In M. Stumptner, D. Corbett, & M. Brooks (Eds.), AI 2001: Advances in Artificial Intelligence. Proceedings of the 14th Australian Joint Conference on Artificial Intelligence. 368-377. Berlin: Springer.

Embodying the JACK Agent Architecture

Emma Norling[1] and Frank E. Ritter[2]

[1] Computer Science and Software Engineering
The University of Melbourne
E.Norling@csse.unimelb.edu.au
[2] School of Information Sciences and Technology
The Pennsylvania State University
ritter@ist.psu.edu

Abstract. Agent-based models of human operators rarely include explicit representations of the timing and accuracy of perception and action, although their accuracy is sometimes implicitly modelled by including random noise for observations and actions. In many situations though, the timing and accuracy of the person's perception and action significantly influence their overall performance on a task. Recently many cognitive architectures have been extended to include perceptual/motor capabilities, making them embodied, and they have since been successfully used to test and compare interface designs. This paper describes the implementation of a similar perceptual/motor system that uses and extends the JACK agent language. The resulting embodied architecture has been used to compare GUIs representing telephones, but has been designed to interact with any mouse-driven Java interface. The results clearly indicate the impact of poor design on performance, with the agent taking longer to perform the task on the more poorly designed telephone. Initial comparisons with human data show a close match, and more detailed comparisons are underway.

1 Introduction

Although it is difficult to find a definition of a software agent that all researchers will agree upon, one aspect that seems to be universally accepted is that an agent is *situated* — it operates within an environment that it senses in some way, and in which its actions are performed. Despite this agreement on the importance of being situated, when it comes to using software agents to model human operators the details of perception and action are too often ignored.

In many cases, software agents are simply given perfect vision, able to see all objects within their field of vision equally clearly, and precise action, with every action being completely accurate and instantaneous. For some types of simulation, these simplifications may have little impact on the results, but in many applications the effects can be significant. In human-computer interaction, for example, the time taken to find an object on the display and move the mouse to this object can be significant in the overall timing of the task, even for experts. In a driving simulation, the accuracy and speed of steering might make

Norling, E., & Ritter, F. E. (2001). Embodying the JACK agent architecture. In AI 2001: Advances in Artificial Intelligence. Proceedings of the 14th Australian Joint Conference on Artificial Intelligence, 368-377. Springer: Berlin.

the difference between safe driving and an accident. As Gray discusses [9], small differences in interface design can have a significant impact on the time taken to perform common tasks.

Perceptual/motor extensions to cognitive architectures, most notably ACT-R/PM [4], have allowed researchers to build models that interact with simulations of the interfaces an operator would use (such as Salvucci's work on telephone use while driving [15]), and in some cases with the interface itself (e.g. the work of Byrne [6, 5] and Amant and Riedl [2] on user interface evaluation). The growing interest in this approach is illustrated in a recent special edition of the *International Journal of Human-Computer Studies* [14]. Although a significant amount of work has focused on GUI testing and evaluation, there are also models which manipulate (simulations of) physical objects, e.g. [11, 15]. These studies all illustrate the importance of including perception and action in the model in order to get a better match between the model and the operator being modelled.

This paper describes an implementation of an initial set of functional perceptual/motor capabilities with the JACK agent language [1]. An agent with these capabilities was used to compare graphical representations of telephone interfaces, such as those shown in Fig. 1. Although we limited the motor capabilities to simple mouse movement and clicking (this is all that was needed for the interface), the addition of further motor abilities will now be straightforward. These capabilities will be particularly useful in the JACK agent language because it is designed for modelling human operators. These capabilities allow a more complete model of the operator.

