Communicative Humanoids

A Computational Model of Psychosocial Dialogue Skills

Kristinn Rúnar Thórisson
B.A. *Cognitive Psychology,* University of Iceland, 1987
M.S. *Engineering Psychology,* Florida Institute of Technology, 1990

Submitted to the Program in Media Arts & Sciences,
School of Architecture & Planning,
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
at the ***Massachusetts Institute of Technology***

September 1996

Author: _____
Program in Media Arts & Sciences
July 19, 1996

Certified by: _____
Justine Cassell
Assistant Professor of Media Arts & Sciences
MIT Program in Media Arts & Sciences
Thesis Advisor

Accepted by: _____
Stephen A. Benton
Chair, Departmental Committee on Graduate Students
Program in Media Arts & Sciences

Communicative Humanoids

A Computational Model of Psychosocial Dialogue Skills

Kristinn Rúnar Thórisson

Submitted to the Program in Media Arts & Sciences,
School of Architecture & Planning on July 19, 1996
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy at the Massachusetts Institute of Technology

Abstract

Face-to-face interaction between people is generally effortless and effective. We exchange glances, take turns speaking and make facial and manual gestures to achieve the goals of the dialogue. Endowing computers with such an interaction style marks the beginning of a new era in our relationship with machines—one that relies on communication, social convention and dialogue skills. This thesis presents a computational model of psychosocial dialogue expertise, bridging between perceptual analysis of multimodal events and multimodal action generation, supporting the creation of interfaces that afford full-duplex, real-time face-to-face interaction between a human and autonomous computer characters. The architecture, called *Ymir*, has been implemented in software, and a prototype humanoid created. The humanoid, named *Gandalf*, commands a graphical model of the solar system, and can interact with people using speech, manual and facial gesture. Gandalf has been tested in interaction with users and has been shown capable of fluid face-to-face dialogue. The prototype demonstrates several new ideas in the creation of communicative computer agents, including *perceptual integration of multimodal events*, *distributed processing* and *decision making*, *layered input analysis* and *motor control*, and the integration of *reactive* and *reflective perception* and *action*. Applications of the work presented in this thesis can be expected in such diverse fields as education, psychological and social research, work environments, and entertainment.

Thesis Advisor:

Justine Cassell

Assistant Professor of Media Arts & Sciences
MIT Program in Media Arts & Sciences

This research was sponsored by the Media Laboratory and Thomson-CSF.

Doctoral Dissertation Committee

Thesis Advisor:

Justine Cassell
Assistant Professor of Media Arts & Sciences
M.I.T. Program in Media Arts & Sciences

Thesis Reader:

Pattie Maes
Associate Professor of Media Arts & Sciences
Sony Corporation Career Development Professor
of Media Arts & Sciences
M.I.T. Program in Media Arts & Sciences

Thesis Reader:

Steve Whittaker
Research Scientist
AT&T Bell Laboratories

Acknowledgments

A number of people helped shape and polish the ideas expressed in this thesis and they deserve credit. I want to thank Justine Cassell, for her unique perspective, professional advice, and for continued support through the completion of this thesis. I thank my readers for their recommendations, expertise and moral support; Pattie Maes, for her initial interest in this work, continued encouragement, and for feeding me with material that proved essential for its completion; Steve Whittaker for pointing out all those important details and for his keen observations from the psychological perspective. Thomas Malone, who was a member on my general exam committee, brought to my attention important issues in system architecture and design. Richard A. Bolt gave invaluable support in the beginning stages.

Thanks to the members of the Gesture & Narrative Language Group, my "new" gang: Jennifer Glos, Marina Umachi, Hannes Vilhjálmsson, Scott Prevost and Erin Panttaja, whose comments, suggestions and work has helped me shape this thesis, and who, by being there, made finishing up infinitely more fun.

Thanks to the students of the past who made life considerably more fun here at the lab, and some of whom I had the additional pleasure of working with on a project or two: Dave Berger, Brent Britton, Karen Donoghue, Tony Ezzat, Ed Herranz, Jen Jacoby, Matt Kaminsky, Alice Lei, Steve Librande, Mark Lucente, Bob Sabiston, David Small, Carl Sparrell, and Chris Wren.

My undergraduate research assistants, who collectively detailed the inner workings of ToonFace: Steven Levis, Roland Paul, Nimrod Warshawsky and Calvin Yuen.

Thanks to my office mate, Joshua Bers, who is always ready to discuss anything from the general to the particular, but most importantly my dedicated partner in all of the most wacky late-night madness generated during long hours of debugging. And to Tomoko Koda for livening up the place.

Thanks to the volunteers in my human subjects study.

Thanks to Bebba and Jóel for donating me the Sanyo Z1 cassette/CD player boom box. Couldn't have made without it!

Thanks to the cheerful crowd on the second floor, especially Linda Peterson, for help and understanding. To the syspro guys for always being on call.

Thanks to Nicholas for the cappuchino machine next to my office.

Thanks to my father, fiórir S. Gu›bergsson, and mother, Rúna Gísladóttir, for invaluable support and for providing me with the skills and will to enjoy my work.

But most of all, thanks to my wife Katrín Elvarsdóttir, whose dedication made it possible for me to do this work.

Time to do the laundry.

To

Katrín Elvarsdóttir

Introduction

1.

We are creating a new arena of human action: communication *with* machines rather than operation *of* machines.

—Card, Moran & Newell (1983, p. 7)

As humans, we are naturally endowed with multimodal input/output capabilities. Multimodal interactions happen between people most every day: we exchange glances, gesture to each other, speak and make facial expressions. The purpose of these interactions is usually to communicate certain information to, and receive information from others. As any student of psychology will know, multimodal I/O as it happens in face-to-face interaction is a complex phenomenon and many of its features and smaller pieces make valid research topics and research fields. Yet most people, when asked about how they manage to communicate complex information in a short face-to-face interaction, they shrug and reply "It's easy—getting a machine to do that should be trivial" (or even worse "Haven't they done that already?"). In a paper on computers and common sense, Phil Agre [1985, p. 72] writes:

> Playing chess is easy, but making breakfast is enormously complicated. This complexity stares us in the face every morning, yet it is invisible.

Face-to-face interaction is like making breakfast. It looks easy. But when it comes to making a computer do the same, things start getting mighty complicated.

Here, the approach taken to this problem is not in the typical tradition of divide-and-conquer, but instead to look at multimodal interaction holistically, with the purpose of constructing a computer system that can sustain and support such interactions with a human. To this end I have designed an architecture that allows for the construction of multimodal agents—agents that can interact with people using speech, gesture and gaze. I have also built a prototype agent in this architecture. These will be discussed in Chapters 7., 8. and 9. In this chapter we will define some important terms, take a close look at the goals of this work and give an overview of the rest of the thesis.

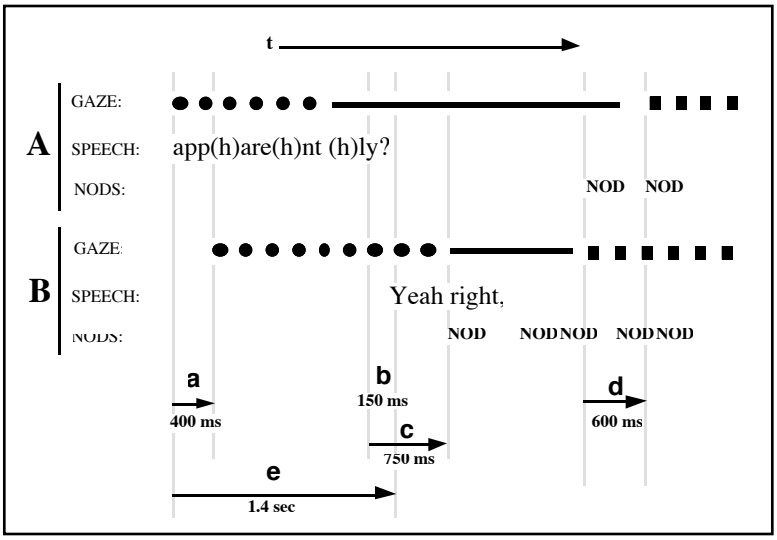


FIGURE 1-1. Transcript spanning 3 seconds of a typical two-person conversation, showing the timing of speech, gaze and head nods for each conversant (Adapted from Goodwin [1981]). "A brings her gaze to the recipient. B reacts to this by immediately bringing her own gaze to A. The two nod together and then ... withdraw from each other, occupying that withdrawal with a series of nods" [Goodwin 1981, p. 119]. Notice that a, b, c and d are listener reactions to speaker actions; these all happen under 1 second. b is a turn transition. e is the estimated minimum time the listener had for generating a response to the content of the speakers preceding turn.

Circles indicate gaze moving toward other, lines indicate a fixation on other, squares are withdrawal of gaze form other, question mark shows rising intonation.

1.1 What is Needed

The transcription in Figure 1-1 demonstrates the complex nature of face-to-face discourse [Goodwin 1981]. Here, rapid responses and more reflective ones are interwoven in a complex pattern. Person A and person B exchange glances that are timed to the decisecond; they give each other feedback and take turns speaking with admirable efficiency. People are obviously very good at doing this, and to date no computer system has been able to replace one of the participants and produce the same pattern as shown in the example. This is because a system that can do this needs to be responsive to the environment, yet be capable of longer-term planning. Moreover, it needs to keep track of multiple sources of information including a person's gaze, facial expression, gesture, intonation, body language, in addition to speech content.

To date, research has fallen short when it comes to these essential topics in face-to-face interaction:

1. *Continuous-input over multiple modes.*
2. *Integration of multimodal inputs.*
3. *Coordination of actions at multiple levels of granularity.*
4. *Bridging between sensory input and action output.*

Instead of a "vending-machine" interaction style (communicate all information ... wait for system response), continuous input allows a system to support interruptions, incremental input and incremental interpretation. Multimodal input contains multiple data types; these have to be integrated in some manner to support correct feedback generation. In dialogue, real-time responses are tightly coupled with more "reflective" ones; "um"s and "ahh"s are automatically inserted while we think of what to say. How we allow a machine to do this as output is also an open question. A complete bridging between sensory input and motor output is necessary if we want to have a platform that allows us to experiment with various designs for humanoid agents.

1.2 Goals of This Work

"Designing computers that are to operate in isolation is one thing, but designing computers that are to occupy an important place in the lives of real people is something else."

—Philip Agre (1994, p. 230)

This thesis describes the efforts of endowing a multimodal, on-screen computer agent with psychosocial dialogue skills aimed at supporting and sustaining dialogue with a human. Two closely related problems or issues are addressed by this work. The first is the general issue of human-computer interaction. The new type of interface proposed takes advantage of people's knowledge about face-to-face interaction, turn-taking and perceptual abilities of interacting parties to provide a consistent metaphor for the relationship between human and computer. By introducing a situated social entity into the human-computer relationship, enabling full-duplex multimodal interaction, a number of benefits may be expected, among them increased flexibility and greater reliability in the interaction sequence. The resulting agent-based system will provide a powerful and intuitive new means for interacting with computers and have potential application in a multitude of systems requiring high-level command.

The second issue addressed is that of dialogue modeling. In order for the multimodal interface agent metaphor to work, the agent has to be capable of a minimum set of skills: its underlying mechanism has to capture elements that are critical to the structure of multimodal dialogue, such as gestural signals, body language, turn-taking, etc., and integrate these in a way that works. I propose a computational architecture of psychosocial dialogue skills, called Ymir, that bridges between

multimodal input analysis and multimodal output generation. A character has been built in this architecture, called Gandalf, that can interact with humans in real-time, perceiving and generating various mutlimodal actions. By testing this character experimentally with human subjects, the validity of the approach is evaluated in various aspects.

1.2.1 Terms & Definitions

A few words on important terms are in order, without diving into the bottomless pit of definitions. The following terms are in special need of discussion: "Multimodal," "interface," "agent", "humanoid" and "psychosocial skills". The term "mode" as used here generally refers to an anatomically separate mechanism on the human body, or mechanisms carrying different kinds of data, enlisted for the purpose of communication with other humans, such as gesture and speech; intonation and body language, etc. "Multimodal" means therefore the collection of many such mechanisms. "Interface" traditionally means the place where two different systems meet: here it is the human and machine that meet, hence the term "human-machine interface." The term "agent" has served numerous meanings, but can be considered here to mean broadly "the conceptual categorization of one or more computer-stored goals, and the collective capability to carry out those goals, to the computer user's interest." A vacuum-cleaning robot would be a good example of an agent according to this definition. As we will see later, this is a slightly too broad definition for the current purposes, but it will do for now. A "humanoid" is that which duplicates many human characteristics, yet is *not* human. The distinction that is being emphasized by using this word is the one between animals, insects and related creatures on the one hand, and human-like creatures on the other. To be grouped with the latter one would have to share with humans at least some of our unique features: a human face, language understanding and generation, social skills, among other things. Lastly, "psychosocial skills" are the skills needed to orchestrate, co-operatively, goal-driven communicative interaction with other agents. The current work is thus a contribution to the broad scope of dialogue management, rather than narrower aspects or smaller parts of dialogue such as language understanding, gesture recognition or agent animation.

Since the emphasis here is on the full loop of multimodal input analysis and multimodal output generation, a number of assumptions have been made and gaps filled where research was lacking or too unwieldy for a one-man project. These include knowledge representation, linguistic issues, cognitive modeling and philosophical questions of all sorts. I hope the reader can forgive these unavoidable gaps in my treatise, and ask that you try to focus on the problem of full-duplex interaction, which, in my opinion, should be the starting point for all other issues of

dialogue. We can then leave it to future research to fill in the missing details.

1.2.2 Outline of Thesis

The first 3 chapters present background material: Chapter 2. discusses the face-to-face metaphor, Chapter 3. reviews the psychological research in multimodal communication and multimodal computer interfaces and Chapter 4. gives an overview of related research on software agents, robots and artificial intelligence.

> The skills themselves are basic: breaking eye contact when you want to speak; noting whether the other person is looking in the right spot when you point something out to them; describing things and events with your hands... Can such general, practical conversational expertise be imparted to computers?
>
> —Richard A. Bolt (1987, p. 2024)

Chapters 5., 6. and 7. present the approach taken here to creating interactive, humanoid characters, and the underlying assumptions. Chapter 5., "Computational Characteristics of Psychosocial Dialogue Skills", focuses on the hard issues in multimodal dialogue, their computational characteristics and ways to formalize these for implementation. A three-layer feedback model of multimodal dialogue is introduced that addresses its real-time constraints and mode integration. Chapter 6., "J.Jr.: A Study in Reactivity", describes a pilot system that served to explore the issues of real-time dialogue feedback, back channel and turn taking. The limitations that emerge from this study motivate many features of Ymir[1]—presented in Chapters 7. and 8.—a generative model for a communicative agent's sensory, decision and motor processes. Chapter 9. describes the first character created in Ymir, Gandalf. Chapter 10. presents the results of an evaluation of Ymir/Gandalf using human subjects, and discusses the methods used to create humanoid agents in Ymir.

Chapter 11. discusses validity in the design of multimodal agent-based interfaces and relates these to possible implementations of communicative humanoids. General conclusions from this work are drawn, and directions for future work given, in Chapter 12.

1. Pronounced "***e***-mir" with the accent on the first syllable. The name comes from Nordic religion; see side bar page 89 for background.

Face-to-Face Interface

2.

In this chapter we will discuss the general advantages and disadvantages of face-to-face interaction and how this relates to human-computer interaction, and look at some of the early history of humanoid agents. We will also take a non-traditional look at the issue of anthropomorphization—the act of attributing human qualities to non-human things.

....at every screen are two powerful information-processing capabilities, human and computer. Yet all communication between the two must pass through the low-resolution, narrow-band video display terminal, which chokes off fast, precise, and complex communication.

—Edward R. Tufte (1990, p. 89)

2.1 Humanoid Agents: Early History

The fascination with humanoid, artificial agents can be traced at least to the beginning of this century—not in research but in fiction. The first multimodal, interactive agents were probably Karel Capêk's mechanoids, in his play *R.U.R.* ("Rossum's Universal Robots") [1920]. This piece is the origin of the word "robot", the Czech word for "worker". Another landmark in robot fiction was Fritz Lange's *Metropolis* [1925], sporting a robot that was so believable it was virtually indistinguishable from humans. An all-time favorite multimodal agent in fiction was the artistically designed Robbie the Robot, first making its appearance in the movie *Forbidden Planet* [1956] and in many others after. Toward the latter half of the century we witnessed the appearance of an awe-inspiring HAL-9000 computer in Kubrick's *2001: A Space Odyssey* [1968] (communicating through multimodal input but only speech output), C3PO of *Star Wars* [1977] (multimodal I/O), and Holly—the ever cynical computer on-board the spaceship *Red Dwarf* [1988] from the BBC series with the same name. Holly is identical to HAL-9000 except for the very important aspect of having an embodiment as an on-screen face, entering the world of the user and capable of multimodal output. It seems that in fiction through the ages multimodal interaction has always been assumed; perhaps because it comes so naturally to us it has never seemed an issue. And, perhaps because robot researchers have been

FIGURE 2-1. Robbie the Robot saves its master.

busy working on vision, smarts and action control in separate corners of their laboratories, the issue hasn't really come up there either, until very recently.

In various recent "visions of the future" promotional videos, companies like Hewlett-Packard and Apple Computer [Laurel 1992] have presented the idea of agents that inhabit the world of the computer, but seem to have at least limited perception for outside things like the user's presence. These agents communicate mostly via speech and visual appearance as output, and simply speech as input. The visual channel as input is highly de-emphasized. However, recent progress in computer vision leads us to believe that recognizing people—where they are looking and what they are doing—may well be within a decade of being commercially viable [Essa 1995, Maes at al. 1995]. The added richness of a visual input channel could well make all the difference in interacting with artificial agents, determining whether people will actually "buy"—pun intended—this kind of interaction style with machines.

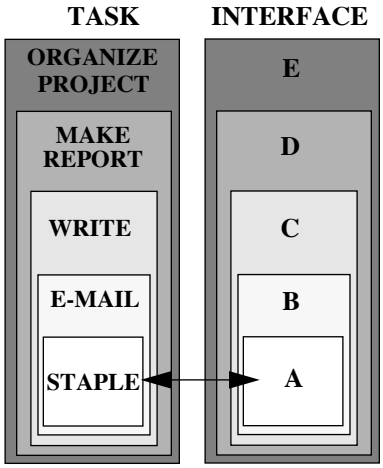


FIGURE 2-2. Darker colors indicate increasing underlying complexity, letters indicate a choice of interface. The task of stapling calls for a simple physical-tool interface such as a stapler (A); a high-level task, such as organizing a project, requires a communicative interface (E) and may require interfaces from the lower levels also. A mismatch between task and interface, for example replacing face-to-face interaction (E), with the interface for writing—a word processor (C) —is likely to compromise efficiency.

Most present-day robots have little idea about a "user" and their design is generally not "user-centered" in the usual sense of the term, although new research seems to be focusing more on this issue. For instance, Cannon's [1992] system employs a camera that the user can point at objects, give simple commands like "put that...and that...there," accompanied by a camera pointing in the directions, and the robot will automatically plan the execution of action for its mobile platform and arm. Brooks' [Brooks & Stein 1993] proposal for a humanoid robot includes a full upper body humanoid with stereo cameras for vision, stereo microphones for hearing, duplication of the human upper body degrees of freedom, and a massively parallel computer for brains. (Who needs fiction?)

While robots have changed relatively little in fiction since Câpek, research on various fronts is filling in missing knowledge and moving us closer to realizing well rounded artificial humans [Pelachaud et al. 1996, Prevost & Steedman 1994, Cassell et al. 1994, Thórisson 1994, Badler et al. 1993, Brooks & Stein 1993]. Although the main focus here is taking another step toward a new kind of interaction—not toward replacing or "bettering" any of the existing human-computer interfaces in existence—for completeness sake we will now quickly review the most obvious benefits and limiting factors of face-to-face interaction.

