wide Web. Knowledge-based Systems **11**(1) (1998) 25-36
2. Bonar, J., Cunningham, R., Schultz, J.: An Object-Oriented Architecture for Intelligent Tutoring Systems. In: OOPSLA'86 Proceedings (1986) 269-276
3. Booch, G., Rumbaugh, J., Jacobson, I.: The Unified Modeling Language User Guide. Addison Wesley (1999)
4. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns - Elements of Reusable Object-Oriented Software. Addison-Wesley (1995)
5. Geib, C., Levison, L., Moore, M.B.: SodaJack: an architecture for agents that search for and manipulate objects. Technical Report MS-CIS-94-16 / CINC-LAB 265. Dept. of Computer and Information Science, University of Pennsylvania (1994)
6. Graesser, A.C., Wiemer-Hastings, K., Wiemer-Hastings P., Kreuz, R.: AutoTutor: A simulation of a human tutor. Journal of Cognitive Systems Research **1** (1999) 35-51
7. Hoeven, G.F. v.d., Andernach, J.A., Burgt, S.P. v.d., Kruijff, G-J.M., Nijholt, A., Schaake, J., De Jong, F.M.G.: SCHISMA: A Natural Language Accessible Theatre Information and Booking System. In: Proc. First International Workshop on Applications of Natural Language to Data Bases (1995) 271-285
8. Lester, J.C., Converse, S.A., Kahler, S.E., Barlow, S.T., Stone, B.A., Bhogal, R.S.: The persona effect: affective impact of animated pedagogical agents. In: CHI'97 Proceedings (1997) 359-366
9. Lester, J.C., Zettlemoyer, L.S., Grégoire, J.P., Bares, W.H.: Explanatory Lifelike Avatars: Performing User-Centered Tasks in 3D Learning Environments. In: Agents'99, Proceedings of the Third International Conference on Autonomous Agents, ACM Press (1999) 24-31
10. Humanoid Animation Working Group: Specification for a Standard Humanoid Version 1.1. http://ece.uwaterloo.ca/~h-anim/spec1.1/ (August 1999)
11. Lin, M., Manocha, D., Cohen, J., Gottschalk, S.: Collision Detection: Algorithms and Applications. In: Laumond, J.P., Overmars, M., Peters, A.K. (eds.): Algorithms for Robotics Motion and Manipulation (Proc. of 1996 Workshop on the Algorithmic Foundations of Robotics) (1996) 129-142
12. Nijholt, A., Hulstijn, J.: Multimodal Interactions with Agents in Virtual Worlds. In: Kasabov, N. (ed.): Future Directions for Intelligent Information Systems and Information Science, Physica-Verlag: Studies in Fuzziness and Soft Computing (2000)
13. Rickel, J., Johnson, W.L.: Animated Agents for Procedural Training in Virtual Reality: Perception, Cognition, and Motor Control. Applied Artificial Intelligence **13** (1999) 343-382
14. Shaw, M., Garlan, D.: Software Architecture - Perspectives on an Emerging Discipline. Prentice Hall (1996)
15. Tekinerdogan, B., Krammer, H.P.M.: Design of a Modular Composable Tutoring Shell for Imperative Programming Languages. In: Proceedings of the International Conference on Computers in Education (1995) 356-363

16. Transom Technologies: Jack Software. http://www.transom.com/Public/transomjack.html
17. Weiss, G. (ed.): Multiagent Systems. The MIT Press (1999)
18. Zhang, D.M., Alem, L., Yacef, K.: Using Multi-Agent Approach for the Design of an Intelligent Learning Environment. In: Wobcke, W., Pagnucco, M., Zhang, C. (eds.): Agents and Multi-agent Systems. Lecture Notes in Artificial Intelligence, Vol. 1441. Springer-Verlag, Berlin Heidelberg New York (1998) 220-230