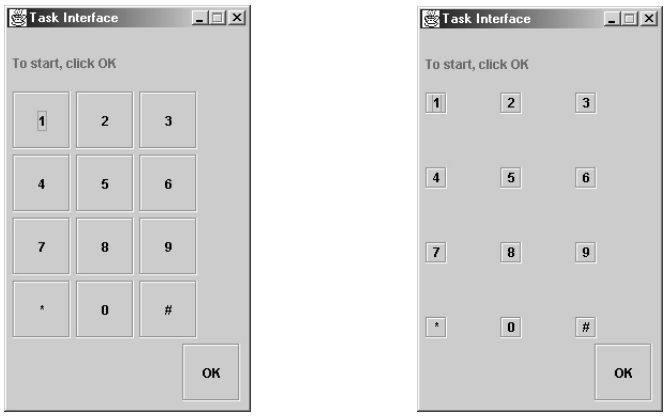

Fig. 1. Two sample interfaces with which the agent and human can interact

In the remainder of this paper, we first discuss perception and action from the perspective of interaction with these GUIs, and then discuss our implementation of perception and action using JACK. We present the results showing the impact of simple good and bad GUI designs on agent performance, and some preliminary

work showing that the embodied JACK agent's performance predicts the human performance on the same interfaces. From these results, we note how including perceptual/motor components helps to model human operators, and that similar effects will influence models of human operators in other types of environments.

2 Interaction with Example Interfaces

The interfaces in Fig. 1 require only simple interaction: reading the instruction at the top of the window, performing the appropriate sequence of mouse clicks (which always ends with clicking "OK"), then getting the next instruction, and repeating this loop until "Finished" appears in the instruction area. The interface does not require any keyboard input, nor does it include any complex mouse navigation, such as pull-down menus. For a description of how keyboard interaction could be included in the model, see the work of Baxter, Ritter and their colleagues [3, 13] or John [10].

2.1 Visual perception

The model of visual perception added to the agent corresponds to the three regions people have in their field of view. The first of these is the fovea, a narrow region approximately two degrees in diameter, which is the region of greatest visual acuity. The next is the parafovea, which extends approximately five degrees beyond the fovea, and provides somewhat less acuity. For example, if a button lay in the parafovea, the operator would probably see the shape, size and location of the button, but not recognise the label on it. The remainder of the field of view is known as peripheral vision. Perception in this area is extremely limited — the operator would probably see that an object was there, but not be able to pinpoint its exact location without shifting focus. (This is a necessary but gross set of simplifications. There are many more subtleties and regularities.)

Because of these limitations, an operator will not be able to clearly perceive the entire interface simply by looking at a single point on it. The eye will have to shift focus in order to perceive the different objects on the display. This is achieved through saccadic eye movements, during which the eye effectively "jumps" from one focus to another. For saccades less than 30° (which covers all saccades with our interface), the "jump" takes about 30 ms, during which the operator is effectively blind, followed by a longer fixation on the new focus, typically of about 230 ms duration [7].

These capabilities and limitations allow models in JACK to find information on interfaces, but they require effort. The model must know where to look, or it must search: it must move its eye, and it must then process what it sees. These efforts take time and knowledge, corresponding to similar time and knowledge that the operator has.

2.2 Manual input

The only manual input required for this interface is mouse movement and clicking. Mouse movements by operators are not accurate, relying heavily on visual feedback. Rather than moving in a single linear motion to the object, the human operator will move the mouse in a series of shorter segments, with a correction at the end of each one, until the mouse pointer appears on the target. Studies have shown that each of these segments has length approximately $1 - \epsilon d$, where d is the remaining distance to the centre of the target, and ϵ is a constant 0.07 [7], p. 53. Each of these movements takes roughly the same amount of time, approximately 70 ms, plus a delay for visual feedback, before the next movement begins. This means that although the final position of the mouse will be within the target, it will not necessarily be at the centre of the target, as shown in Fig. 2.