2.2 *Face-to-Face: When & Why*

In answering the question of when and why we would want to use a face-to-face[1] interface, two different perspectives can be taken:

1. *What kinds of tasks and systems are amenable to a face-to-face interface?*, and
2. *what are the necessary qualifications a system has to have to justify sporting a face-to-face interface?*

These issues are both really part of the same problem: how to fit an interface to a system (Figure 2-2). The issue boils down to two simple arguments: {1} Certain kinds of real-world tasks, namely supervision, need different interaction methods, namely communication, and {2} better systems can better support the complexity of natural interaction methods such as language, gesture and facial expression.

When determining the kind of interface for a system, we need to ask ourselves *What is the system capable of doing?* In other words *What is the nature of the task?* It makes little sense to install advanced speech recognition and in a normal, dumb, toaster when all it can do is turn on and off—the interface needed for such a dumb device is simply a switch labeled *"off-on"*. By the same token, if the toaster is extremely intelligent and can do many different things besides toast bread, crumpets or bagels[2], it is equally inappropriate to provide a user with a single on-off switch to interact with it. Because the relationship between the user action (turning the toaster on) and the outcome (the toaster heating up) is always the same—it is a completely reflex-based system. Such systems don't make any decisions of their own; they follow blindly the user's input. Even in systems such as nuclear power plants, which are orders of magnitude more complex than toasters, the interface is based on the same principle. The main difference is that the number of variables is exponentially higher, and the operators of a nuclear power plant have trained for months in how to interact with the system (plant) through the interface (control room, Figure 2-3).

Laurel [1992] lists some of the chores that intelligent interface agents might help with (Figure 2-4). A number of these tasks can be communicated about with less-than-human multimodal capabilities. Sheth and Maes [1993] for example describe agents that retrieve, filter, sort and organize a person's electronic news that simply use a "point-and-click" interface. So why would we even want to discuss face-to-face interaction?

On the other side of the coin is the intelligence level of the system: is the system really capable of supporting a face-to-face interaction? Can it support natural language without constant misunderstandings, break-

FIGURE 2-3. A nuclear power plant is really just a giant toaster. (Control Room One, D.C. Cook Nuclear Power plant, Ann Arbor, Michigan.)

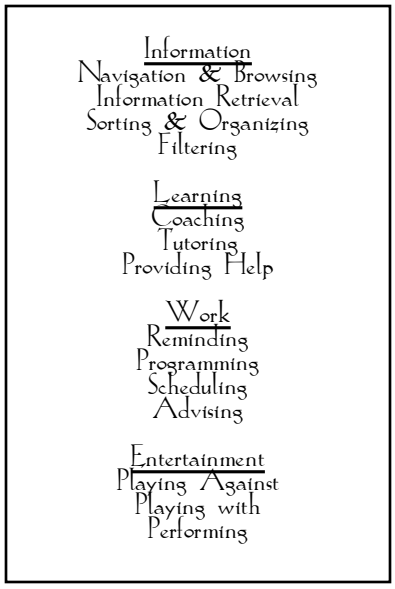


FIGURE 2-4. Laurel [1992] suggests these kinds of tasks as being ideal to delegate to a semi-autonomous computer agent.

1. I use the term "face-to-face" not only to refer to the presence of faces, but in general to the issue of co-presence and non-mediated communication.
2. My hat goes off to Grant and Naylor, the writers of Red Dwarf [1988], for the AI toaster example. You have to see it.

downs in communication or breaches in trust? Is it capable of feedback at multiple levels of granularity [Thórisson 1994, Clark & Brennan 1990]? If it is missing any of these, then it may be a mistake to try to force the link. However, even a dog-level intelligence justifies a multimodal interface, as evidenced by all the dog owners who happily use prosody, gesture and keywords to interact meaningfully with the canine friends. It may well be that dogs are an example of the lowest-level, non-verbal intelligence worthy of a multimodal interface. As the system gets better at understanding language, making its own decisions and executing actions, the need for a richer interface arises. The more capable a system is, generally speaking, the more complex the tasks that can be delegated to it, the more it makes sense to use high-level interaction methods like natural language. If the task to be accomplished with the system is formulated at a high level, including concepts such as *goals*, *intentions*, *plans*, etc., then it is very likely that natural language and a face-to-face metaphor will be a useful interface to the system. Natural language is also unique in that it allows for a kind of "downward compatibility": it has great flexibility in the level of detail it can address ("I want to make a pizza"; "is the oven on?"), and in that way does not paint a user in a corner.

Before discussing when *not* to use face-to-face interaction, let's look at some more reasons *for* it.

2.2.1 Some Compelling Reasons for Interacting Face-to-Face

A common knee-jerk reaction to the face-to-face interface might be the following: Interacting multimodally is really our only choice in the case of humans and animals, but *Why should we even consider communicating with a computer program in the form of a human when we can interact with it in a million other ways?* In this section I will review the most general arguments for using agent-based interfaces and in the next discuss some of the limitations of these as well as the relationship between the choice of interface with relation to a task.

The arguments in favor of the face-to-face metaphor can be divided into two categories: implementation dependent—related to particular realizations of multimodal systems—and implementation independent—related to the psychological and physical makeup of human beings. The latter is based on 4 key points:

1. *Synergy,*
2. *naturalness,*
3. *flexibility and*
4. *limits of metaphor*.

We will first take a look at these, and then discuss the implementation dependent arguments.

Synergy

The issue of benefits involves several interconnected questions: Why multiple modes? Why dialogue? Why face-to-face interaction? I say interconnected because in face-to-face interaction, sensory organs, bodily constraints, attentional and other mental limitations are linked together in a way that is intimately integrated and intertwined with the dialogue process. If we want to interact with an intelligent machine, it is therefore a big win if we model its interface in our own image, i.e. with a head, face, gaze, arms, hands, and a body—organs that have to do with communication.

Bolt [1987] discusses some of the strongest arguments for such multimodal interfaces. He points out the clear benefit of increased redundancy in the input, potentially reducing errors in the communication. Signals that occur in verbal communication are tightly linked with nonverbal cues. Recognizing both of these can increase the reliability of the interaction. He also points out the added richness of a multimodal interface: different modes have different ways of communicating. A face is an incredibly rich information display [Tufte 1983], and, more importantly, a natural part of the human communication mechanism. It is important to recognize that this argument serves on both sides of the equation, not only for input, but for computer output as well. These points relate to the *synergy* of multimodal communication, i.e. they argue in favor of an interface that integrates many features of face-to-face interaction rather than one that selects or singles out one or two features in isolation.

Related to the point of synergy is the following argument: It is the year 2010. I walk up to a speech recognizer in a train station I have never been to before to buy tickets. What kinds of words am I allowed to use? What kind of sentences are acceptable? Just speaking into a microphone, it is hard enough to pace the interaction, not to mention selecting the right things to say. When interacting with beasts of unknown intelligence, with vaguely known skills and unknown linguistic capabilities, we humans need all the help we can get to make it easier to predict what kinds of things we can talk about with it, what kinds of words we may use and what kind of performance we may expect from it. This information can be given by the interactive intelligence's appearance, body language, facial expressions, gaze behavior and turn taking skills. The stilted way I am asked if I can be helped, the fact that the face on the screen looks non-human, the hesitating manner of answering, the jerkiness of the smile all tell me that I should use simple language and get straight to the point as I ask if for the schedule of the D-train.

Naturalness

Why dialogue? Why not just write a letter or send the computer an e-mail to tell it what we want? Dialogue is structured around the turn. People cannot, as it were, talk and listen at the same time. Turn taking makes using language, as well as the various multimodal communicative devices, very efficient and effortless [Sacks et al. 1974]. It also makes it easier to integrate interaction between collaborating agents into an ongoing, common task by giving interlocators greater process control [Clark & Brennan 1990]. Thus, dialogue and turn taking are both an integral part of any language-based multimodal system. This relates to the *naturalness* of face-to-face interaction.

Flexibility

Why use a metaphor of human face-to-face communication instead of simply designing each system to accept exactly the kinds of modes needed for the task it is to perform? Pen and speech here, gesture and gaze there, etc. This question relates to the *flexibility* of the interface. It has a two-part answer. To a certain extent, of course, people do this when interacting with each other: we grab a pen and scribble on a napkin, they gesture at certain times and not others, etc. But notice that these options are all available in an instant, once we decide to use them. It is flexibility that makes multimodal dialogue so attractive. And although speech has been shown to be sufficient to successful human communication in many cases [Ochsman & Chapanis 1974], in its "high-bandwidth" instantiation it is accompanied by feedback mechanisms on multiple levels [Goodwin 1981, Yngve 1970]. A primary thrust for using social communication as a metaphor in human-computer-interaction stems from thus the presumed increase in "bandwidth" as when compared, for example, to command-line or graphical user interfaces [Brennan 1990, Clark & Brennan 1990], and the flexibility of being able to switch reliably between—and freely combine—gesture, language, glances and facial expressions to convey one's wishes and requests [Whittaker & Walker 1991, Bolt 1987].

The second part of the argument centers on *coordination:* pacing dialogue is difficult in the absence of feedback [Nespolous & Lecours 1986]. Thus, if we want to communicate complex commands to the computer that involve multiple steps, the best method is doing it face to face in the presence of clear, socially compliant feedback mechanisms that indicate comprehensibly to us that our commands have been understood.

Limits of Metaphor

My personal favorite support for multimodal human-computer interaction comes from a simple observation: computers are becoming more and more capable; speech recognition, gesture recognition, face recognition, object recognition ... the list goes on. If we continue to interact with computers in the old style that is modeled after the way we manipulate inanimate objects in the world, then computers will continue to appear more and more complex and confusing to their operators, until they become so cumbersome that new additions are not worth the trouble. We can try to imagine a typical error message in this hypothetical future:

> WARNING! FILE ERASE PREVENTED BY COLLABORATIVE AGREEMENT. You cannot erase those files because the trash folder recognized your voice and asked the files to verify your identity; they in turn have identified your face and warned the trash folder that they are 87% certain (average certainty for all files in question; standard deviation = 28.23%) that you don't have the right privileges to delete them.

This future hell of files with perceptual abilities, thinking folders and decision-making icons can and should be avoided. What is needed is a new interaction metaphor that takes us to the next level of human-machine relationship. Fortunately this metaphor exists: we already use such an interaction style with each other. Its called social interaction and is based on the notion of localized agency (a person is a localized agent capable of action). Since we interact with intelligent beings (agents) by communication, it only makes sense to start looking at communication as the next logical step in the evolution of the computer. And the most basic method of such interaction—the one that all others are and will continue to be compared to[3]—is face-to-face dialogue.

Implementation-Dependent Arguments

So far we have reviewed implementation independent arguments for the face-to-face metaphor. However, other more practical concerns related to technology also come into play. One relatively new line of argument for focusing on robustness in this kind of communication is the promise that future machines will be equipped with cameras that can sense their users [cf. Essa 1995, Maes et al. 1994]. By introducing cameras the user is freed from having to "dress up" into body-tracking gear such as

3. Some may object to this claim on the grounds that we could simply re-engineer ourselves to allow us wireless transmission of thought or perception of multiple places and times simultaneously. When this becomes a viable option, I am willing to reconsider my stance. In the mean time this argument will belong in the science fiction domain.

gloves and suits [see Bers 1995a]. However, because of various confounding factors such as variation in lighting, occlusion, etc., reliability in the analysis of input may be expected to drop.[4] Capturing information from multiple sources and modes will enable the computer to make more reliable inferences about the state of the dialogue and the user's input, and make it possible for the user to adapt to the situation by dynamically choosing the most appropriate mode combinations depending on the computer's multimodal responses.

A similar case can be made for speech recognition: by collecting information such as a speaker's direction of gaze, direction of head-turning, etc., a speech recognition system can know when an utterance is meant for it and when it is meant for a by-stander. Variations on this theme could allow a system to switch dynamically between vocabularies during interaction and thus increase the reliability of the recognition process.

Numerous other arguments have been put forth about the benefits of multimodal, socially-oriented interaction with machines [Brennan 1990, Laurel 1990, Bolt 1987]. However, the strongest argument for interacting socially with computers comes from the simple observation that most people in the world interact frequently with other people, and are thus constantly practicing this kind of communication.

2.2.2 Face-to-Face: When *NOT?*

We have already mentioned that if a system cannot support the most important features of face-to-face interaction, we shouldn't try to attach that kind of an interface to it. However, given that we want a communicative-style interface, when would face-to-face provide the right features? The following discussion in this section is mainly based on Clark and Brennan's and paper *Grounding in Communication* [1990] and Whittaker and Walker's *Toward a Theory of Multimodal Interaction* [1991].

Clark and Brennan's work is directed toward the process of grounding, the process in which two interacting agents come to share mutual knowledge, mutual beliefs and mutual assumptions. This process is considered to be inherent in any communication task. Whittaker and Walker [1991] show how these concepts are generalizable to the analysis of the cost of different media for various tasks in the computer domain. The key concepts the authors identify are:

4. It may be argued that a certain level of uncertainty will always be present, save perhaps for highly artificial environments, because the world is far too complex to be completely predicted, hence the increased need to ensure reliability.

1. EXPRESSIVITY — what kind of information can be conveyed in the medium?
2. PERMANENCE — how permanent is the medium; does it allow for review and revision?
3. INCREMENTALITY — what is the granularity of feedback the system can give to user actions?

They reach the conclusion that for tasks with strict requirements in permanence, speech is not a good medium—if we have the choice of a single medium only. Examples of tasks that rely heavily on permanent media are writing, drawing or construction in general. However, in almost all tasks is there a use or preference for a separate, less-permanent channel. The inverse is also true: for tasks such as brainstorming, planning or coordination that rely heavily on speech, the use of permanent media (e.g. a pencil and paper) enriches the interface tremendously, while leaving information transmission mainly to the speech channel.

If a system is highly restricted to a single level of granularity, a face-to-face metaphor is unlikely to provide the most efficient interface. Examples of such tasks would be manipulations of single, unique objects, where the need to repeat the same action on multiple objects does not exist, and that requires few or no abstract relations between objects and actions (e.g. "Find *all* Ys that are *part of* X *and* have attribute Z, *excluding* Ys that are also Ts"). The same can be said for tasks with limited need for temporal specification ("Do X and Y simultaneously, then Z") and tasks with a minimal real-time component [Walker 1989].

Rather than restricting conditions of the task, as in the above examples, restricting the transmission channel has more obvious effects on our choice of interaction method. When there is a high latency in the information transmission channel, face-to-face interaction is generally a bad choice of interaction, because to be effective it requires rapid, full duplex feedback on multiple levels in multiple modes. If the transmission medium allows for only limited bandwidth, face-to-face interaction is not feasible, since out-of-sync sound and pictures tend to disorient rather than enhance [cf. Whittaker 1994, Whittaker & O'Connaill 1993, O'Connaill et al. 1993]. This condition, however, may still be perfectly suitable for speech-only communication. Asynchronous delays in message transmission will further diminish our reasons to choose face-to-face or speech-only interaction over for example, e-mail, fill-forms, or any other method where the permanence of the transmission medium allows error-free communication to take place.

Any good theory of communication should be able to allow us to constrain at will the initial conditions of the system for which we want to design an interface, and this is precisely what the Whittaker & Walker

[1991] research tries to do. The interested reader should be able to follow this thread in more detail through their work.

an.thro.po.mor.phism *n* (1753): an interpretation of what is not human or personal in terms of human or personal characteristics: humanization — **an.thro.po.mor.phist** *n*

— Merriam-Webster's Collegiate Dictionary, Tenth Edition

2.2.3 Anthropomorphization: A Non-Traditional Perspective

Orthogonal to the task of choosing the right interface are the issues of agency, autonomy and anthropomorphization. Is anthropomorphization of an agent-based interface necessary? Considerable fuss has been made over the perceived pitfalls of anthropomorphization—the act of attributing human-like qualities to inanimate objects or animals of low intelligence. With regard to the anthropomorphization of computers, researchers seem to have varied opinions [Lanier 1995, Maes 1993, Chin 1991, Laurel 1992, Laurel et al. 1990, Laurel 1990] and the systems to date that deliberately use anthropomorphization seem as varied as people's opinions of them.

As I have already alluded to, the kind of interface chosen should be justified by system capabilities, and be suited to the task being performed with that system. In this view, whether the interface follows a face-to-face metaphor or is less a question of personal preference and more an issue of efficiency. But what about anthropomorphization—should it be avoided—can it be avoided? I would argue that for many complex tasks, there is little choice on the designer's part whether the system is presented in anthropomorphic terms or not. The argument is based on the simple observation that as systems become "smarter", i.e. become capable of handling behaviors and concepts that are normally attributed to people, like integrating various data sources, perceiving their environment and making independent decisions, understand speech, people's willingness to anthropomorphize increases. To take an example, we would have a hard time convincing anyone that a rock is autonomous or has any amount of intelligence. So, it follows that it is difficult for us to imagine a rock has having a character or being an agent. A rock represents one extreme end of a continuum from "dead" to "alive". Moving along this continuum, our ability to anthropomorphize is made somewhat easier given systems that handle simple delegation, for example systems for fetching electronic mail at certain times of the day. Because you delegate the task of fetching mail to the system, the system embodies some level of autonomy, and hence is easier to anthropomorphize. Most people would probably agree that dogs are very easily anthropomorphized. This path from dumb to smart systems indicates a trend that implies an impasse toward high intelligence: as a system becomes increasingly smarter, the designer's ability of that system to influence a user's tendency to anthropomorphize that system decreases toward nothing. I call this "The Intelligent System Designer's Deadlock", or just *Designer's Deadlock* for short.

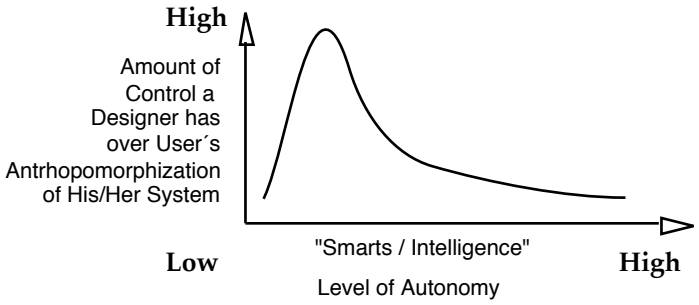


FIGURE 2-5. As the "smarts" or level of autonomy of a system increases, its designer has less and less control over the user's tendency to antrhopomorphize that system.

What we get is a curve that looks something like Figure 2-5. For any system, as we select that system's level of intelligence on the abscissa, we get a value on the mantissa that shows the freedom the system's designer has in controlling a user's anthropomorphization of that system. Somewhere toward the lower end on the "smarts" scale people are very good at imagining a system as being an agent; this is the level of our most intelligent systems today. The Designer's Deadlock effect is exacerbated as we add to the system features that borrow visual human or animal-like features like faces, hands, eyes, facial expressions, etc. Two of the strongest factors in driving home anthropomorphization of an intelligent system are probably speech (even a system capable of very limited speech may be seen as a "stupid" humanoid) and a face (even a toy with a face can be perceived as having human-like characteristics; a toy with no face has much less of a chance).

This argument is supported by recent research on users' perception of technology [Nass et al. 1994, Nass et al. 1993]. This research has shown that when computers are equipped with human-like capabilities such as speech synthesis or speech recognition—in fact, even when it communicates with simple text—users perceive them as agents with human-like capabilities. Rather than ignoring or trying to eliminate the agent-like qualities that computers are perceived to have, one can capitalize on the fact and make the interaction more stable and effective.