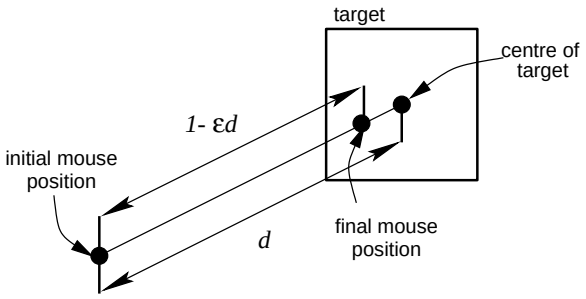

Fig. 2. Moving the mouse to a target

Of course, because of error in the movement, the distance will not be exactly $1 - \epsilon d$, nor will the point lie directly on the line between the origin and the centre of the target. Unfortunately, as discussed by MacKenzie et al [12], while there have been many studies which report error *rates* when using a mouse or other pointing device, very few report the types or magnitudes of errors. We have extrapolated from the results of MacKenzie et al to get a mean variability in final position that is equal to 5% of the distance travelled — further studies are required to confirm this figure.

3 Implementation

The implementation of the system consists of two parts: a simple GUI that was used for testing purposes, and the embodied JACK agent that interacts with this GUI.

3.1 Interface

The telephone GUI was written in Java 1.3 using Swing components. The user can specify the size of the telephone buttons and the spacing between the buttons as command line arguments. A transparent pane overlays this GUI, and it is via this pane that the agent interacts with the GUI. When the agent "looks" at the GUI, the pane returns the details of objects in the fovea and parafovea. When the agent moves or clicks the mouse, the pane passes this information to the GUI. The eye position of the agent is displayed on the pane, as well as the current position of the agent's mouse pointer. Although the agent was tested only using the telephone GUIs, it is designed to interact with any mouse-driven GUI written in Java, by overlaying this same pane.

A control panel is also provided to control the agent and test the interface. (See Fig. 3.) This allows the user to adjust the fovea and parafovea size, switch between a crosshair display for eye position or a full indication of fovea and parafovea boundaries, manually control the eye and mouse positions (for testing purposes), disable the controls completely (to interact directly with the telephone), and create or destroy the agent. The objects that the agent can see (both in the fovea and parafovea) are displayed on the control panel, as well as the actions that have been performed so far.

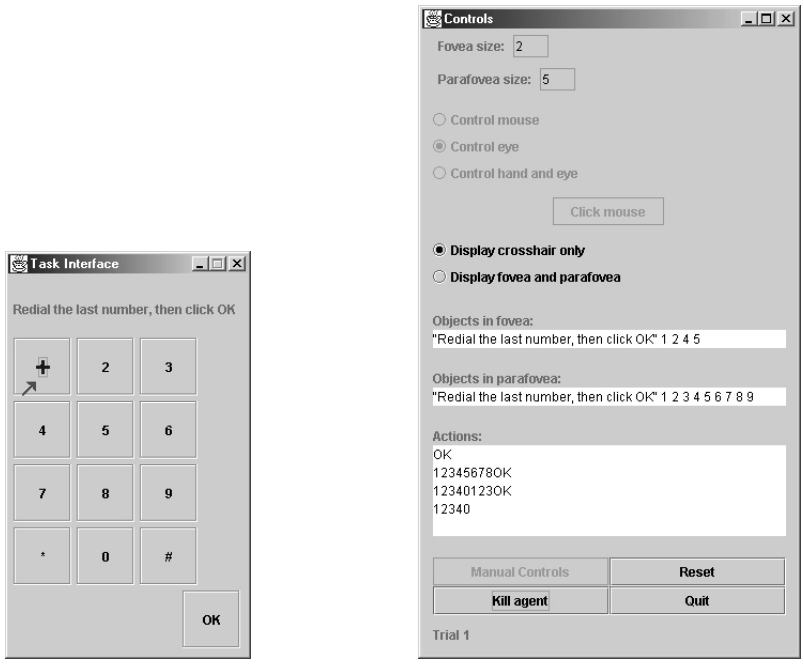

Fig. 3. An agent interacting with the interface, and the associated control panel

3.2 Agent

The agent was written in the JACK agent language, a language implementing a BDI (beliefs-desires-intentions) architecture as an extension to the Java programming language. Other than the perception and action capabilities, the agent is extremely simple, with just two plans: one that interprets the instructions and another that dials a number (retrieving it from memory).