2.3 Summary

In this chapter we reviewed some of the early fascination in the arts with humanoid artificial agents. We discussed the advantages and disadvantages of the face-to-face interaction metaphor as applied to interaction with computers, and the nature of anthropomorphization. We con-

cluded that in spite of human-like interfaces not being problem-free, it is better to acknowledge them and try to take advantage of them in interface design, than to wish they would go away and get stuck with unsolvable problems. Whatever may be said against face-to-face interaction as a method of communication, the evidence reviewed certainly supports the argument that trying to build a computer system based on these ideas is far from being a waste of time.

Multimodal Dialogue: Psychological and Interface Research

3.

In this chapter we will look at some of the psychological and computational research relevant to the task of building face-to-face interfaces. When explicitly applying the face-to-face metaphor to computer systems, 3 interdependent elements stand out:

- Dialogue structure. *The structure of human face-to-face dialogue is organized around the turn taking system. This system has the properties of requiring real-time responsiveness and concurrent input and output.*
- Multiple modes. *The inputs and outputs are multimodal, including speech, gesture and other visible behaviors.*
- Embodiment. *Face-to-face interaction requires participants that are embodied, which in turn gives meaning to their situated visual and auditory behavior.*

These will be used to focus the discussion in this chapter. In addition, an overarching theme is the notion of reciprocity in dialogue. Reciprocity is not only a major part of content coordination, as convincingly shown by numerous researchers [Clark & Brennan 1990, Goodwin 1986, Grosz & Sidner 1986, Kahneman 1973], but part of all elements of discourse. A major assumption in this work is that for the multimodal conversation metaphor to reach its full potential, we need to support the full feedback loop from user to machine and back, and address the metaphor's key elements on all levels. The following discussion will therefore necessarily be broad, covering first the structure of dialogue at all levels, as well as multiple modes and embodiment, and then go into implemented computer systems based on the multimodal metaphor.

3.1 Human Multimodal Communication

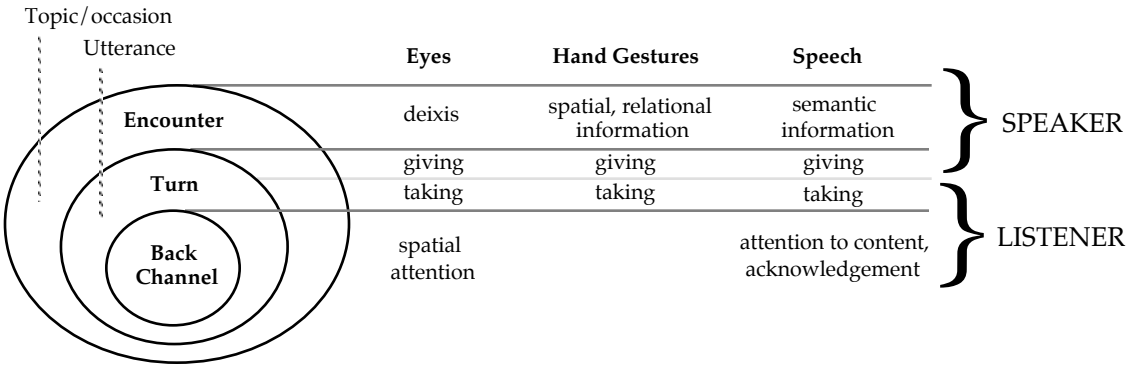
3.1.1 Dialogue Structure

Recent research in linguistics has indicated that in discourse, communicating parties strive to reach a common ground, a process that has been referred to as grounding [Clark 1992, Whittaker & Walker 1991, Clark & Brennan 1990]. The success of the grounding process depends on the successful support of dialogue by the common organizational principles of turn-taking, back-channel feedback and other multimodal communicative mechanisms [Sacks et al. 1974], as well as on focus of attention, indicated through gaze and spatial orientation of the interlocators, directing each other's attention with gaze and gestures [Clark & Brennan 1990, Goodwin 1986, Grosz & Sidner 1986, Kahneman 1973]. For the current purposes, we will adopt a 3-level hierarchical model of face-to-face interaction according to dependencies of its coordination constituents and the granularity of time. The highest level can be said to be the encounter. The encounter includes the whole interaction sequence that occurs when two or more people meet, including greetings and good byes [Schegloff & Sacks 1973], choice of topic, reason for the meeting, etc. Actions at this level happen at the slowest rate. The psychosocial actions in a conversation happen at the next two levels down, the first of which is the turn, the second being the back-channel. We shall now look at each in turn.

3.1.2 Turn Taking

When people communicate in face-to-face interaction they take turns speaking [Duncan 1972]. Goodwin [1981, p. 2] says about the turn:

FIGURE 3-1. The three main processes in face-to-face interaction can be thought of as hierarchically nested within each other (circles) according to their time span and time-criticality; the functional roles of speech, gesture and gaze in each conversational process are shown to the right.



> "In the abstract, the phenomenon of turn-taking seems quite easy to define. The talk of one party bounded by the talk of others constitutes a turn, with turn-taking being the process through which the party doing the talk of the moment is changed."[1]

The turn system's main function is to manage the sequential nature of talk. It organizes the information exchange between two (or more) communicating parties and ensures efficient transmission between them. The information can be constructed through speech, hand gestures, body language, gaze, facial expressions, or any combination thereof [Sacks 1992b, McNeill 1992, Goodwin 1981]. Turn-taking and back-channel feedback have both been shown to be important for conducting successful dialogue [Sacks et al. 1974, Nespolous & Lecours 1986]. Turn-taking is, for example, crucial in both negotiation and clarification [Whittaker et al. 1991, Whittaker & Stenton 1988, Sacks et al. 1974].

Sacks et al. [1974] put forth a model of turn taking that models the structure of human conversation as an emergent property of local decisions based on prediction by the participants. Because theirs is a thorough model, as psychological models go, and relates directly to psychosocial dialogue skills, I will briefly recap its main points. In their view, turn taking is locally managed and participant-administrated. Local management means that "all the operations [within the system] are 'local', i.e. directed to 'next turn' and 'next transition' on a turn-by-turn basis" [p. 725]. In this view, any pattern that arises out of interaction is "emergent"—i.e. results from the interaction of rules. They say further [p. 725-6] that

> "the turn-taking system is a local management system ... in the sense that it operates in such a way as to allow turn-size and turn-order to vary and be under local management, across variations in other parameters, while still achieving both the aim of all turn-taking systems—the organization of 'n at a time'—and the aim of all turn-taking organizations for speech-exchange systems—'one at a time while speaker change recurs'".

Party-administration refers to the fact that the rules of turn-taking are subject to the conversants' control, i.e. that the rules are designed for being used by each participant to manage their communication with others. By hypothesizing the existence of turn-constructional units, they

1. Goodwin [1981] then goes on to say that on closer inspection things are not as simple as they look. However, the notion argued here is that the principle of turn taking is simple while the behavior emerging from the interaction of the principles of Sacks et al. [1974], especially when observed "in the world," can be quite complicated.

were able to model turn taking with only five—albeit relatively complex—rules. The particulars of the rules are not important here: by far the most important part of their theory is the set of turn-constructional units they propose, which are *sentential*, *clausal*, *phrasal* and *lexical*. These components are used by speakers to construct a turn. For example, recognizing that a particular sentence of type A is being uttered by a speaker, a listener can use her knowledge about sentence type A to predict when it ends, making it possible to take turns with no gaps. However, Sacks et al. fail to specify what kinds of turn-constructional units distinguish one type of utterance—and multimodal act—from another. If we assume that a listener is continuously looking for clues about types, or functions, of utterance segments, a resulting conclusion would be that what is important for extracting these are the features of the utterer's behavior, because, apart from the content of the speech, these are the only clues to the function of the speaker's actions. From a descriptive point of view, turn-constructional units may be valid, but they say nothing about the way people actually recognize these units. What is needed is a mechanism that allows sentential, clausal, phrasal and lexical features to be recognized in real-time and integrated with a discourse participant's actions to allow the pattern of turn taking to emerge. In Chapter 7. we will present a general approach to achieving this.

In what seems to be an incompatible approach, Duncan [1972] proposed the existence of "cues" for turn signalling. It may be argued that Duncan's cues are simply parts of the features that conversants use to identify the turn-constructural units of Sacks et al. In reality, a person uses her perception to make the best or most appropriate decisions at any time regarding her behavior; perception decision is constrained by time, accuracy and the knowledge of the participant. We will come back to this issue in later chapters.

3.1.3 Back-Channel Feedback

Face-to-face interaction quickly breaks down if communication can only happen at or above the turn level [Nespolous & Lecours 1986]—there needs to be a two-way incremental exchange of information. Part of the task for a listener is to make sure that the other party knows that she is paying attention, and indicate that she is at the same state in the conversation. This is done mainly in the back channel [Yngve 1970]. Back channel feedback is in effect information exchange that supports the interaction itself and helps move it along the right path [McNeill 1992, Goodwin 1981]. It includes using paraverbals such as "m-hm," "aha," etc., indicating confusion, expressing feelings (by facial gesture, laughter, etc.), and indicating attentional focus. The absence of such regulatory gestures from a listener may disrupt the discourse [Dahan, as referenced in Nespolous & Lecours 1986].[2] While it may be argued that overlapping talk in the main communication channel is counter-pro-

ductive because it interferes with the flow of a conversation [Sacks 1992b], co-occurring speech in the paraverbal channel does not [Yngve 1970]. The main stream of information (from the speaker) and back channel feedback (from the listener) can therefore be modeled as two separate information channels that can be used simultaneously without interfering with each other. One rule-of-thumb definition of back-channel feedback then is that it is the ongoing (communicative) behavior of a listener that does not change who is in control of the dialogue at the moment.

The above discussion strongly implies that a simple "transmitter-receiver" model will not be sufficient when transferring multimodal interaction to the computer domain. Let us now take a closer look at the role the modes play in multimodal conversation.

3.1.4 Embodied Conversants

Two spatial constraints are of importance to conversation. The first has to do with the location of discussants to each other and the surroundings, referred to here as positional elements, and the second has to do with the conversants' relative orientation, what will be referred to here as directional[3] elements. Surprisingly, research on this topic in psychology is relatively scarce.

Obviously the position of a conversational participant has implications for spatial reference: glances, pointing gestures and direction-giving head nods will be done differently depending on where the speaker and listener are positioned in space. The display of visual cues such as facial gesture is bound to a specific location, i.e. the participants' faces. Multimodal conversants have to be able to find their conversational partners in space—otherwise they would not know where to find the necessary visual information when interpreting each other's utterances or assessing dialogue status. This is important, since a number of turn-taking signals rely on participant location and facial cues [Duncan 1972], and many back-channel feedback cues are given through the face [Goodwin 1981]. Manual gesture are usually given in the area right in front of the gesturer's body [McNeill 1992], and these have to be located in space as well. Gaze is often used to reference this space [Goodwin 1986], and can be indicative of the kind of gesture being made [McNeill 1992, Goodwin 1981].

---

2. Nespolous & Lecours [1986, page 61] say: "... Dahan [see ref., op. cit.] convincingly demonstrated that the absence of regulatory gestures in the behavior of the listener could lead the speaker to interrupt his speech or to produce incoherent discourse."

3. Thanks to Steve Whittaker for suggesting the term "directional."

Orientation has to do with how the participants are turned relative to each other, how various body parts are oriented, and how this changes over the course of the interaction. [Goodwin 1986] For example, turning your head away right after your partner finishes speaking could indicate to him that you think he's done and that you are now preparing a response [Goodwin 1981]. Research has shown that when talking face-to-face, people generally prefer to orient their bodies approximately 90° to each other rather than directly face-to-face [Sommer 1959].

3.1.5 The Multiple Modes of Face-to-Face Interaction

Speech

It has been argued that speech is the main content carrier in face-to-face communication [Sacks 1992a, Sacks 1992b, Ochsman & Chapanis 1974] and may even be the critical medium [McNeill 1992]. Research on language is far more advanced than other aspects of the multimodal interface and is by now a highly mature field compared to other aspects of human communication. Various techniques for parsing natural language have been proposed [cf. Allen 1987]. A clear indication of this is that speech recognizers can now be bought off-the-shelf that are speaker-independent, have a relatively large vocabulary and recognize continuous speech. Researchers have also begun to investigate the link between speech and other aspects of discourse [McNeill 1992, Pierrehumbert & Hirschberg 1990]. For example, McNeill [1992] argues that while on the surface gesture may seem dependent on speech, they often carry different information from the speech they accompany. He proposes that speech and gestures both arise from a common knowledge representation. Pierrehumbert & Hirschberg [1990] have shown how intonation affects the interpretation of speech in context.

The vocal channel is of course also used to give back-channel feedback and other feedback related to process-control. Among the implications of the theory put forth by Sacks et al. [1974] is that turn-taking is a necessary element of any conversational system. They argue that the turn-taking system for speech in fact makes its understanding easier. As a consequence, implementing turn-taking rules and dialogical conventions in multimodal interfaces should make speech communication more robust [Brems et al. 1995], for example by making it easier for the computer to infer where utterances begin and end—still a serious limitation of continuous speech recognition [BBN 1993].

Manual Gesture

In multimodal dialogue, gesture frequently happens along with speech. McNeill [1992] has suggested that gestures and speech are generated

from the same underlying representations in the brain, and others have suggested that the first hominid language was in fact based on gesticulation [Zimmer 1995]. Many classification systems have been used to describe the kinds of gestures people make in discourse [Rimé & Schiaratura 1991, Poyatos 1980], most of them being modifications of Effron's [1941/1972] classification scheme (Figure 3-2). To recap this classification scheme: Symbolic gestures have a direct interpretation in a given culture. An example is the "thumbs up" sign. Deictic gestures are generally referred to as "pointing" gestures. They direct a listener's visual attention to a spatial area or location. To date, symbolic and deictic gestures have been the primary gestures of study at the computer interface (see Table 1). Other kinds of gestures, classified as iconics, beats, pantomimics, metaphorics, tend to carry equally important (and often more complex) information [McNeill 1992, Cassell & McNeill 1991]. Iconic gestures are the kinds of gestures where a body part, often the hands, play the part of another object for the purpose of demonstration. An example would be moving your hand forward, palm down and saying "The car drove like this" meaning that the car moved in some sense the same way your hand does. Pantomimics are gestures where the hand or body of the gesturer are interpreted as real hands. An example is miming the action of hammering or opening a door. Metaphorics are iconic in that they assign meaning to space, but instead of representing concrete objects or events, they present abstract ideas. Beats are rhythmic gestures that accompany speech that have been found to play a large role in the sequencing of turns in dialogue [Duncan 1972], and also to be related to shifts in the dialogue narrative, for example from the main story line to side issues [McNeill 1992]. A last category of gesture is one that should perhaps be classified under "action" instead of being called a "gesture." This is the class of self-adaptors [McNeill 1992, Ekman & Friesen 1969]. Self-adaptors are actions like fixing one's hair, scratching, etc. It has been shown that people attend to such gestures and integrate information conveyed by gesture into their representation of a narrative [Cassell, McNeill & McCullough, forthcoming].

Facial Gesture

Facial gesture has been extensively studied by Ekman & Friesen [1978]. Facial gestures have been found to regulate interaction and they are the primary method, along with intonation, for displaying affect [Ekman 1979]. Pelachaud et al. [1991], following Ekman [1979], classify facial gesture into emblems, emotional emblems, affect display, conversational signals, punctuators, regulators and manipulators. Emblems are movements whose meaning is culturally dependent. An example is nodding for agreement. These gestures correspond to the type of hand gesture that has been referred to by the same term.[4] Emotional emblems convey signals about emotion. The crucial distinguishing feature here is that the gesturer does not feel the emotion at the time of the gesture, but

1. Nondepictive gestures: speech markers (beats)

   A. Stress some elements of the speech for the sake of clairity.
   B. Parallel the introduction of some new element in the discourse.
   C. Chunk the sentence following the steps of the underlying reasoning.
   D. Related: batons, minor qualifiers, beats, paraverbals.

2. Depictive gestures: ideographs.

   A. Sketch in space the logical track followed by the speaker's thinking.
   B. Parallel abstract thinking.

3. Iconographic (iconic) gestures

   A. Present some figural representation of the object evoked in speech.
   B. Subclass: (a) pictographic: represents the shape.
   (b) spatiographic: represents some spatial relation.
   (c) kinetographic: represents some action.
   C. Related: physiographic, motor-primary, illustrative gestures.

4. Pantomimic gestures

   A. Play the role of the referent.

5. Deictic gestures (pointing)

   A. Point toward some visually or symbolically present object.

6. Emblematic gestures (symbolic)

   A. Are devoid of any morphological relation with visual or logical referent.
   B. Have direct translation into words.
   C. Have a precise meaning known by the group, class, or culture.
   D. Usually deliberately used to send a particular message.

FIGURE 3-2. Classification of the kinds of gestures encountered in natural dialogue (after Rimé & Schiaratura [1991]).

merely refers to it via the facial display.[5] Affect display, on the other hand, is the direct expression of emotion. Conversational signals are facial gesture made to punctuate speech, to emphasize it. An example is that raised eyebrows often accompany accented vowels. Punctuators are movements that occur during pauses. Regulators control the speaking turn in a conversation. Manipulators correspond to self-adaptors of hand gestures. An example for the face would be blinking to keep the eyes wet.

Gaze

Most psychological research dealing with gaze has used it as an indirect measure of something else: how long does it take to read a word, what are the mental stages we go through when we try to understand some-

4. Some researchers use the term “symbolic” instead of “emblems”.

5. This classification makes a boolean class out of a continuum, since a facial emotional emblem could be related in any degree to the underlying emotions.

thing three-dimensional, how much resolution do we have in our peripheral vision, etc. The emphasis here is on the role that gaze plays in communication.

Gaze has been shown to be related to a person's attention [Kahneman, 1973], deictic references [Cooper 1974], mental activity [Rayner 1984, Yarbus 1967], and personality, interpersonal attitudes and emotional states [Argyle et al. 1974, Kleinke 1986]. Primarily, gaze is an indicator of a person's attention over time [Kahneman, 1973], and provides therefore crucial information in the conversational setting. People have a strong tendency to look toward objects referred to in conversation [Cooper 1974], which can provide listeners with important deictic information. People will even look where they are listening [Riesberg et al. 1981]. Research has shown that people are extremely good at estimating the direction of gaze of others [Anstis et al. 1969, Gibson & Pick 1963]. The accuracy is dependent on the 3-D aspects of the eyes, the presence of a face around them and the position of the viewer in relation to the eyes [Anstis et al. 1969].

Yarbus [1967] was among the first to show that eye movement patterns vary according to the mental activity of the looker. Subtle differences in gaze pattern were observed to correlate with subtle differences in the task that the looker is engaged in. For example, a picture containing people will be scanned slightly differently depending on whether the onlooker is trying to estimate the people's ages or their wealth. Whether subtle differences like these can be picked up by participants in a conversation is, on the other hand, a question that is difficult to investigate.

Since the eyes are used to gather information, their movements also tell others about this information gathering process. It is therefore not surprising that the eyes also are important in the regulation of turn-taking between dialogue participants [Argyle & Cook 1976]. Argyle and Cook [1976] have shown that the "...gaze patterns of speakers and listeners are closely linked to the words spoken, and are also important in handling the timing and synchronizing of utterances" [p. 98]. They have found gaze to serve three main functions: sending social signals, opening a channel to receive information, and controlling and synchronizing speech. There is a "...very rapid and complex coordination between speech, gaze and other non-verbal signals" [p. 114].

At the initiation of conversation, and during farewells, the amount of gaze between the conversants increases. For the period of the conversation they tend to reach an equilibrium in the amount of mutual gaze. The amount of expected mutual gaze given two speakers' look time can be found by using the following formula [Argyle & Cook 1976, Argyle & Ingham 1972, Strongman & Champness 1968]:

$$EC = EC_1 + EC_2 = \frac{L_T(A) \cdot L_L(B)}{A\text{'s talking}} + \frac{L_T(B) \cdot L_L(A)}{B\text{'s talking}}$$

where EC is expected mutual gaze, L_L represents looking (at other person) while listening, L_T is looking while talking, and A and B are the conversants. In a normal conversation, the average amount of looking at the other person while listening is 75%; the average time spent looking while talking is 41% (given that neither party is trying to avoid or seek visual contact). A and B's look times are determined by the social context (how close people are, who is the other's superior, etc.). Although this formula could hypothetically be used for controlling the gaze behavior of a computational agent or robot by approximating the value of EC in real time during conversation, given the user's gaze input, a more realistic approach would try to model the mechanisms underlying the gaze pattern observed. A number of factors complicate the matter, among them the fact that in addition to being dependent on dialogue state, gaze behavior also varies with the topic of discussion [Cooper 1974]. On top of this lie multiple mental processes influencing the exact observed gaze pattern.

Multimodal Synergism

An important claim of the turn-taking theory put forth by Sacks et al. [1974] is that to get reliable interaction, interactors need to have an understanding of multiple modes. Thus any system that proposes to use turn-taking—as it occurs in human-human interaction—as part of a computer interface will need to incorporate multimodal analysis and interpretation. As we have already mentioned, the flexibility of social interaction stems both from an ability to switch dynamically between representational styles and from combining modes for displaying a single message [cf. Goodwin 1981, Poyatos 1980]. A synergism of multimodal interaction results from the combinatorics of various modes and signals at specific times in the interaction sequence. Any system that tries to introduce flexibility into multimodal human-computer interaction has to take this into consideration. Research on the combinatoric aspect of face-to-face dialogue is still in its infancy [Poyatos 1980] although some guidelines are emerging. Clark and Brennan [1990] present a cost model for combining multiple modes given various constraints in the communication channel. Whittaker and Walker [1991] discuss further the advantages of media types for interface design. The advantages of exploiting the synergistic nature of mode combinations at the computer interface are discussed in Bolt [1987] (see "Face-to-Face: When & Why" on page 26).

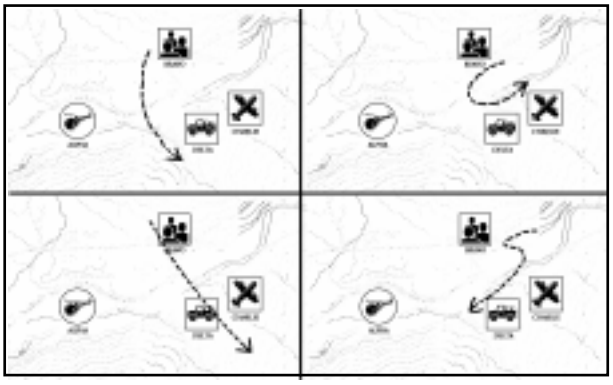


FIGURE 3-5. Example of a multimodal interaction. The user can say "Delete [gesture] these icons" and do a gesture (dotted arrows) near a group of objects. A simulated perceptual grouping algorithm enables the computer to infer which objects the gesture refers to—independently of its precise form [from Thórisson 1994].

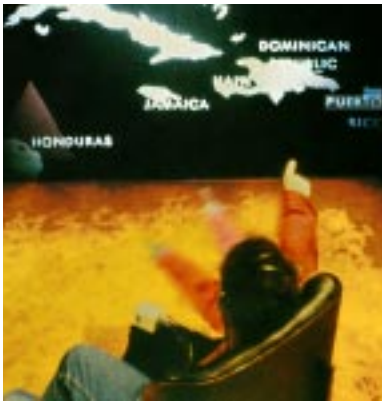


FIGURE 3-3. *Put-That-There* was an early multimodal interface prototype [Bolt 1987].

3.2 Multimodal Computer Interfaces

Having looked at psychological and linguistic research, we now turn to previous computer systems that build on the idea of multimodal, social communication.

In the past, implementations of multimodal computer interfaces have included the use of natural language, either spoken or written, and, to varying degrees, gestural input and eye tracking. A comparison of recent systems is shown in Table 1. One of the first (if not *the* first) system to demonstrate gesture and speech at the computer interface was *Put-That-There*, developed by the Architecture Machine Group at M.I.T. [Bolt 1987, Bolt 1985, Bolt 1980]. *Put-That-There* used speech-recognition and a six-degree-of-freedom space sensing device to gather input from a user's speech and the location of a cursor on a wall-sized display, allowing for simple deictic reference. Recently there has been an increased effort to combine gestures and language at the interface [Bers 1995a, Thórisson 1995a, 1994, Wexelblatt 1994, Koons et al. 1993, Sparrell & Koons 1994, Sparrell 1993, Neal & Shapiro 1991, Wahlster 1991].

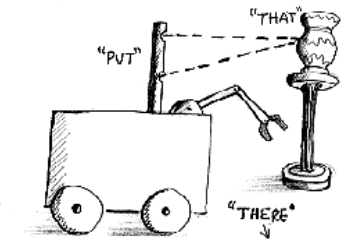


FIGURE 3-4. Cannon [1992] developed a robot that could understand deictic commands by triangulating camera orientation.

CUBRICON [Neal & Shapiro 1991] used typed and spoken sentences as input, along with deictic (pointing) mouse clicks to allow for interaction with a two-dimensional map. A similar system developed at the M.I.T. Media Laboratory [Koons et al. 1993] also uses a two-dimen-

SYSTEM	DESCRIPTION			INPUT				OUTPUT	
Authors	**Goal**	**Metaphor**	**Topic**	**Speech**	**Gesture**	**Gaze**	**Hardware**	**Visual**	**Auditory**
Bolt & Herranz 1992	Manipulation of 3-D graphics	One-way multi-modal	Graphics manipulation	Discrete word recognition	Iconic	Deictic	Gloves, head mic, head eye-tracker	3-D graphic objects	No
Koons et al. 1994	Arranging 2-D icons	Multi-modal dialogue	Firefighting/ 2-D map	Discrete word recognition	Deictic	Deictic	Gloves, head mic, head eye-tracker	2-D map with icons	Synthesized speech
Neal & Shapiro 1991	Information access	Multi-modal dialogue	Military activities	Discrete word recog. Typed NL	Deictic	No	Mouse, keyboard, microphone	2-D map w/ icons, deictic refs., text	Synthesized speech
Sparrell & Koons 1993	Arranging 3-D, graphical objects	One-way multi-modal	Furniture in a virtual room	Continuous recognition	Iconic	No	Datagloves, head mic	3-D graphic objects	No
Starker & Bolt 1990	Interest-responsive storytelling	User as observer, comp. as storyteller	Little Prince's planet	No	No	Deictic, attention	Table-mounted eye-tracker	3-D graphical world	Synthesized speech
Maes et al. 1994	Playful interaction in virtual worlds	Non-verbal interaction	Dogs, creatures and critters	No	Emblems, full body	No	Cameras	3-D graphics	No
Chin 1991	Help for line-command systems	Computer as tutor, user as student	UNIX commands	Typed NL	No	No	Keyboard	Typed NL	No
Jacobs 1990	Object selection	Augmented direct manipulation	Boats on a 2-D map	No	No	Deictic	Keyboard	2-D map with icons	No

TABLE 1-1. Comparison of recent systems that have employed a combination of gaze, gesture and/or speech/NL at the computer interface.

sional map using spoken commands (Figure 3-5), deictic hand gestures, but with the addition of deictic eye movement analysis [Koons & Thórisson 1993]. Starker & Bolt [1990] describe a system that used gaze as an indication of focus of attention and level of interest. Bolt & Herranz [1992] describe a system that allows a user to manipulate graphics with semi-iconic gestures. Koons et al. [1993] demonstrated how gestures can be very efficient for accomplishing many types of spatial manipulations within graphical worlds. Maes et al. [1994] employ a camera to capture the user's behavior and relieve the user from having to "dress up," at the cost of recognizing only symbolic gestures.

At the other end of the spectrum, Cannon [1992] designed a robot that could interpret speech and deictic gestures made with a camera (Figure 3-4). By pointing camera reticles at objects and locations and commanding the robot to "Put that there", the robot used triangulations and planning to execute acts communicated to it in this manner.

Bers [1995b] developed a system that allows the user to combine speech and gesture to direct a bee how to move its wings (Figure 3-6). Rather than mapping the body directly to the wings, the user communicates her intention to the system by saying "Fly like this", showing the wing action with either her arms, fingers or hands. The salient gesture (see below) is mapped onto the bee's body, making it move as prescribed by the user's pantomime. A user can do the gesture before, during or after the speech. The reason for this flexibility is that the system only allows the user to input one kind of gesture, thus bypassing the problem of gesture classification (see page 44).

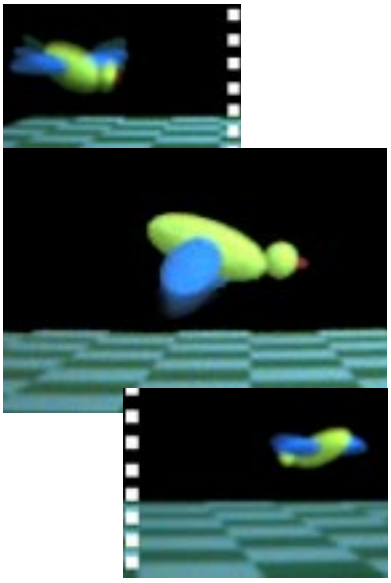


FIGURE 3-6. Bers' [1995b] bee could move its wings according to a pantomimic gesture example provided in a communicative fashion to the system.

Thórisson [1994] began to look at some of the real-time issues of multimodal dialogue by predicting turn constituent boundaries at run-time. This work, which was a precursor to the main contributions of this thesis, is described in detail in Chapter 6., page 81.

3.2.1 Multimodal Analysis and Interpretation

Analysis of Modes

In order to understand, or interpret, a coherent multimodal act, the multiple modes need to be brought together in some way. Such interpretation obviously draws on many resources, including speech recognition, gesture recognition, gaze-following, facial expression analysis, etc. In a free-form interaction, a major problem with gesture is finding which segments are of importance. Sparrell [1993] used a scheme based on a "stop-motion" analysis: whenever there is a significant stop or slow-down in the motion of the user's hand [cf. McNeill 1992, Kendon 1980], the preceding motion segment (called "gestlet" by the author) is

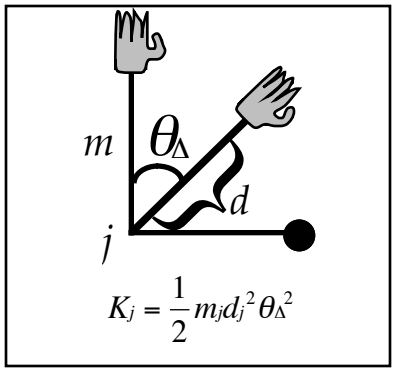


FIGURE 3-7. Kinetic energy of a moving body segment with one degree of freedom is found by looking at its connecting joint: Kj = kinetic energy of joint j, Θ_Δ = difference of joint angle j at t_1 and t_2, m_j = mass of body segment distal to joint j, dj = length of the segment; md = 1 for shoulder, 0.75 for elbow, 0.2 for wrist and 0.25 for fingers. Using a cutoff of 20 units, Bers [1995b] was able to select the meaningful segments of a pantomimic gesture from a stream of body motion data.

grouped and analyzed for lower-level features, such as finger posture and hand position. The interpretation would only happen after the user had finished his utterance. A similar approach was taken by Wexelblatt [1994], adding the ability to refine gesture-interpretation on the fly, as more "evidence" about a motion's trajectory was available. This system was not integrated into a multimodal system, but showed promise.

Bers [1995b] implemented a gesture segmentation scheme based on an orignial idea by the author that utilizes the kinetic energy of body part motion (Figure 3-7). In contrast to Sparrell's approach, this method allows for continuous gesture input. The system computes a "salience" map of body motion (Figure 3-8). Using a cutoff point for the "strength" of a motion, along with time stamping, motions can be selected that relate to the intended speech segment. This method could perhaps be extended to use body motion salience to predict the probability that a gesture is communicative, or to group together symmetric body motions with similar motion strengths.

Recognizing facial gesture has seen some progress in recent years [Essa 1995, Essa et al. 1994, Pentland et al. 1993, Turk & Pentland 1991, Bledsoe 1966]. Essa [1995], employing a camera to provide input to the computer, used optical flow methods to provide an analysis method of facial expression based on Ekman's FACS [1978] model of facial action. This system has not been integrated with other modes or used in an autonomous system.

Automatic intonation analysis has had a very short history and remains for the most part a topic unsolved [Thórisson 1993, Wang & Hirschberg 1992]. This is a problem that needs to be solved in order to create systems that can interact with humans using real-time speech. Speech recognition and natural language understanding have on the other hand been studied for a long time as part of linguistics and computational linguistics [Allen 1987] and will not be treated further here. It suffices to say that current natural language processing systems can have a vocabulary of thousands of words, can be speaker-independent and have a response lag of about 1-3 seconds. The main challenge in these systems remains dealing with brittleness resulting from lack of sensitivity to context and integration of multimodal cues to aid the recognition process.

Multimodal Integration & Interpretation

Koons [Koons et al. 1993] proposed the use of nested frames to gather and combine information from the modes. In his approach the speech is an initiator of gesture analysis: If information is missing from speech (e.g. "Delete that one") the system will search for the missing information in the gestures and/or gaze. Using time stamps, actions in various modes are re-united like pieces in a puzzle, to arrive at a coherent mean-

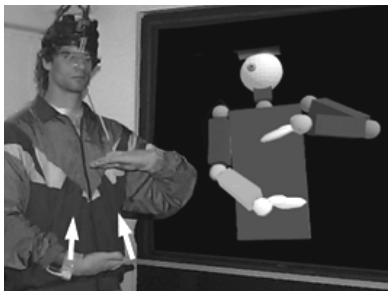


FIGURE 3-8. In this demonstration, salience of the motion of a person's body parts is shown as increased brightness in the corresponding body part of the marionette. Here, using kinetic energy, the gesturer's right-arm gesture (white arrows) is automatically separated from other incidental body motion.

ing. In a functionally similar system, Neal & Shapiro [1991] used a generalized augmented transition network[6] (ATN) that can receive input from a multi-media stream, instead of being limited to linear textual input. This system bypasses the complexities of free-hand gestures by allowing only deixis via a mouse. Others have used a similar method for simplification [Tyler et al. 1991, Wahlster 1991]. However, these put higher emphasis on the complexity of linguistic input allowing, among other things, the use of anaphora.

Compared to machine understanding of natural language, automatic multimodal interpretation is still a relatively undeveloped field. The missing parts include flexibility in interaction, use of cues from one mode to help interpret input from another. This will be discussed more closely in Chapter 5. on the computational characteristics of multimodal dialogue.

3.2.2 Missing Pieces in the Multimodal Metaphor

It may be argued that the main limitation of the multimodal interfaces to date stems from an incompleteness of the metaphor employed. Assuming that face-to-face dialogue is the generic model from which multimodal (and dialogue-based unimodal) interfaces draw [cf. Brennan 1990], one finds that key components are still missing from current implementations. Making this metaphor more explicit will make several things clearer for the user of a multimodal system, as well as for its designer. For example, an invisible, omnipresent agent (as opposed to an embodied one) makes it more difficult for users to pace the interaction and assess its progress at the turn-level [c.f. Clark & Brennan 1990]. Such limitations have been dealt with in various ways; the *Iconic* system [Sparrell & Koons 1994] employs an e-mail style of interaction (construct and send command ..... wait for response) to minimize failures in the interaction sequence. If the interpretation fails, a user has to wait an unknown length of time before re-issuing the command in full. In Wahlster's [1991] system the user selects the desired sub-type of deictic gesture from a menu of icons; the interpretation of the subsequent deictic gesture (a mouse click in a chosen region of the screen) is based on the type of icon selected. Although the interaction in systems such as these can not be called tool-level, it is not fully dialogical either—it seems to occupy a position somewhere between tool-based and communication-based metaphors. As will be argued throughout in this thesis, the critical features of face-to-face communication are not available to a user giving multimodal commands to a computer unless the computer has some command of human multimodal faculties and is explicitly modeled as an interactive agent.

6. See Chapter 6., page 84, for a discussion of the limitation of FSM-style approaches to multimodal interpretation.

To reiterate from Chapter 1, the missing pieces for a fully realized multimodal interface are:

1. Bridging between sensory input and action output,
2. continuous input over multiple modes,
3. integration of this multimodal input (in real-time), and
4. coordination of actions at multiple levels of granularity.

These will be the factors we focus on in the following chapters.

Agents, Robots & Artificial Intelligence

4.

We now turn to the literature on computer agent and human interfaces based on the idea of *agency*. In contrast to the previous two chapters, which dealt with the idea of agency in a general sense, this chapter presents it as it instantiates itself in what has been called *software agents* (c.f. quote in margin). A natural extension of the idea of sofware agents is the notion of a *social agent*, which are software agents, robots, or autonomous creatures that posses some social-interaction know-how—the kind of which we reviewed in the last two chapters—and can thus engage in social interaction with people on some level. First we will look closely at ways of embodying and presenting agents, and then review the most relevant work on agent architectures and artificial Intelligence.

"The idea of an agent originated with John McCarthy in the mid-1950's, and the term was coined by Oliver G. Selfridge a few years later, when they were both at the Massachusetts Institute of Technology. They had in view a system that, when given a goal, could carry out the details of the appropriate computer operations and could ask for and receive advice, offered in human terms, when it was stuck. An agent would be a 'soft robot' living and doing its business within the computer's world."

—Alan Kay (1984, p. 58)

4.1 The Agent Metaphor

Although the idea of software agents dates back to the sixties [Kay 1984], it is not until recently that the potential value of the agent metaphor for human-computer communication is becoming accepted [Hasegawa et al. 1995, Nagao & Takeuchi 1994, Rich et al. 1994, Maes 1994, Maulsby et al. 1993, Chin 1991, Laurel et al. 1990, Laurel 1990, Oren 1990, Crowston & Malone 1988].

Searching for an unambiguous definition of the term "agent" would be futile, but *a* definition is better than none. The Merriam Webster's Collegiate Dictionary has 4 different definitions for the term (side bar), none of which will suffice on its own. Kozierok and Maes [1993] define an agent as "a semi-intelligent, semi-autonomous system which assists a user in dealing with one or more computer applications." The definition used here is similar, but leaves out the reference to applications:

An interface agent is a metaphor for an agenda or a collection of task-level goals in the computer, imparted to it by the user, and the capability to carry out those, within reasonable expectations.

agent *n* (ME, fr. ML agent-, agens, fr. L, prp. of agere to drive, lead, act, do; akin to ON *aka* to travel in a vehicle, Gk *agein* to drive, lead) (15c) **1:** one that acts or exerts power **2a:** something that produces or is capable of producing an effect: an active or efficient cause **b:** a chemically, physically, or biologically active principle **3:** a means or instrument by which a guiding intelligence achieves a result **4:** one who is authorized to act for or in the place of another: as **a:** a representative, emissary, or official of a government (crown-) (federal -) **b:** one engaged in undercover activities (as espionage): spy (secret -) **c:** a business representative (as of an athlete or entertainer) (a theatrical -)

—Merriam Webster's Collegiate Dictionary, Tenth Edition

In addition to leaving out references to current computer applications, this definition slightly more specific than the one presented by Kozierok and Maes [1993]. It still contains direct reference to computers since computers are the only known synthetic entity to possess the necessary power to make anything close to what we intuitively might refer to as agents. The definition does not distinguish between kinds of intelligence or kinds of skills, and these will be addressed as we go on.