Interaction with the GUI is provided through two capabilities: a vision capability which controls eye position and fixations, and an action capability which controls mouse position and clicks.

The vision capability can achieve three goals: to look at a particular object on the screen, to look at a particular position on the screen, and to simply observe at the current eye position, storing information to memory. The times for eye movements and fixations are included using JACK `@waitfor(`*time*`)` statements, so that the agent takes the appropriate time to achieve these goals.

Similarly, the actions capability can achieve a limited number of goals: to move the mouse to a particular point or object, to click on a particular object, and to click at the current mouse position. Timing to perform these tasks is incorporated for these actions as for the eye movements.

4 Comparison of Model Predictions and Human Data

For preliminary testing, we created a short sequence of tasks for both the agent and human operator to perform using our telephone interface. These tasks were displayed in the instruction section at the top of the interface. The instructions (in order) were:

- To start, click OK
- Dial home, then click OK
- Dial work, then click OK
- Redial the last number, then click OK
- Dial directory enquiries, then click OK
- Call your mother, then click OK

After this sequence, "Finished" appeared in the instruction section. The user was told in advance which numbers they would be asked to dial, and in one case "your girlfriend" was substituted for "your mother" because the user did not know that number — the aim was to use numbers that were known "by heart," so that the time to recall them was short and uniform.

Each of the three users was asked to perform this sequence of tasks 20 times — 10 for the interface with the "standard" size buttons, and 10 for the one with small, widely-spaced buttons. (The two interfaces in Fig. 1 show the scale of differences but are smaller than the real size of 5.5 cm by 9.5 cm.) Every user encountered the standard interface first. The users were instructed not to pause between starting a sequence of tasks and seeing "Finished" appear, but to take as long as they wished between sequences. The time was recorded each time the

user clicked "OK." The results for each user were then averaged over each group of 10 sequences.

The JACK agent performed the same sequences of tasks, but in this case there were 30 repetitions for each version of the interface, and these were averaged. Because the number dialled can have a significant impact on the performance of this task (e.g. "555" is dialled more quickly than "816"), the agent was compared against individual users, dialling the same numbers, rather than aggregating all users [8]. Figures 4 and 5 show results from two subjects — the first is the worst fit of all subjects, and the second is the best.

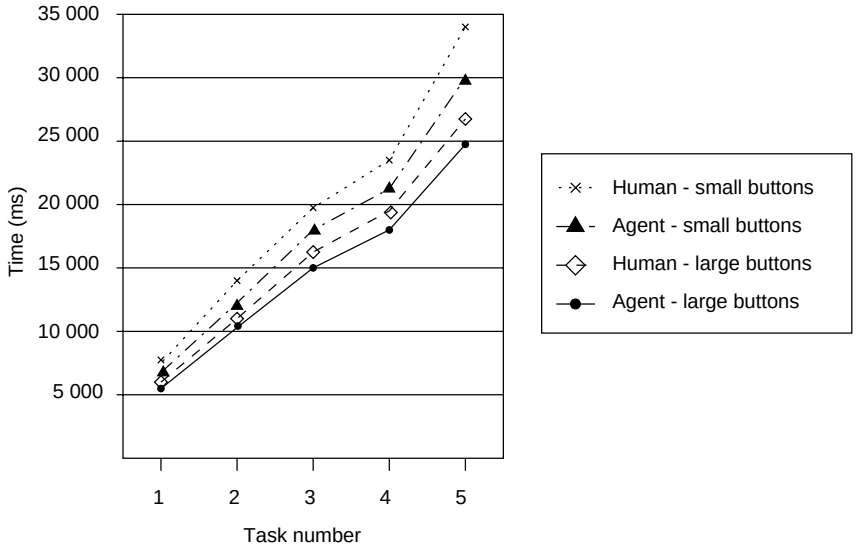

Fig. 4. Time taken to perform the sequence of tasks (the worst fit, subject 1)