Not all agents in the real world are humanoid: dogs for example communicate via a subset of the human mutlimodal "command set". Since we are interested in the full range of multimodal interaction, the discussion will naturally focus on humanoid agents—those that bear a resemblance to humans in *appearance* and *skills*—as opposed to agents that resemble arachnids, insects or dogs. Such a distinction is necessary because so much of face-to-face communication is based on assumptions about skills (i.e. intelligence level, or competence) and appearance, both spatial and visual representation.

Agents represent thus the ability of the computer to accomplish something on behalf of the user [cf. Minsky & Riecken 1994]. To do this they posses high-level knowledge about a particular task domain or domains.[1] How the user conveys these wishes to the agent is an issue of human-computer interface design, and of course a central issue of this thesis. For example, Chin [1991] describes an agent that gives users advice about UNIX commands during interactive sessions. This system is a text-based natural language system using a keyboard as the input device and written English as the means of communication. Maes & Kozierok [1993] describe an agent that selects information from news sources depending on their relevance to what the user has found interesting in the past. These kinds of agents could be called terminal-based, because they rely on the traditional interaction methods of keyboard, mouse and monitor. For agents that can see and listen to the user, the issue is somewhat more involved.

1. The main reason for creating agents, and not simply making a suite of "tools" that one can select between, is that in addition to making the "tools" very sophisticated—i.e. moving toward their automation—we also want to automate the selection between these "tools". What inevitably merges out of such a creation is somethingone is hard-pressed to call anything but an "agent".

FIGURE 4-1. Cartoon illustrating the issue of embodiment and multimodal interaction.

When its owner addresses it, Tobor the vacuum cleaner turns in the direction of the speech and starts to decode the audio emanating from the human. The owner tells it to vacuum in a particular location, as indicated with a manual gesture. Miraculously, Tobor recognizes this as a deictic gesture and looks in the right direction even as the owner continues to speak. It then looks back when the utterance is finished. When asked if it understood, it nods enthusiastically.

A robot with such sophisticated communications skills still doesn't exist, but when it does I sure will buy one.

4.1.1 Agent Embodiment

"There is no place ... for a disembodied 'system' as a source of agency, communication, or collaboration: indeed, such disembodiment forces its mirror image on the participant and precludes the possibility of holistic response."

— Brenda Laurel (1992, p. 69)

The case for computer embodiment is most obviously seen in robotics, where the distinction between an agent and its environment is by default: a robot has a body that separates it from the rest of the world, and can thus be addressed and treated as an individual entity (see e.g. Brooks [1990] and Bares et al. [1989]). The issue of embodiment is important for face-to-face communication since the face/body system serves several functions, as we saw in the last chapter (page 37). The body parts that play the largest role in non-verbal communication are obviously the face, head, hands and, to some extent, the trunk. These have been treated thoroughly in Chapter 4. Here we will discuss two orthogonal issues of embodiment: *Visual representation* and *spatial representation*. Both play an important part in social communication. Visual representation includes the appearance of the agent—its physical form. Spatial representation is the kind of embodiment the agent has in the world and the way it can change its position in space.

4.1.2 Visual Representation

"With cartoon faces... becoming data measures, we would appear to have reached the limit of graphical economy of presentation, imagination, and, let it be admitted, eccentricity."

—Edward R. Tufte (1990, p. 142)

It may be argued that the most obvious and important form of embodiment for social interaction is a face. One of the earliest uses of facial expression to display machine status were the "Chernoff Faces" [Chernoff 1973]. Various features in a graphically generated face, like distance between the eyes, width of mouth, size of head, etc. were linked to variables in the status of a nuclear reactor: heat, pressure, etc. Since physical variables have nothing to do with human communication, this use of a face as a display method stands in strong contrast to the use of a face for social interaction, where its purpose is to facilitate dynamic, continuous exchange between the computer and its user.

While computer graphics work concerned with faces has to date focused extensively on their visual appearance, interactivity and effectiveness for information transmission has not been of primary concern. Convincing facial animation has proven to be a difficult task. A common limitation of physically-modeled faces [Essa 1995, Essa et al. 1994, Pelachaud et al. 1991, Waters 1990, Takeuchi & Nagao 1993, Waite 1989] is that the meaning of their expressions is often vague and a computer-controlled human face looks abnormal, even repulsive. An ideal solution to this would be to exaggerate the facial expressions, but within a physical modeling framework this may look unconvincing or awkward. An alternative is what might be called a "caricature" approach [Thórisson 1993a, 1993b, Librande 1992, Britton 1991, Laurel 1990] where details in the face are minimized and the important features exaggerated (see Hamm [1967] for an excellent discussion on cartooning the head and face). Brennan [1985] created a system that could automatically generate caricature line-drawings of real people from examples that had been entered by hand. Librande [1992] describes a system called *Xspace* that can generate hundreds of artistically acceptable two-dimensional drawings from a small example base. Simplified faces seem a very attractive alternative to physical modeling for animating interface agents, both in terms of computational cost and expressive power.

Another important issue in visual representation are the hands. Hands, as discussed above, can carry a lot of meaning and are also crucial in process control—directing the flow of the dialogue [McNeill 1992]. Again, details in the hands' representation below the gross anatomy level are not important for this purpose since crucial communicative information is generally not carried in their photo-realistic aspects.

Of primary concern in the visual representation of interface agents is the dynamic appearance of the agent: how it moves and reacts over time. This is even more important than static appearance, as we know from the qualitatively different experience of looking at people on photographs and interacting with them in real-time. Most of the work in this arena has been in animation [Sabiston 1991, Lasseter 1987, Thomas &

Johnston 1981], and adopted to a limited extent in interactive agent design [Bates et al. 1992]. This is an area that requires much more research and is closely linked with research on animal motor capabilities. In this thesis, a choice was made to use a cartoon-style representation of the agent (see "Character Animation" on page 203).

4.1.3 Spatial Representation

Giving a listening computer a spatial location makes it possible for its user to rely on conventions about "address" and point of view in his interactions[1]—something that is impossible if the computer listener is omnipresent.[2] And by making a computer agent situated in the real-world along with the user and the task at hand, a person can move between the agent and the task by virtue of social convention.

Common space between a user and a computer agent can be accomplished in two prototypical ways: The user can be brought into the computer's space, as is done in immersive virtual environments (Figure 4-3) where the user wears head-mounted goggles with stereoscopic graphics [Held & Durlach 1992, Sheridan 1992], or the agent can be brought into the user's world, as seen most clearly in robotics. This can be done by

FIGURE 4-2. Although HAL-9000's omnipresent fish-eye lens (on left) in *2001: A Space Oddyssey* [1968] proved highly effective for dramatic effect, ergonomists are quick to point out its inanimate embodiment and lack of visual feedback as troublemakers in a conversational interface.

1. This is true whether the computer's location is within the user's interaction space, such as in face-to-face conversation, or external, such as in a phone call.
2. From the human's perspective, of course, in other words, the user can make no assumptions can be made about the computer's visual or auditory "point of view."

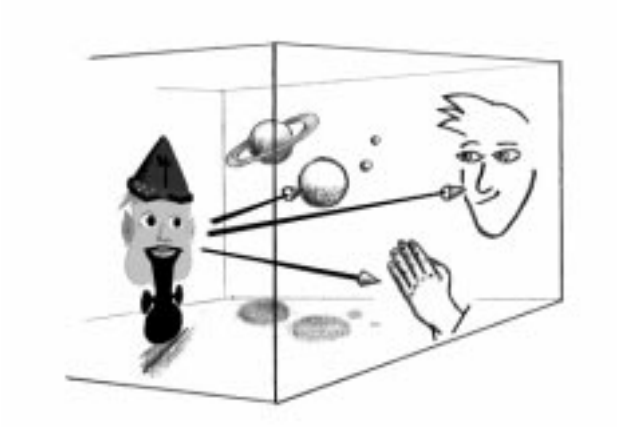


FIGURE 4-3. To achieve common space, the user can be brought into the agent's world

either making a robotic head (and body) or by allowing agent and objects to be displayed on separate monitors, placed at an angle to each other (Figure 4-4), or by giving the agent a physical body. These two methods are really two extremes on a continuum of ways to achieve integration between virtual and real worlds. Both extremes have their problems and virtues. Immersion allows both the user and agent to reference things and each other within the graphical world, eliminating the complexities involved in sensing and referencing real-world objects. It probably would be the method of choice for adventure games where the goal is total immersion and all the objects of interest are within the computer's world. A disadvantage of this approach is that the user has to "dress up" to have a common space with the agent. This precludes the agent from perceiving anything outside its own virtual world. The second option places an agent in real-space, which allows it to reference objects in the computer's world (although the reference space is now a 2-D projection) and still keeps open the option of referencing real-world objects, depending on the agent's perceptual prowess. One problem with this approach is the need to represent two distinct spaces: one within the workspace world and one in the real world. Another is sensing the surroundings and the user. However, if this can be done in a non-intrusive way, bringing the agent into the user's world offers more seamless integration of user-agent interaction with the user's work. This is the approach taken here.

The terminal-based interface agents to date have been represented visually by simple icons [Maes 1994, Maes & Kozierok 1993, Seth & Maes 1993], pre-recorded video clips [Laurel et al. 1990, Laurel 1990, Oren 1990, etc.] and presented inside windows on regular desk-top computers (or they have simply been hidden from the user's view [Mitchell et al. 1994, Sparrell & Koons 1994, Chin 1991]). Because of this, their spa-

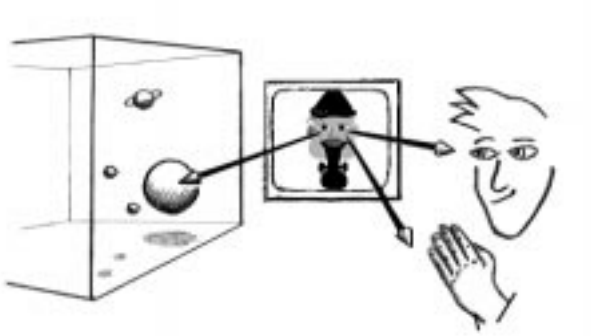


FIGURE 4-4. The agent can be brought into the user's world by giving it a physical embodiment such as a screen.

tial position has generally had no function at all. This is in part because of a lack on the machine's side to localize the user (and itself) in real-space. Another inherent complication is that representing both agent and work space in the same 2-D plane makes agent actions that rely on three-directional cues—such as deictic gestures of gaze and hands—difficult save for the simplest cases.

An exception to a history of a single 2-D plane representation was a system by Schmandt et al. [1985] employing speech recognition, called the Conversational Desktop. Their system employed space sensing technology to demonstrate how directionality—one of the cues for inferring "address"—plays a role in communication: If you are turned toward someone when speaking an utterance, chances are the utterance is meant for him or her. The system would only listen to the user's speech if s/he was turned to the computer screen. By giving computers information about spatial layout of users and objects—including themselves—the agents' glances and deictic gestures, as well as "point of view," can begin to have meaning in the context of the interaction.

4.2 Agent Architectures

Agent design in AI has mainly been in the area of robotics, where a physical entity—often mobile—is used as a testbed for the development of control strategies. Approaches taken to date can be classified into two categories, "classical AI" and "behavior-based AI" (c.f. Maes [1990b]). As Brooks [1990] has pointed out, even though a happy marriage of the two has yet to come about, the approaches are somewhat complementary and as I will argue later, both have features to offer for

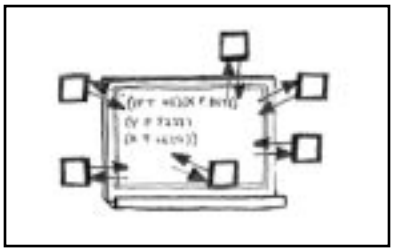


FIGURE 4-5. A blackboard serves as the common storage of intermediate and final results produced by a collection of independent processing modules (small squares) or 'Knowledge Sources.'

social agent design. Here we will look briefly at 7 system architectures, three from the classical AI pool, two from the behavior-based one, and two hybrid.

4.2.1 Classical A.I.

Blackboard systems were designed to handle unpredictable information like that encountered in speech recognition or planning [Hayes-Roth et al. 1988, Nii 1989]. This architecture is relevant here because it provides potential solutions to some of the problems multi-modal interfaces present, namely those of multiple levels of detail, multiple data types and high variability. The blackboard architecture attacks the problem of unpredictability by the use of a common data storage area, or blackboard, where results of intermediate processes, or knowledge sources (KS), are posted and can be inspected by other processes working on the same problem (Figure 4-5). Indeed, the problem of social behavior control includes some of the same problems as automatic speech recognition, where information on many levels—phonemic, lexical, syntactic, semantic, discoursal, pragmatic—can come to bear on the recognition process [Allen 1987]. HEARSAY [Reddy et al. 1973] was the first system to apply this architecture to a real-world problem. It consisted of multiple knowledge sources, each designed for recognizing and classifying a specific feature of natural speech. The system, and its modified version, HEARSAY-II, were designed to exhibit the following properties absent in prior systems [Nii 1989, p. 21]:

1. *The contribution of each source of knowledge (syntax, semantics, context, and so on) to the recognition of speech had to be measurable.*
2. *The absence of one or more knowledge sources should not have a crippling effect on the overall performance.*
3. *The system must permit graceful error recovery.*
4. *Change in performance requirements such as increased vocabulary size or modifications to the syntax or semantics should not require major modifications to the model.*

The interesting points to notice here are 2, 3 and 4. These do not only apply to requirements for speech recognition systems: they apply to any system that is to function semi-autonomously in a dynamic environment. Variations on the original version of the blackboard architecture have been successfully applied to areas such as vision and distributed computing [Nii 1989]. Jagannathan [1989] discusses approaches to applying blackboard systems to real-time applications. Modifications to the original versions for this purpose include mechanisms to allow interleaved execution of subsystems, as well as communication between them [Fehling et al. 1989], resource management, speed/effectiveness trade-off and reactive systems behavior [Dodhiawala 1989].

Another system using traditional AI methods is Chin's [1991] UCEgo. This system is an addition to a natural language UNIX consultant system (UC) that gives advice to users about commands and command options. The system's task can generally be described as that of goal detection and maintenance, using traditional planning techniques. The main difference between this system and the others discussed here is that it is specifically designed to interact with humans. The interaction is of the step-lock, unimodal kind, via a teletype. An example of interaction between a user and the system is shown in Figure 4-6.

> What does who -b do?

who does not have a -b option

>What does ruptime -t do?

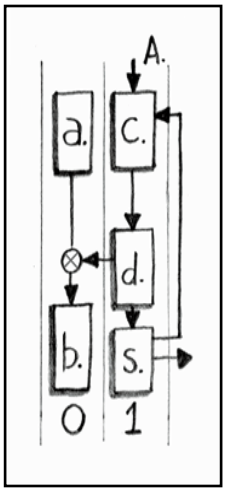
I'm sorry, I do not know that.

FIGURE 4-6. Example of a user-agent interaction in the UCEgo system [from Chin 1991]. The user's input starts with a >.

A third architecture in the classical AI category is Schema Theory[1] [Arbib 1992], which is historically an outgrowth form blackboard systems. Schema theory is an attempt to deal with the complexity of large systems that interact with the real world. A schema is both a storage of knowledge and the description of a process for applying that knowledge, and in this respect bears both resemblance to blackboard architectures' knowledge sources and Maes' competence modules (see below). This system postulates a set of basic perceptual and motor schemas that provide simple, prototypical perceptual capability and movement patterns. The schemas are combined to form assemblages of coordinated control programs which interweave their activations in accordance with the current task and sensory environment. Schema activations are largely task-driven, reflecting the goals of the organism and the physical and functional requirements of the task. As Arbib [1994, 1992] has noted, the generality of schema theory puts most distributed and layered systems, such as Minsky's Society of Mind [1989] or Brooks' *Subsumption* architecture [1990], under its umbrella.

4.2.2 Behavior-Based A.I.

As an example of the situated action or the behavior-based approach (cf. [Brooks 1991, 1990, 1986], [Meyer & Wilson 1991] and [Maes 1990a]), Brooks [1990] proposed what he calls a subsumption architecture where low-level behaviors of a robotic agent can be subsumed by higher-level, later-designed behaviors (Figure 4-7). This allows for incremental development of robot skills and a robustness that is difficult to achieve with traditional methods. Another example is Maes' [1989] architecture that is based on *competence modules*—software modules that contain enough information to execute a particular behavior from beginning to end (Figure 4-8). The modules are connected together by activation links that control their sequence of execution. The input to the modules can come both from internal goals and the environment. This architec-



GURE 4-7. In this example, a bsumption architecture has been ilt for a tour-guide robot. Level 0 ntains the behavior modules *local-nder* (a) and *move* (b). Modules in vel 1 include *up-counter* (c), *ndmark-list* (d) and *speak* (s), which tputs spoken information. The ndmark-list module suppresses the bot's wander behavior (valve arked x) so that it ends up ccessively at each landmark, and ggers the speech for each one as propriate, while the up-counter eps track of which landmarks have en visited. (Adopted from Lyons & endriks [1992].)

1. Arbib's Schema Theory should not be confused with Shank's scripts [Schank 1990, Schank & Abelson 1977], sometimes also referred to as schemas, which is a construct invented for modelling human memory and production mechanisms for stories.

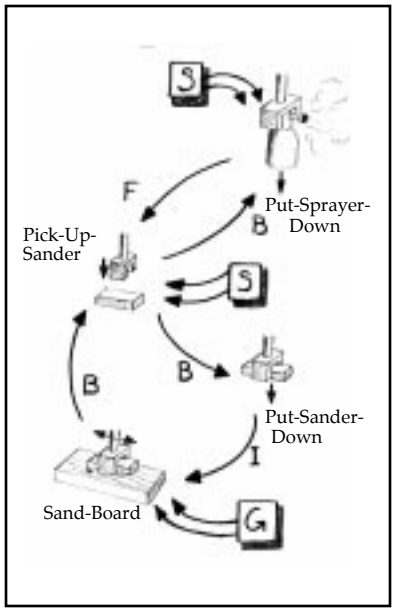


FIGURE 4-8. Toy example of the interaction between goals, states, inhibition links and spreading activation links in Maes' system [1990c]. The initial situation is {sprayer-in-hand, sander-on-table} a top of figure, and the initial goal is {board-sanded}. (B = backward spreading activation, F = forward, I = inhibition, G = goals, S = state. Adopted from Maes [1990c].)

ture, and other reactive approaches [Wilson 1991, Steels 1990, Agre & Chapman 1987] are very good for effective action selection, and they allow their robots to learn over time. However, they lack methods to deal with external and internal time-constraints and are limited in the planning they can handle.

4.2.3 Hybrid Systems

The traditional approach to the sensory-motor problem is a three-part system with a pipelined architecture. The sensory input feeds into the sensory module, which is followed by the cognition module, which is followed by the motor module. This pipelined approach has recently been questioned by the behavior-based AI research. However, hybrid systems trying to combine the best of both approaches have been few. One approach is worth mentioning, though. It is the NASREM architecture [Albus et al. 1987]. This model tries to incorporate knowledge gleaned from animal research on sensory-motor capabilities and integrate this into a comprehensive scheme for autonomous and tele-robot control. The system contains multiple levels of processing, each level containing the three components of sensory processing, world modeling and task decomposition. A global data storage is accessible from any level, but sensory modules also receive information from the level below, and task modules receive input from the level above. There are five levels all together: *Mission* (information relating to a full mission), *service* (information related to parts of a mission), *task* (sub-components of a service), *elemental move* (transition from symbolic commands of movements to spatially-defined commands), *primitive move* (generates smooth trajectories) and *servo* (simple hardware control). Although the layered approach of this model sounds promising for achieving the best of both worlds—fast responses to time-constrained events and slower responses to less time-constrained events—it has been criticized for trying too hard to encompass all possible systems, and thus losing its descriptive power [Thorpe 1992].