In all cases, the time taken to perform the tasks was significantly lower for both the human and agent using the GUI with large buttons. The agent has a tendency to out-perform the human user on both interfaces, and we suspect that this is because the error that we introduce during mouse movement is too small. As mentioned previously, further studies are needed to get an accurate figure for the magnitude of the error. The raw data (not presented here) also shows more variation in the human timing than that of the agent, further reinforcing the suspicion that our error magnitude is too small.

These results are only preliminary results, and we have only used a very small sample of three users, but these results are extremely promising. We are now gathering more detailed human data, logging all mouse actions, and using an eye tracker to record eye movements, so that more detailed comparisons between the users and the model can be made. We will also collect more data

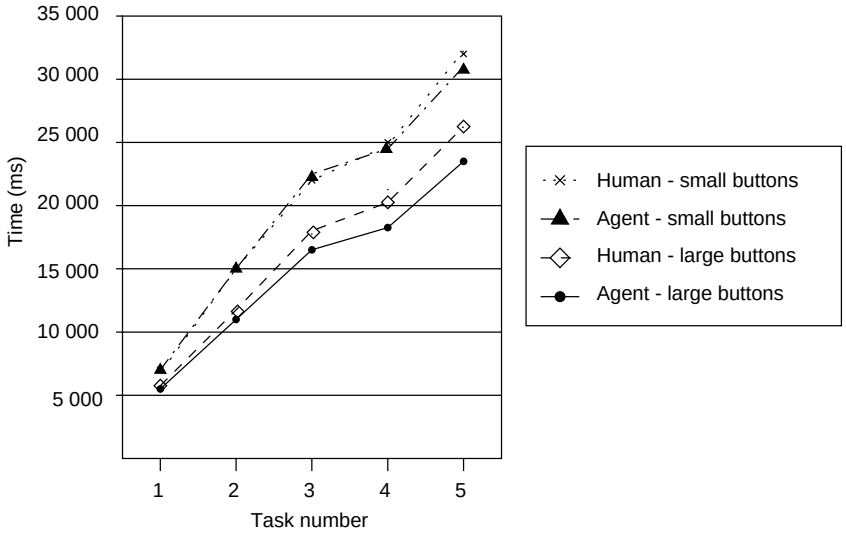

Fig. 5. Time taken to perform the sequence of tasks (the best fit, subject 3)

on the magnitude of errors in mouse movement. The detailed comparison will allow us to further validate the model.

5 Conclusion

The work presented here represents a first step in embodying a JACK agent, giving it the ability to interact with a GUI by "looking" at the interface, seeing it as a human would, and moving and clicking a mouse on the interface. As discussed, the early results are promising, and we expect the more detailed comparison with human users to further refine the model. Although we have focused on visual perception and mouse input, other modes of perception and action could be added in a similar fashion, using the vast wealth of human engineering data that has been collected over the years.

The initial results here clearly indicate the impact of a "bad" user interface design, with the agent taking significantly longer to perform the task on the bad interface (as did the human users). Our results suggest that an embodied agent of this type can be used to test user interfaces, in time eliminating much (though probably not all) of the costly user testing stage of GUI design.

Another application of an agent that is embodied in this way is in a simulation environment where the agent replaces a human operator, for example, in a training simulator. If the agent does not have accurate delays for its actions, or perceives the environment in an unrealistic manner, the value of the training may be questioned. The trainee may develop unrealistic expectations of their team members' abilities, or they may use tactics that would be unnecessary or inappropriate to beat a real world opponent.