Of particular interest here is Cassell et al.'s [1994a, 1994b] system for automatic speech and gesture generation. The system employs two computer drawn human-looking characters that interact with each other (in non real-time) using speech, gaze, intonation, head and manual gesture. The system employs what the authors call PaT-Nets (Parallel Transition Networks; Figure 4-9) in which synchronization between gestures and speech is accomplished as simultaneously executing finite state machines (FSMs). (See page 84 for a discussion of the limitations of an FSM-based approach.) While the system is focused only on the generation of multimodal acts and is not concerned with the complications resulting from temporal constraints in perception, action planning and execution, it provides an insight into the complexities of synchronizing various levels of multimodal action generation, from the phoneme level up to the phrase and full utterance.

FIGURE 4-9. A PaT-Net for generating gaze movements. Nodes specify actions; transitions between nodes are both conditional and probabilistic. All leaf nodes branch back to the root node unconditionally (adopted from Cassell et al. [1994]).

4.3 Summary

We have now covered background material in two areas: multimodal research, in the previous chapter, and AI and agent-based systems in this chapter. Embodiment has been dealt with somewhat in the robotics literature, perhaps because disembodied agents are more common in this area than in psychology. Whereas the psychological literature tends to be descriptive, the computational approaches focus on both descriptive and prescriptive models. As of yet, computer implementations are mostly concerned with getting something to work, as opposed to modelling human face-to-face interaction correctly, and at all levels, but this may simply be because the field is relatively young. Robotics and cognitive science research has made several contributions relevant to the task of full-duplex feedback, among them the blackboard architecture and the behavior-based approach to planning. However, this work needs to be adapted to the task of generating face-to-face computer systems. The next step is then to characterize the specifics of multimodal interaction to make this possible.

Computational Characteristics of Psychosocial Dialogue Skills

5.

Why is a multimodal interface not just a relapse to the old idea of the teletype where a user would ask the system "questions" or tell it "commands" and the system would reply? Is it really so different? Why, yes. As will be touched on several times throughout this thesis, the difference lies in the interaction itself. Instead of restricting the interface to a vending-machine paradigm, with the call-answer sequence only happening at one level, multimodal interaction calls for a different model where interpretation and action response are intimately tied together, forming a multi-layered interaction space between the user and machine: actions are generated in response to events that happen on various time-scales; these happen in parallel with perceptual and interpretive actions.

```
C:\> cd myfiles\newfiles\cool
Invalid directory
C:\> why?
Bad command or filename
C:\>_
```

FIGURE 5-1. The command line interface has often been incorrectly exemplified as a typical dialogue system.

As explained in the previous chapter, this paradigm contrasts sharply with the computer-as-a-tool metaphor in that the computer is viewed as a dynamic entity, as opposed to a non-acting, dead tool. As pointed out by Laurel [1990], a computer behaves. The actions of computers are sometimes so complex that we cannot not understand them as simply as the light turning on when we flip the switch—we perceive it as if the computer had an agenda of its own. This trend is becoming clearer every year, with ever-increasing complexity at the interface.

Thus, the multimodal metaphor is different both from the old teletype interface and the currently popular object/tool metaphor. When successful, it will feel to the us, computer users, as different as the experiences of hammering in a nail and talking to our children.

We can now begin to take a closer look at the issues behind psychosocial dialogue skills and the unique problems that result from what can be called a holistic approach to multimodal dialogue. We will look at the arguments behind four claims about the process of multimodal interaction:

1. To produce coherent behavior in real-time dialogue, *reactive* and *reflective* behaviors have to co-exist in the same system,
2. analysis of the *contextual function*[1] of speaker actions and control of the process of dialogue are intimately linked through what I refer to as *functional analysis*,
3. the information necessary for *correct and efficient content analysis* is also the necessary information for providing *correct and efficient multimodal feedback behavior*, and
4. tracking of dialogue state should be at the top of the sensory-activities list.

The first relates to the integration of real-time and less-real-time actions, and is dealt with in section 5.2; the second relates to the nature of multimodal processing and is discussed in section 5.3; the third is a derivative of the third and is supported in section 5.3.1. The fourth deals with the priorities of a communicative agent's sensory processes and how this relates to turn-taking, and is found in section 5.4.

In section 5.5 we will look at the important issues of morphological and functional substitutability. At the end of the chapter a layered model of face-to-face dialogue will be presented.

But first we will try to tease out the features of face-to-face dialogue necessary for a computational model.

5.1 Challenges of Real-Time Multimodal Dialogue

Features

From a computational perspective, many features set real-time face-to-face interaction apart from other topics in human-computer interaction and artificial intelligence [Thórisson 1995b]. For the current purposes, these may be identified as:

1. *Incremental interpretation,*
2. *multiple data types,*
3. *seamlessness,*

1. My use of the term "function" is roughly equivalent to its use in speech act theory [Searle 1975, 1969], i.e. as the goal-directed use of communicative acts in context, and is thus close cousin to Austin's [1962] "illocutionary acts", albeit broader. See also Searle's [1971] discussion of function-indicating devices and Silverstein's [1987] treatment of function. The term "contextual" refers to the effect dialogue context, or "state", can have in determining an action's function.

4. *temporal constraints,*
5. *multi-layered input analysis and response generation, and*
6. *functional substitution/morphological substitution.*

1. *Incremental Interpretation.* Multimodal interpretation is not done "batch-style:" There are no points in an interaction where a full multimodal act or a whole sentence is output by one participant before being received by another and interpreted as a whole. Interpretation of multimodal input happens in parallel with multimodal output generation.

2. *Multiple Datatypes.* Multimodal interaction contains many data types, as any quick glance at the human communication modes will show: Gestures [McNeill 1992, Goodwin 1986, Ekman 1979, Ekman & Friesen 1969] provide metric (spatial and spatio-relational information), speech [Allen 1987, Goodwin 1981] provides lexical tokens, semantic and pitch (prosodic) information [Pierrehumbert & Hirschberg 1990], gaze [Kleinke 1986, Argyle et al. 1974, Kahneman 1973], head and body provide directional data related to attention and the dialogue process. These data are both Boolean and continuous over various ranges.

3. *Seamlessness.* When interacting with each other, people generally do not realize that interjecting an iconic gesture into the discourse constitutes a different kind of information than a deictic one, and they don't particularly notice the mechanism by which they take turns speaking. The various data types encountered in face-to-face dialogue have to be combined into a coherent system to allow for seamless multimodal interaction.

4. *Temporal Constraints.* The structure of dialogue requires that participants agree on a common speed of exchange [Goodwin 1981]. If the rhythm of an interaction is violated, it is expected that the violating participant make this clear to others, at the right moment, so that they can adjust to the change. This speed sets an upper limit to the amount of time participants can allocate to thinking about the dialogue's form, content, and to forming responses. (See "Temporal Constraints" below.)

5. *Multi-layered Input Analysis and Output Generation.* In discourse, responses in one mode may overlap another in time, and constitute different information [McNeill 1992, Cassell & McNeill 1990, Goodwin 1981]. The layers can contain anything from very short responses like glances and back channels, to tasks with longer time spans, such as whole utterances and topic continuity generation. In order for purposeful conversation to work, reactive and reflective[2] responses have to coexist to provide for adequate behavior of an agent.

6. Functional substitutability refers to the phenomemon when *identical looking acts can serve different dialogical functions.* Morphological substitutability is the reverse: *Different looking acts can serve the same function.* We will look at this closer in Section 5.5.

Assumptions about the Nature and Quality of Input

When trying to incorporate the above principles into the design of artificial agents, it becomes apparent that certain additional characteristics of the human interpretive processes and quality of "input data" have to be taken into consideration:

1. *Interpretation is fallible.*

Because of inaccuracies in the information delivery of humans, among other things, there will be errors in the interpretation no matter how powerful our interpreter is, whether human or artificial. This problem is worsened in artificial agents by the use of faulty sensors, occlusion when using cameras, background noise masking audio signal, etc.

2. *There are both deficiencies and redundancies in input data.*

It is an inevitable fact that we have to deal with missing information, and, in certain cases, redundancy as a possible solution, both in interpretation and output generation.

3. *Sensory data is collected to allow an agent to produce action or inaction.*

This reflects purpose-directed sensory and cognitive abilities of any situated agent, and prescribes an ego-centered design when producing social behavior in machines.

4. *Behavior is based on data from multiple sources, both internal and external, including dialogue state, body language, etc.*

In multimodal communication action can be taken—and perhaps most often is—based on more than a single piece of information.

5. *behavior is eventually always produced, no matter what data is available.*

Both a listener and a speaker in dialogue are expected to exhibit the necessary behaviors to allow the other to take the necessary steps for clarifying, modifying, and, in general, following the pace of the interaction.

2. This is the issue of how much time one has available for planning a response to a situation. Intuitively, the terms *reactive* and *reflective* refer to fast and slow responses, respectively. There is also a more specific meaning that will become apparent in later chapters. See "A Notation System for Face-to-Face Dialogue Events" on page 108.

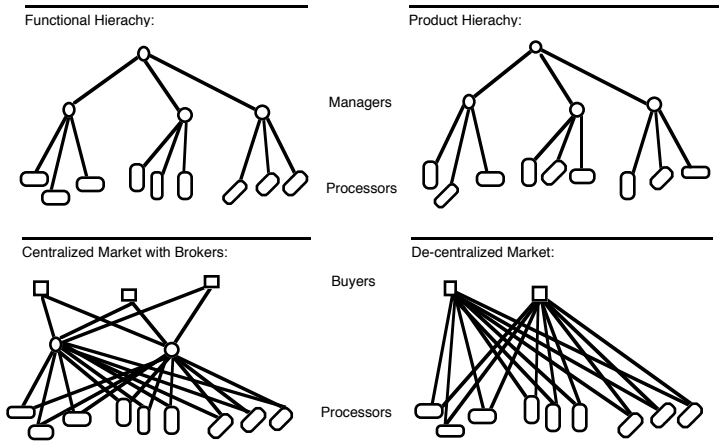
5.2 Temporal Constraints

A useful tool for viewing time-constraints of dialogue is Coordination Theory [Malone & Crowston 1991, Crowston et al 1988, Malone et al. 1988]. Coordination Theory classifies coordination mechanisms broadly into two categories: markets and hierarchies (Figure 5-2). Generally speaking, markets have a relatively high coordination cost and low production cost, whereas hierarchies are the opposite. According to the theory, an object is highly asset specific if it is constrained by extraneous factors, such as place, knowledge, or time. For example, eggs are highly time-specific because they will lose their value if not delivered and consumed before they go bad. Malone et al. [1988] have noted that any highly specific asset is more likely to be handled through a hierarchy. Having seen an example of the time-specificity of dialogue behavior in Figure 1-1 (page 20) it would be natural to choose a hierarchically organized system for its coordination. (We will come back to this issue in Chapter 7.)

As Dodhiawala et al. [1989] have pointed out, real-time performance is not just a matter of speed. They have identified the following four aspects of real-time performance:

- **A.** *Responsiveness:* The system's ability to stay alert to incoming information.
- **B.** *Timeliness:* The system's ability to manage deadlines.
- **C.** *Graceful adaptation:* The system's ability to reset task priorities in light of changes in resources or workload. We should also include under the last part the need to rearrange tasks when problems arise, e.g. the missing of deadlines.

FIGURE 5-2. Four kinds of coordination methods (after Malone et al. [1988]). The mechanisms controlling dialogue behavior are most likely arranged in a product hierarchy.



D. *Speed*

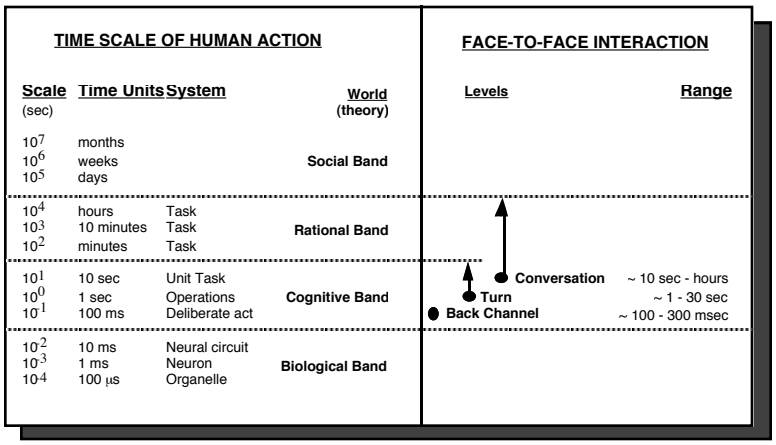
Simply stated, the issue of *speed* may be split into three stages of processing: *speed of analysis* (or perception), *speed of decision*, and *speed of action*. Face-to-face conversation is unique because it contains processes that span as much as 5 orders of magnitude of execution[3] time, from about 100 ms to minutes and hours[4] (Figure 5-3).

As mentioned in Chapter 1., face-to-face discourse [Goodwin 1981] contains rapid responses and more reflective ones interwoven in a complex pattern. This kind of interaction is the basis for the dialogue management system proposed. It calls for an architecture that is responsive to the environment yet is capable of longer-term planning. This is referred to here as "combining reactive and reflective behaviors."

This leads us to claim one:

{1} *To produce coherent behavior in real-time dialogue,* reactive *and* reflective *behaviors have to co-exist in the same system.*

FIGURE 5-3. Comparison between the timing in face-to-face interaction and the time scales of human action as classified by Newell [1990] (from Thórisson [1994]).



3. Notice we are talking about the sense-act cycle, not just motor response.
4. We should say from 0 ms since turn taking is often seen happening with no pauses between speakers. Such phenomena obviously would require some sort of prediction mechanism (if we want above-chance performance) since simple reaction time in humans is typically in the >100ms range and choice reaction time in the >300 ms range [Coren & Ward 1989]. Although it has been shown that prediction mechanism are at work in human dialogue [cf. Sacks et al. 1974], they will not be dealt with here. Suffice it to say that predictive mechanisms could easily fit into the model proposed (Chapter 7.).

This issue is a large one, and one that †mir, presented in Chapter 7. & Chapter 8., provides a solution to.

5.3 *Functional Analysis: A Precursor to Content Interpretation and (sometimes) Feedback Generation*

> ...if participants are to use each other's bodies as sources of information about their talk they are faced with the task of distinguishing relevant body behavior form that which is not. ...such classification is not simply a hidden cognitive process, but one that has visible consequences for the actions of the party doing that analysis.
>
> —Charles Goodwin (1986, p. 29)

Since any multimodal system works under time-constraints, the natural way to proceed with analysis of the environment is to extract the most important information first. But what constitutes the most important information? How do we look for it? The claim here is that this information is the *function of discoursal actions*, and we look for it using a system of specialized processes that have a relatively high speed/accuracy trade-off.[5]

Initial (basic, elementary) interpretation of a speaker's behavior[6] should not primarily be concerned with what lexical elements can be best mapped onto the user's utterance, or whether the utterance at any point in time is grammatically correct.[7] It should be concerned with distinctions that determine broad strokes of behavior, i.e. extracting the features that make the major distinctions of the dialogue. For example, computing answers to a questions like "is this person addressing *me?"* would be a necessary precursor to start listening. Likewise, answering the question "is the person *pointing*?" would have precede looking in the direction of the pointing arm/hand/finger to find what is being pointed at. These examples constitute analysis of high-level *function*. Computing functions of multimodal actions thus precedes processing the information that is being conveyed.

On the feedback generation side, a listener's behavior of looking in the pointed direction is a sign to the speaker that he knows that her gesture is a deictic one, and that he has correctly extracted the relevant direction from the way her arm/hand/finger are spatially arranged. The gaze behavior resulting from correct functional analysis serves double duty as direct feedback (in this example at least), and constitutes therefore efficient process control.[8]

5. To say that a process ha a high speed/accuracy trade-off simply means that it is more important for that process to provide output in a timely fashion than to be absolutely certain of the accuracy of its output.
6. The claim is made for both computer and human interpretive processes.
7. A similar point is made by Winograd [1988].

Functional analysis—determining the function of a multimodal action—is thus a necessary initial step to both content analysis and correct feedback generation. Let's look at another example, using only the speech mode.

The following exchange may look perfectly fine:

A. *So, aliens ate my Buick.*
B. *I'm so sorry to hear that!*

until we add the accompanying intonation, which goes up at the end of the word "Buick" as indicated with a question mark:

A. *So, aliens ate my Buick?*
B. *I'm so sorry to hear that!*

We find B's response inappropriate and would infer that B thought A was making a remark, not asking a question. If B had "computed" the correct function for A's utterance, (i.e. eliciting information—a question) her response would probably have been different, along the lines of "No, silly!" or "I wouldn't know."

UTTERANCE*	GESTURE**	PROCESS
Speaking	Gesturing	Paying attention
Assertive	Deictic	Addressing me
Directive	Iconic	Giving turn
Commissive	Pantomimic	Taking turn
Declarative	Symbolic	Wanting turn
Expressive	Butterworth♥	
Interrogative	Self-adjustor	
Back channel	Attention-grabber	
Filler♠		

TABLE 5-1. Some main high-level functions of multimodal actions in dialogue. It may be noted that most of a user's utterances directed to an interface agent would probably be directive (commands) and interrogatives (questions).

*See Searle [1975] for a treatment of speech acts. **This applies to both facial and manual gestures [Rimé & Schiaratura 1991, Ekman 1979]. ♠Also referred to as "filled pause;" utterances like "aaaah" and "uuuuuh." ♥The gestural equivalent to filled pauses.

8. It would also be correct and efficient feedback if an agent erroneously concluded that the gesture was iconic and therefore looked at the speakers hand instead, since this would clearly indicate to the speaker the error made. In this case generation of the correct feedback coincides with the actions necessary for further interpretation of the input.

This leads to our second and third claims:

{2} *Analysis of the* contextual function[9] *of speaker actions and control of the process of dialogue are intimately linked through* functional analysis.

{3} *The information necessary for* correct and efficient content analysis *is also often the necessary information for providing* correct and efficient multimodal feedback *behavior*.

The strength of these claims lies in the double support they provide for extracting function before interpreting.

Table 5-1. shows the main functional categories found to date in multimodal dialogue. The issue of functional analysis is neither one of computational power, nor of top-down/bottom-up processing; it is an issue of sequencing. Nothing prevents the use of either top-down or bottom up analysis to extract functional attributes of a speaker's behavior, and adding computational power will certainly speed up the process of analysis. But neither will eliminate the sequential dependency between the two steps of determining an action's function and analyzing its (possible) meaning(s). The second reason why this dependency is important is simple: More assumptions can be made with global information than local—an agent can do a lot more with general information when details are missing than with detailed information when the global perspective is lost.[10] By giving the high-level functions, such as those in Table 5-1., highest priority, the most useful responses can be generated even if other information is missing, resulting in increased robustness.

The functional aspects of multimodal behavior can, and should, be extracted by means of multimodal analysis; any feature, body part, intonational cue or even lexical analysis could assist in the process. A major part of creating multimodal computer agents is finding how to pull out the necessary information.

5.3.1 The Link Between Functional Analysis and Process Control

As we saw in the pointing example, correct and relevant feedback generation often follows automatically from correct functional analysis, but only if we fulfill two conditions. The first is that the behaviors pro-

9. See footnote page 66 for a treatment of "function."

10. This can be seen by a simple example: If I know that I have just been asked a question, but missed some of the words, I can exclude all forms of utterances except questions from consideration and ask the speaker to repeat, increasing the probability of correct interpretation significantly. This does not work in the other direction.

duced by the system be guaranteed execution within a given time limit, as determined by the pace of the dialogue. Because dialogue state is constantly shifting, we need a mechanism that ensures that behaviors be executed at the time they are relevant—not before and not after.

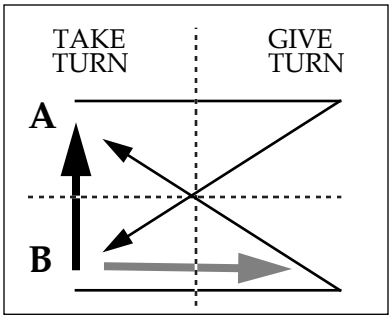


FIGURE 5-4. The problem of efficient turn taking includes detecting the correct transition points. In this figure, A and B are participants in a dialogue. Thin arrows demonstrate smooth turns; solid bold arrow constitutes an interruption (of B by A) with the possibility of overlapping speech, gray bold arrow shows a failure of the listener (A) to take the turn when it is given (by B), possibly with an unwanted silence.

The second condition we need to fulfill is that we model the agent in our own image, i.e. with a head, face, gaze, arms, hands, and a body—organs that have to do with communication. This is because in face-to-face interaction, sensory organs, bodily constraints, attentional and other mental limitations are linked together in a way that is intimately integrated and intertwined with the dialogue process. This provides dialogue with an intricate feedback mechanism, the absence of which has been shown to disrupt discourse [Nespoulous & Lecours 1986].[11] In other words, if any parts of this mechanism are broken or missing, dialogue may break down.[12] It may be added that providing an agent with misleading actions or visual features may of course lead to the same results.

5.4 Turn Taking

As we established in Chapter 3., a key element of dialogue is turn taking. Sacks et al. [1974] maintain that the purpose of the turn taking system is to minimize overlapping speech and pauses in interaction. When we refer to the *seamlessness* of dialogue, we are essentially referring to a collection of mechanism grouped under this hat. Figure 5-4 shows the possible outcomes of turn taking with two participants. The difficulty of correct turn taking by machine lies first and foremost at the perceptual/knowledge level, because a participant has to infer what constitutes a valid turn-giving signal for each role. Generating that signal for the speaker is, on the other hand, simple. So computational turn-taking modelling is first and foremost a perceptual problem.

As numerous researchers have shown [Walker & Whittaker 1990, Goodwin 1986, Sacks et al. 1974], turn taking defines the two main roles of conversants: listener and speaker. Each role calls for its own repertoire of behaviors and perceptual tasks. My proposal is to define two very different classes of behaviors, both of which include percep-

11. Nespoulous & Lecours [1986, page 61] say: "... Dahan [see ref., op. cit.] convincingly demonstrated that the absence of regulatory gestures in the behavior of the listener could lead the speaker to interrupt his speech or to produce incoherent discourse."

12. It is also possible that some violations can be fixed with clever engineering of the agent behavior, its visual appearence or its environment. This topic will be revisited in Chapter 11.

tual, decision and motor tasks, that participants in a dialogue have to switch between. Thus, for the period that person A takes the role of listener, one can expect him to be engaged in a set of mental activities—mental activities that are different from those he is engaged in when in the role of speaker. To take an example, Goodwin & Goodwin [1986] discuss the activity of searching for a word and how this can be a cooperative activity. A speaker may indicate to her listener, using gaze and body language, that she is looking for a word. The listener will offer to assist in the search by interjecting plausible words. Although the process is cooperative, it is the speaker who has the turn, and thereby the power to accept or reject the listener's suggestions. In each role it is not only the behavioral repertoire that is different but also the demands on the participant's perceptual and decision-making systems. The roles can be thought of almost as roles in a play; they are part of the same plot but the rules for each character are very different. According to this proposal, a speaker's perceptual system is preoccupied with monitoring the progress one is making in the narrative production of output, what little is left of attentional capacity is spent distinguishing between acts of the listener that are insignificant to the dialogue (such as the listener scratching himself) or constitute communicative actions, such as a wish to interrupt. The listener's role revolves around interpreting what the speaker is saying and making sure she knows that he is following her, as well as interrupting when problems arise.

This emphasis on the listener-speaker distinction has the important effect of putting the tracking of dialogue state at the top of the sensory-activities list. It is a process that happens at the decisecond level of granularity (see Figure 1-1 on page 20) and as such has a relatively high speed/accuracy trade-off—i.e. it is highly temporally constrained.

This leads us to claim four:

{4} *Tracking* dialogue state *is at the top of the sensory-activities list.*

But what should these sensory activities be?

5.4.1 A Situated Model of Turn Taking

The model of turn taking advanced by Sacks et al. [1974] is very broad and can be considered to be about as good as a descriptive model of turn taking can get by just using data from human observation and video tape analysis. To design a system that can actually generate turn taking behavior and exhibit the rules described in their model, one needs to make several decisions about the nature of the underlying perceptual mechanisms. Here, two hypotheses are put forth for making the creation of such a system possible.

Earlier we claimed that functional analysis of multimodal actions is necessary for providing correct multimodal feedback (page 71). The need for reactive responses puts definite time constraints on this analysis that have to be met for the system to work. This leads us to hypothesis 1:

1. Reactive behaviors[13] are based on opportunistic processes: *High-speed functional analysis in multimodal dialogue draws on cues from any number of number modes, as long as they are informative*.

We also need to specify how the information from various modes is combined. The second hypothesis is this:

2. *Features extracted from a particular multimodal speaker action are logically combined (in the mathematical sense of the word) by the listener to arrive at a plausible dialogue function for that action* [cf. Duncan 1972].

To relate this back to the issue of the speed/accuracy trade-off in perception and action, according to these hypotheses, an increased number of features and modes included in a single analysis will strengthen the *accuracy* of that analysis, but do not affect its *speed*. Thus, the reliability of the turn-taking process should increase with an increased number of cues, but the speed of analysis will not change. However, and here is the rub, increased reliability may affect the speed at which the extracted functions will be *acted on*. Thus, upon interpreting the multimodal act "He went [deictic manual gesture & gaze] that way," a listener may look sooner in the relevant direction if the manual pointing gesture is present, than if she only has gaze as an indication of direction, since a manual deictic gesture is a more reliable indicator of direction than gaze alone.

No claims are made here whether this model is "true"—that remains to be answered by experimentation. It enables us, however, to start building a system that allows such experimentation to take place.

5.5 Morphological and Functional Substitutability

- Morphological[14] substitutability[15]: *Different looking acts can serve the same function.*
- Functional substitutability: *Identical looking acts can serve different functions.*

13. The terms reactive and reflective are dealt with in Section 5.2, page 69.
14. The term "morphological" is used here as relating to "form".
15. My thanks to Steve Whittaker for the term "substitutability".

One of the problems of multimodal interaction is the variability in the way people communicate the same meaning. To take an example, a pointing gesture can be made with a head nod, a wave, a pointing finger, etc. Is the list infinite for any given function? No, clearly, if it was, people couldn't understand each other. The assumption here is that the morphology—or certain features of the morphology—of a multimodal action is mapped to its function by social convention. We can call this the *morphological-functional* link. Thus, for any given society, there are approved ways of communicating certain information. We can choose any of the above ways for pointing out an object in our surrounding. This phenomenon is referred to here as morphemic substitutability. If we want to be as clear as possible when pointing out an object in the environment, the best way to this in English speaking countries is with an extended arm and extended index finger, with the index finger pointing approximately in the desired direction. Clearly, this is not only a matter of intelligent use of the human figure to convey information, but also a matter of establishing a morphological-functional link. The more sloppy we are in extending the arm and finger, the more noise we introduce into the communicative act. This, then, suggests a second property of morphological-functional mapping: A graded index of flexibility, where certain morphologies are more strictly mapped to function than others. Mapping from morphology to function is strictest for words and symbolic gestures, and most flexible for sentences, speech acts and iconic gestures.

The corollary to morphemic substitutability is functional substitutability, where identical looking (or sounding) acts can serve different functions. An example is scratching your face while listening versus scratching your face when talking about an itch you had yesterday. The functional extraction in these cases has then to proceed by other indicators than morphology, the primary ones being content, dialogue state and the attentional state of participants [Grosz & Sidner 1986].

5.6 *Multimodal Dialogue as Layered Feedback Loops*

The model put forth here of multimodal interaction can be characterized as a layered feedback-loop[16] model, and is intended to be *de*scriptive—

16. "Feedback" in this context refers to the reciprocal nature of any speaker-hearer relationship, where a participant's [P1] multimodal action [P1-1] is met by the other's [P2] re-action [P2-1]. This loop can be more than one level deep; a common format is the sequence [P1-1→P2-1→P1-2]. See e.g. Clark [1992].

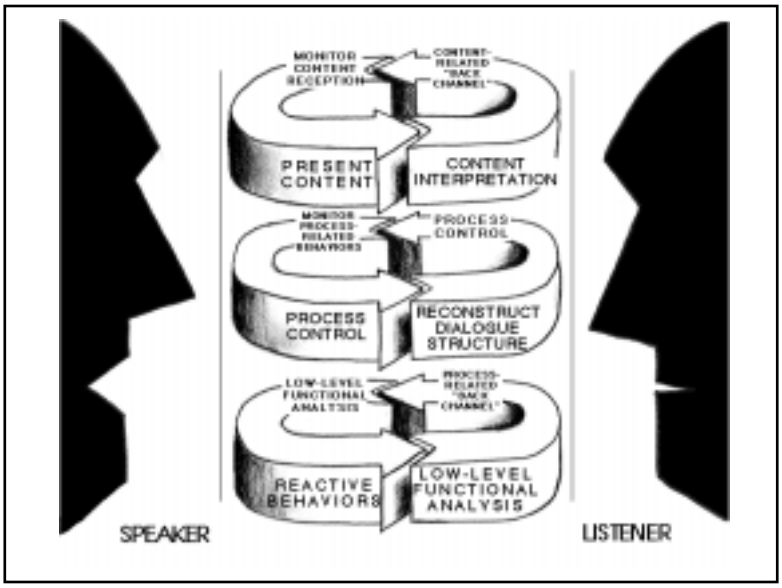


FIGURE 5-5. The proposed three-layered model of multimodal dialogue.

it is based on research from the psychological and linguistic literature—as well as *pres*criptive—it specifies how a conversant can be constructed (Figure 5-5). The three layers in the model are based on the time-scale of actions found in face-to-face dialogue (Figure 5-3). At each level various sensory and action processes are running, thus belonging to the category of functional hierarchy in Coordination Theory [Malone & Crowston 1991, Crowston et al 1988, Malone et al. 1988]. The set of sensory and action processes at work in each layer at any point in time is mostly determined by the role of the participant at that point in time: speaker or listener.

The lowest level is concerned with behaviors that generally have recognize-act cycles shorter than 1 second. This is the Reactive Layer. The middle layer concerns behaviors that are usually slower than 1 second. This is the Process Control Layer. Together these two layers define the mechanisms of dialogue management, or psychosocial dialogue skills. Direct references to the process of dialogue—e.g. utterances like "I'm trying to remember..." and "Let's see..."—belong in the Process Control layer and are generated in response to the *status of processes* in the other layers. Highly reactive actions, like looking away when you believe it's your turn to speak [Goodwin 1981] or gazing at objects mentioned to you by the speaker [Kahneman 1973], belong in the lowest layer. The third part of this model is the Content layer, where the content or "topic" of the conversation is processed. This layer deserves its own discussion,

and will be dealt with more in Chapter 7. and Chapter 8. The layers will all be more closely examined in these sections as well.

5.7 Summary

In this chapter we laid the foundation of a computational framework for face-to-face interaction. We identified the issues of real-time interaction that have to be solved for a satisfactory computer model as being [1] incremental interpretation, [2] multiple data types, [3] seamlessness, [4] temporal constraints, and [5] multi-layered input analysis and response generation. A proposal was made for a well-defined distinction between processes responsible for the behaviors of listeners and speakers. It was maintained that analysis of the function of multimodal acts has to happen before content can be successfully extracted from any such act. we presented the concepts of morphemic and functional substitutability—based on the observation that different looking acts can serve the same dialogical function, and that identical looking acts can serve different functions. The morphological-functional link is the proposal that morphology of an act is mapped to function by *social convention*.

Lastly, a proposal was also made for multimodal dialogue as semi-independent layered feedback loops, with each layer being responsible for separate parts of sensation and perception of an agent.

We will now see where this groundwork leads to: The next chapter will look at J. Jr.—a pilot study in face-to-face reactivity, and then, in Chapter 7., present †mir, a model for the generation of real-time multimodal dialogue behavior.

J.Jr.: A Study in Reactivity

6.

The J.Jr. system [Thórisson 1992] was a pilot system designed to explore the idea of reactive multimodal behavior in an interface agent. This system served as a precursor to the development of Ymir and highlights important problems in multimodal dialogue, which will be addressed at the end of the chapter.

6.1 System Description

In the J.Jr. system, dialogue control in is based on an FSM (finite state machine), augmented with a global clock. It uses data from three input modes: the user's hand gestures, gaze and intonation. Data about gaze and gestures is provided by a human observer in a Wizard-of-Oz manner (a person monitors the user's actions and keys them in according to a pre-determined scheme); data about intonation in the user's speech is obtained with automatic frequency analysis (Figure 6-1). This information is in turn used to control the gaze of J. Jr.'s on-screen face (Figure 6-2), its back-channel paraverbals, and turn-taking behavior, which consists of asking questions at appropriate points in the dialogue.[1]

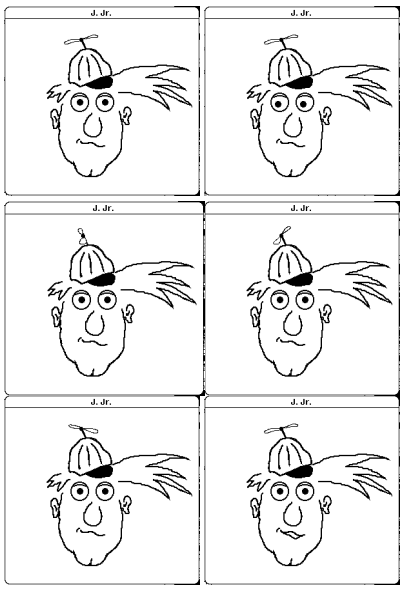


FIGURE 6-2. J.Jr.'s face is capable of looking around, blinking, rotating the hat propeller and opening and closing the mouth inrough synchronization with synthesized speech.

6.1.1 Input: Gestures, Gaze & Intonation

In the J.Jr. system gestures and gaze are quantified into Boolean variables; if the line of gaze intersects the on-screen agent face, the variable GAZE-ON? is set to TRUE , else it is FALSE. If the user moves his or her

1. Since asking questions and saying "m-hm, a-ha" are the exact qualifications for hosting a talk-show, J. Jr. is named after a well known American talk-show host. Like any respectable host, J. Jr. asks only questions that are very general and have no relation to what the user says.

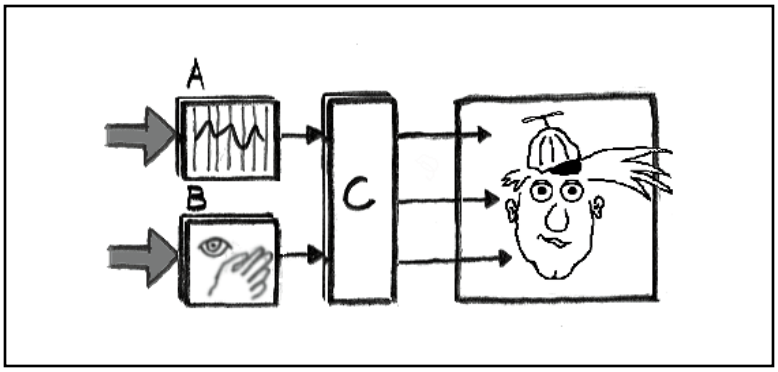


FIGURE 6-1. System structure. of J.Jr. The user's speech is automatically processed for intonational constituents and pauses (A). Information about gaze and gestures are monitored and input through a keypad by a human observer (B). The dialogue system (C) controls the cartoon character's speech, gaze and hat propeller

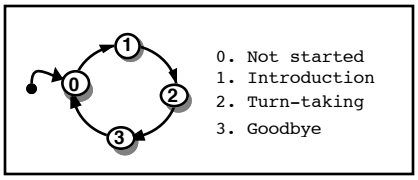


FIGURE 6-3. State diagram showing the control structure of the social encounter in J. Jr. Each state has a specific set of actions that the agent is capable of performing, as well as conditions (see text) for jumping to the next possible state.

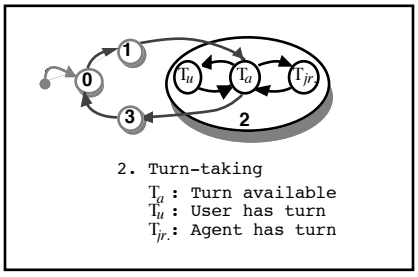


FIGURE 6-4. State diagram showing the three dialogue states, Tu, Ta, and Tjr, embedded within "encounter" state 2. In the original implementation the agent always asks the user a question in state Tjr.

hands in a way that obviously relates to the dialogue (i.e. excluding "self adjusters"—fixing of the hair, scratching, etc. [Rimé & Schiaratura 1991]), the variable GESTURES-ON? gets a TRUE value, else it is FALSE. This relatively sophisticated analysis of gesture is possible by using a human observer to code the user's behavior in real-time.

Pierrehumbert & Hirschberg's [1990] work strongly indicates that intonational features are important indicators about the intentional and structural features of discourse. Utterances contain intonational phrases made up of combinations of high and low pitches. Phrases can be divided up into sub-phrases, or intermediate phrases, which contain relatively small variations in pitch. The intonational phrase as a whole ends with either an increased high or low. In the J.Jr. system a simple filter is used to detect whether the speaker's pitch is rising or falling. Other speech variables used were SPEECH-ON? which is given a TRUE if the user is speaking, otherwise FALSE; and Silence, which contains the time in milliseconds since the user spoke. A third variable, PITCH-DOWN?, is set to TRUE if the intonation is falling, otherwise, if the intonation is rising or stays constant, it takes on a FALSE value.

The variable SILENCE contains the time in milliseconds since the user spoke. This turns out to be a very important element to time the actions of the agent.

6.1.2 Output: Speech, Turn Taking, Back Channel, Gaze

The agent's gaze and back-channel behavior is controlled with two variables: BACK-CHANNEL-ALLOW? and LOOK-AT-USER. The variable BACK-CHANNEL-ALLOW? is set to TRUE only when the user has turn and

is used to control when the agent gives back-channel feedback [Yngve 1970]. It is also used to prevent multiple paraverbals in a row, by setting it to FALSE immediately after a paraverbal has been given and waiting for the user to continue before resetting it to TRUE .

As discussed before ("Gaze" on page 44), results of research on gaze behavior in multi-modal interaction [Goodwin 1981] shows that the eyes play an important role in turn-taking; a speaker looks away at the beginning of his or her utterance, but as the utterance approaches termination gazes back to the recipient. The variable LOOK-AT-USER controls the gaze of the agent and is set to TRUE at appropriate points in the dialogue. (Looking at the user is accomplished by having the face look straight out of the screen.) If this variable is FALSE, the agent looks around at random.

6.1.3 Dialogue States

The dialogue control mechanism is a finite state machine augmented with a global clock. There are four "general" states for the dialogue encounter, and three for the turn-taking or dialogue itself (Figure 6-3 & Figure 6-4; the encounter states are numbered from 0 to 3).

Encounter state 2 is divided into three sub-states, or turn taking states, shown in Figure 6-4. These are marked Tu, Ta, and Tjr, for "user has turn," "turn available" and "agent has turn," respectively. Transitions between the states requires certain conditions to be true, determined by the values of the input variables.

6.1.4 State Transition Rules

In the following discussion a state change is denoted `Change-State` [a → b] or simply [a → b], where a is the prior state and b is the new state. The interesting states to look are the turn-taking states and how the agent achieves back channel feedback (state 2 in Figure 6-4). The initial sub-state is Ta. To make the transition [Ta → Tu], the simple condition **R1** (Figure 6-5).