The capabilities we have added to the JACK agent make it more situated, interacting with its environment in a manner that more closely matches the human operator being modelled. This embodiment of the agent gives more realistic performance, making the model suitable for a broad range of applications in which the timing and accuracy of perception and action will have a significant impact on the performance of the agent.

Acknowledgements

Support for this work was provided by a Strategic Partners with Industry Research and Technology (SPIRT) award from the Australian Research Council and Agent Oriented Software Pty Ltd, a Postgraduate Overseas Research Experience Scholarship (PORES) from the University of Melbourne, and the Space and Naval Warfare Systems Center San Diego. The content of the information does not necessarily reflect the position or policy of the United States government, and no official endorsement should be inferred.

References

1. Agent Oriented Software Pty. Ltd. JACK Intelligent Agents. http://agent-software.com.au/jack.html.
2. Robert St. Amant and Mark O. Riedl. A perception/action substrate for cognitive modeling in HCI. *International Journal of Human-Computer Studies*, 55:15–39, 2001.
3. Gordon D. Baxter and Frank E. Ritter. Designing abstract visual perceptual and motor action capabilities for use by cognitive models. Technical Report 36, ERSC Center for Research in Development, Instruction and Training, Department of Psychology, University of Nottingham, 1996.
4. Michael D. Byrne. The ACT-R/PM project. In Michael Freed, editor, *Simulating Human Agents: Papers from the 2000 Fall Symposium*, pages 1–3, North Falmouth, Massachusetts, November 2000.
5. Michael D. Byrne. ACT-R/PM and menu selection: Applying a cognitive architecture to HCI. *International Journal of Human-Computer Studies*, 55:41–84, 2001.
6. Michael D. Byrne, Scott D. Wood, Piyawadee Sukaviriya, James D. Foley, and David Kieras. Automating interface evaluation. In *Proceedings of the CHI'94 Conference on Human Factors in Computer Systems*, pages 232–237, New York, NY, 1994.
7. Stuart K. Card, Thomas P. Moran, and Allen Newell. *The Psychology of Human-Computer Interaction.* Lawrence Erlbaum Associates, 1983.
8. Fernand Gobet and Frank E. Ritter. Individual data analysis and unified theories of cognition: A methodological proposal. In N. Taatgen and J. Aasman, editors, *Proceedings of the Third International Conference on Cognitive Modelling*, pages 150–157, Groningen, Netherlands, March 2000.
9. Wayne D. Gray and Deborah A. Boehm-Davis. Milliseconds matter: An introduction to microstrategies and to their use in describing and predicting interactive behavior. *Journal of Experimental Psychology: Applied*, 6(4):322–335, 2000.

10. Bonnie E. John. TYPIST: A theory of performance in skilled typing. *Human-Computer Interaction*, 11:321–355, 1996.
11. Gary Jones and Frank E. Ritter. Over-estimating cognition time: The benefits of modelling interaction. In Michael Freed, editor, *Simulating Human Agents: Papers from the 2000 Fall Symposium*, pages 67–74, North Falmouth, Massachusetts, November 2000.
12. I. Scott MacKenzie, Tatu Kauppinen, and Miika Silfverberg. Accuracy measures for evaluating computer pointing devices. In *Proceedings of CHI 2001*, pages 9–16, Seattle, Washington, April 2001.
13. Frank E. Ritter, Gordon D. Baxter, Gary Jones, and Richard M. Young. Supporting cognitive models as users. *ACM Transactions on Computer-Human Interaction*, 7(2):141–173, 2000.
14. Frank E. Ritter and Richard M. Young. Embodied models at simulated users: Introduction to this special issue on using cognitive models to improve interface design. *International Journal of Human-Computer Studies*, 55:1–14, 2001.
15. Dario D. Salvucci. Predicting the effects of in-car interface use of driver performance: An integrated model approach. *International Journal of Human-Computer Studies*, 55:85–107, 2001.