The constant DIALOG-UNITS is set to 100 ms. This is the smallest unit of time measurement in the system; all other thresholds are multiples of this value. To go back to Ta we look for the conditions shown in **R2**, where (* 5 DIALOG-UNITS) is a multiplication of the constant Dialog-Units by five. For the agent to take the turn ([Ta → Tjr]) we wait for situation **R3** to arise.

The agent will look away and rotate the hat propeller as a clue to indicate that he is taking the turn. Since the agent cannot use body language to indicate dialogue states, these turn out to be fairly useful cues for the

R1: *User Takes Turn*

```
IF  (OR
        speech-on?
        gestures-on?)
THEN
    (Change-State [Ta → Tu]))
```

R2: *User Gives Turn*

```
IF  (OR
        (AND
          look-on?
          pitch-down?
          (not speech-on?)
          (not gestures-on?))
        (> Silence
          (* 5 Dialog-Units)))
THEN
    (Change-State [Tu → Ta])
    (Look-at-User ← TRUE)
```

R3: *Agent Takes Turn*

```
IF  (OR
        (AND
            (> Silence
                (* 2 Dialog-Units))
            look-on?))
        (> Silence
            (* 6 Dialog-Units))
THEN
    (Change-State [Ta → Tjr])
    (Look-at-User ← FALSE)
    (Turn-Propeller)
    (Ask-Question [Next-Q])
    (Change-State [Tjr → Ta])
```

R4: *Agent Gives Back Channel*

```
IF  (AND
        allow-back-channel?
        (not gestures-on?)
        (> Silence
            (* 1.1 Dialog-Units))
THEN
    (Give-Back-Channel)

(allow-back-channel? ← FALSE)
```

FIGURE 6-5. Pseudo code control algorithms for J.Jr.'s turn taking and back channel feedback behaviors.

user. The dialogue goes back to state Ta, [Tjr → Ta], immediately after the agent has finished the question. The function Ask-Question takes an argument, NEXT-Q, which contains the question to be vocalized by the speech synthesizer. This is read from a canned script of questions.

6.1.5 Back Channel Feedback

The back-channel mechanism is the only behavior to make use of the smallest unit of time measurement in the system, DIALOG-UNITS, which is set to 100 msec. In state Tu the rule **R4** (Figure 6-5) will produce back-channel feedback from the agent: The variable ALLOW-BACK-CHANNEL? is set to TRUE when entering state Tu. It is set to FALSE immediately after the back-channel feedback has been given, and back to when the user has started speaking again if the state is still Tu. The multiplier for DIALOG-UNITS in this case will undoubtedly vary depending on the "pace" of the dialogue, but judging from research on humans (see "Back-Channel Feedback" on page 40), is unlikely to need to be less than 1.0.

6.2 Discussion

First-time users often get the impression that the system makes use of powerful automatic speech recognition and language understanding to produce the observed behavior. This speaks for the relative quality of the turn-taking behavior and back channel, giving an informal "context-independent Turing test" for the dialogue behavior of the agent. A real interaction scenario with J. Jr. is described in Figure 6-6. While this system shows that accurate timing, intonation and crude gesture/gaze analysis can provide a sufficient mixture to take turns correctly, it also points to the problems of creating extensive systems that integrate reactive abilities with higher-level competence.

6.3 The Problem with J.Jr.

The system (and the illusion of semi-intelligence) breaks down when users start to speak nonsense to it—usually a somewhat disappointing moment for users, but not at all unexpected to the designer. I refer to the problems typified in this system as [1] the sensing problem, [2] the lack of behaviors problem, [3] the reactive-reflective integration problem, and [4] the expansion problem.

6.3.1 The Sensing Problem

Using a human observer to classify the kinds of gestures that the user does totally bypasses the problem of automatic gesture classification. Even though morphemic features of body motions are relatively gross, compared to intonation for example, they still may be difficult to analyze automatically because of the phenomenon of morphemic substitutability ("Morphological and Functional Substitutability" on page 76). One of the inherent problems lies in selecting the correct time-scale to analyze a person's behavior on. The importance of determining simple features like whether the user is addressing the computer agent or another person, whether a vocalization is a filler or an actual utterance that contains semantic information, cannot be stressed enough. These are the features that make system behavior robust.

6.3.2 The Lack of Behaviors Problem

A human conversant has a wealth of behaviors to choose from. On any occasion, these are chosen based on various features of the dialogue, and they are chosen in real-time. J.Jr. provides only a simple mapping between a state and its behaviors, but more importantly has no way to select or compose alternative multimodal acts if it did (this should perhaps be called the arbitration problem).

6.3.3 The Reactive-Reflective Integration Problem

How would we integrate natural language understanding into the J.Jr. system? If we want to integrate the content of utterances with intonation analysis and body language, we have to deal with complications like delayed production of results, backtracking time of occurrence of events and guaranteeing response (see "Computational Characteristics of Psychosocial Dialogue Skills" on page 65). How are we to integrate inofrmation content with real-time process control? When should a user utterance like "huh?" spin off a process that tries to re-plan a previous utterance? These are questions of internal and external process control, and they covary closely with the methods we employ for extracting information from the multimodal input stream. An outline of a solution to these problems will be provided in the next chapter.

6.3.4 The Expansion Problem

By using a finite state machine (FSM) as the basic mechanism of dialogue tracking, a serious limitation is set to the amount and ease of expansion. This means that building complex characters, with hundreds of behaviors (from blinking to planning many kinds of utterances), will be extremely difficult and time comsuming. FSMs are good for tracking states, and clearly we want to keep track of states in any dialogue

system. But for anything else in a dialogue system, perception, action control, multimodal integration, FSMs are not the right kind of mechanism, firstly because these processes are hard to describe in terms of states and their transitions, and secondly because the complexity of multimodal dialogue requires an incremental approach to behavior building, and FSMs don't lend themselves easily to such an approach. A possible solution to low-level (reactive) behavior would seem to be something like Brooks' [1986] subsumption architecture, but no clear mechanism exists in that approach to deal with higher-level analysis and output generation.

Kris: [00:000] Hello J. [00:550]
J.Jr.: [01:450] Hi, welcome, nice to see you. [04:100]
K:[09:650] Nice to see you too, you know, I've been ahh [09:650] ...
[10:350] working on you for a long time now and it seems like it's about time that you start behaving. [12:150]
J:[12:950] Yes. [13:400]
K:[14:250] And, ahh, [14:650]
J:[15:150] Yeah [15:600]
K:[15:600] and you seem to be doing that you know. [17:050]
K:[17:600] Oh, by the way, I like that propeller on your hat. It's kind of nice [20:450] ...
K:[21:000] it's kind of
K:[22:000] cute [34:060]
J:[22:050] Yeah, aha. [24:800]
J:[25:700] So what's new? Jnything interesting? [28:900]
K:[29:900] Well, you know, just, just the regular stuff [32:850] ...
K:[33:150] playing [33:450]
J:[33:200] Yes? [33:400]
K:[33:550] playing a little racquetball, keeping in shape. [35:800]
J:[36:950] Yeah. [39:150]
K:[39:700] Yepp. [40:000] ...
J:[41:450] Well, that's interesting. What else? [44:300]
K:[45:000] I don't know what to tell you, ahh [46:750] (Butterworth gesture) ...
K:[50:450] Well, you seem to be working properly now, that's, that's one good thing. [54:600]
J:[55:050] Yes. [55:750]
K:[56:750] And, ahh, I, I just hope you [58:800]
K:[59:650] gain a little bit more in your intelligence so that ... [1:02:250]
J:[1:02:250] Really? [1:02:750]
K:[1:03:400] Yeah, well, I'd...I'd, I'd like to see you [1:05:950] ...
J:[1:07:050] Aha. [1:07:650]
K:[1:07:450] have a little bit more brains, really [1:09:950]
J:[1:10:100] Yeah? [1:10:350]
K:[1:11:250] and, ahh, [1:11:700]
K:[1:12:400] but otherwise you're fine, [1:13:500]
K:[1:14:02] you know [1:13:950]
J:[1:14:150] Yes. [1:14:350]
K:[1:14:550] I can assure you. [1:15:150]
J:[1:16:350] Yeah. [1:16:600]
J:[1:18:000] Well, thank you for sharing that. [1:19:650]
J:[1:21:250] It was nice talking to you. [1:23:150]
J:[1:23:800] Good-bye. [1:24:400]
K:[1:25:250] Good-bye. [1:25:500]

FIGURE 6-6. Sample dialogue between the author and the J.Jr. system. Notice that while the content of all utterances is pre-determined, their timing is not. Paraverbals are randomly selected every time.

Speaker:[beginning, sec:ms] Utterance [ending, sec:ms].
Three dots (...) mark a pause longer than half a second; commas are pauses that are less than that. The agent's turn taking (and utterance of canned questions) are marked in bold.

Ymir: A Generative Model of Psychosocial Dialogue Skills

7.

In the past chapters we have looked at the complex issues involved in human face-to-face interaction. Some of those have been addressed individually in the literature, while some have not. The biggest piece missing though is a general way to put these items together to create a full model of face-to-face communication. The argument made here is the following: We need to look at the full loop of perception-action of an agent to come up with a correct model, because actions in dialogue are a mixture of closed-loop (guided with perceptual feedback) and open-loop (ballistic), and dialogue is interactive, with real-time planning happening on many levels. To do this we need a foundation where components that have already been developed can be accommodated, and new developments in the theory of multimodal dialogue can be "plugged in", tested, redesigned and retested.

Why "Ymir"?
Nordic religion, as preserved in Icelandic Sagas [Sturluson 1300~1325], tells about Ymir—a giant who lived in times before the heaven and earth. Ymir was killed by the Nordic gods Ó›inn, Vili and Vé, who turned Ymir's "blood into the seas, his bones into the mountains, his teeth and broken bones into rocks and gravel, his head into the heaven and his flesh into the earth." The earth then became a source of many new imaginative humanoid life-forms.

Ymir is pronounced *e*-mir, with the accent on the first syllable.

In this chapter I propose a new generative model of human psychosocial dialogue skill. Instead of dealing with a single issue, or a few small elements of face-to-face, multimodal dialogue, this model is intended to be a bridge, addressing all issues necessary to fill the gaps which in the past have prevented us from creating artificial characters that can engage in such dialogue.

The model is Ymir. Ymir does essentially what Fehling et al. [1988] call resource-bounded problem solving. The problem is dialogue; the resources are time, information and computational power. On the practical side, the general thought is that Ymir be used for creating softbots (and robots) whose purpose in life is to receive commands from humans, ask questions when appropriate, but otherwise do the job as best their knowledge allows them to. In the following discussion we can therefore envision building up to a humanoid robot who receives commands and turns them into executable actions in its domain of expertise. On the theoretical side, Ymir could be used to test theories about human discourse, because it provides the possibility to turn cer-

tain dialogue actions on and off at will—something that was impossible to do before, even with a skilled actor.

The approach taken to dialogue expertise can be likened to that taken to an expert system: we want a system that is expert at multimodal, face-to-face communication. The justification comes from the fact that unlike expert systems that tackle a niche area for limited purposes, multimodal dialogue is a general communication method used by all, and therefore we need only build this system once.

7.1 Overview of Architectural Characteristics

Following the model of face-to-face dialogue introduced in Chapter 5. (“Multimodal Dialogue as Layered Feedback Loops” on page 77), Ymir is a layered system. It employs one or more topic knowledge bases, and it uses a special action scheduler module for composing and scheduling motor actions. Motor actions are expected to be carried out by an animation system that either has addressable absolute positions for each “muscle” (analogue—e.g. a servo system, or digital—e.g. computer graphics or stepper motors), that lies below the system itself.

Sensory input is expected to be multimodal, and although the system *works* with a single mode input, no advantages would be taken of multimodal synergistic effects in that case. Ymir can accommodate any number of sub-modules, running in parallel or serial, that work in concert to interpret and respond to the dialogue as it unfolds. By being modular, Ymir offers researchers opportunities to experiment with various computational schemes for handling specific sub-tasks of multimodal interaction, such as natural language parsing, natural language generation and arbitration of multimodal motor responses. At the highest level, Ymir makes no specifications about the content of particular agent behaviors or interpretive processes and is therefore culture-independent.[1]

Ymir addresses all the features of dialogue presented in Chapter 5. (page 65). A summary of these manifests itself as the following list of requirements, all of which Ymir fulfills:

1. Co-existence of *reactive* and *reflective* behaviors,
2. *incremental interpretation* co-exists with *real-time response* generation, providing *seamlessness*, and
3. it handles *multiple data types* (spatial, boolean, symbolic, analogic).

1. Just like a telephone asks no questions about the language spoken on it, Ymir is not limited to the conversational rules of any single culture.

Features of three AI approaches have been adopted in Ymir: *Blackboard systems* [Adler 1992, Nii 1989, Engelmore & Morgan 1988, Selfridge 1959], *Schema Theory* [Arbib 1992] and *behavior-based* systems [Maes 1990a, 1989]. In the broadest sense, Ymir uses multiple knowledge sources that cooperate to provide a solution to a problem—in this case to interpret user actions and generate appropriate responses. Like Schema Theory, Ymir is highly distributed and contributes therefore to research in distributed artificial intelligence (DAI) [cf. Bond & Gasser 1988, Huberman 1988, Huhns 1987]. In contrast to the many approaches proposed various problem domains in the AI literature, the main novel and distinguishing features of Ymir are:

1. A distributed, modular approach to perception, decision and action.
2. A layered combination of reactive and reflective behaviors.
3. Dialogue-related interpretation is separated from topic interpretation.
4. Dialogue management is viewed as having complete process control (*when* something happens as opposed to *what* happens) of overt and covert actions.
5. Motor actions are split into two phases; a decision (or intentional) phase and a composition/execution phase.
6. Intentions to act vary in their specificity: the more specific an intention is (e.g. blinking) the fewer morphologies (ways to do it) exist; the less specific it is (e.g. looking confused) the more options there are in the way it will eventually be realized.
7. The final morphology of an intention is chosen at run-time.

Following Nii [1989] we can describe a computational system at any of three levels: The *model* is the least specific, showing the ideology behind the approach, the *framework* is more specific, detailing the pieces of the system and their interconnections, and the *specification* being the most detailed one, showing how to implement the particular system. In this chapter we will focus on the model and framework perspectives. We will turn to a more in-depth look at the implementation in the next chapter (page 111). Now let's get an overview of Ymir's main elements.

7.2 The 6 Main Elements of Ymir

The six main types of elements in Ymir are:

1. A set of semi-independent processing layers, Γ.
2. A set of blackboards, Φ.
3. A set of perceptual modules, ρ.
4. A set of decision modules, Π.

5. A set of behaviors, β, and behavior morphologies, βm (specific motor programs).
6. A set of knowledge bases, κ.

Starting with the layers, we will now take a closer look at these six elements. Then we will go into the Blackboards, followed by a discussion on virtual sensors, multimodal descriptors and decision modules (starting on page 97), and the behaviors and behavior morphologies.

7.2.1 Layers

There are four layers in Ymir:

1. *Reactive Layer (RL).*
2. *Process Control Layer (PCL).*
3. *Content Layer (CL).*
4. *Action Scheduler (AS).*

Each of these layers contains particular element types:

$$\Gamma_{(RL)} = \{\rho, \Pi\}$$

$$\Gamma_{(PCL)} = \{\rho, \Pi\}$$

$$\Gamma_{(AS)} = \{\beta, \beta m\}$$

$$\Gamma_{(CL)} = \{\kappa\}$$

FIGURE 7-1. The Reactive Layer contains mainly two types of processes, {1} perceptual (sphere and prism), and {2} decision modules (cube).

Each layer contains processes with similar time-specificity and functionality. The PCL is the main control element, with partial control over the other three layers. Processes in the RL can exert limited process control. Processes in the he Reactive and Content Layers perform functional[2] and content analysis of input. The AS produces specific motor morphologies, while knowledge bases in the CL interpret input about a specific domain and produces actions that are applicable in response to the content of that input. Different kinds of delays and delay constants exist at each of its four levels. Time stamping and the use of synchronized clocks is a general way to deal with temporal constraints: Using time stamping, delays are logged and treated like any other data in the system. We will now take a closer look at each of the layers.

2. See "Functional Analysis: A Precursor to Content Interpretation and (sometimes) Feedback Generation" on page 71.

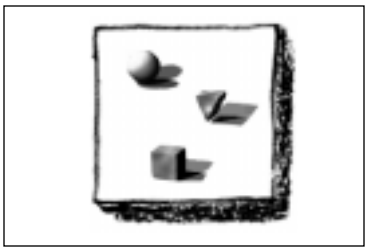


FIGURE 7-3. The Process Control Layer contains two kinds of processing modules (in multiples), {1} perceptual modules (sphere and prism) and {2} decision modules (cube).

Reactive Layer (RL)

The role of processes in the Reactive Layer is to compute timely information on user actions and make these available to decision modules in the same layer. Sensory data from each mode is processed with modules called *virtual sensors*, and mode-specific data is combined in processes called *multimodal descriptors*. Reactive Decision Modules use this data to issue actions with a relatively high speed/accuracy trade-off. I.e. as long as the results are quick, and correct more than 50% of the time—above chance performance[3]—it doesn't matter that they are less than 100% accurate, because *a* response is more important than it being correct.

Computation at this lowest level is assumed to be "immediate", i.e. without delay. This simplification can be made by using computing mechanisms that are significantly faster than the fastest response needed in human interaction, ideally a fraction of 100 msec. In spite of actions being immediate in this layer, every event at the RL level is nonetheless time stamped for the benefit of computations in other layers.

ACTION: smile
CREATOR: PCL
EL: 600 ms
STAMP: 35243

FIGURE 7-2. Example of a behavior request message sent from the Process Control Layer to the Action Scheduler.

Process Control Layer (PCL)

The role of processes PCL is to control global aspects of dialogue: to turn the correct kinds of internal processes on and off, recognize the global context of dialogue and manage communicative behavior of the agent. It deals with issues such as *when* a question should be answered; what to do when information is missing, when to greet; etc. It also controls internal actions related to the dialogue such as starting to listen, making predictions about the knowledge needed in a particular interaction; making predictions about what to expect next; managing multi-turn information exchange, etc. Modules in the PCL can control (turn on and off; change thresholds in) the processes in the Reactive Layer, as well as its own. This feature is essential since many of the lower level processes don't have global enough information to decide when they should be active and when not.

3. This is not to say, of course, that striving for as high a recognition accuracy as possible should not be tried.

In addition to the perceptual and decision modules, the PCL also has a few processes specifically related to "book keeping". These will be discussed in Chapter 8.

Action Scheduler (AS)

The AS can be thought of as a kind of "cerebellum". Its role is to receive *behavior reques*ts (βr) from the Reactive, Process Control and Content Layers, prioritize these and choose a specific morphology for them (see "Morphological and Functional Substitutability" on page 76). There are many ways to realize behavior requests, which can be thought of as the "intention" to perform a specific act: one example is given in Figure 7-2. A behavior request is thus a decision to do a specific action, independent of its form. Behavior requests can be specified at various levels of detail. Specific behavior morphologies (βm) are chosen from a library of alternatives[4]. Increasing generality in the specification of a behavior, e.g. "pull left corner of mouth up halfway" vs. "smile", means more options in its morphology (see page 101).

To chose between βm options, the AS uses a trade-off algorithm. To take an example of how the algorithm works, if the AS receives a request for the behavior **acknowledge**, it can use dialogue state and the amount of load on the various degrees of freedom of the agent's motor system to choose a way to express this. The usual method could be to say "yes", but if the user is speaking, perhaps a nod would be more appropriate; however, if the agent's head is moving, it might choose to give verbal back channel feedback anyway. There are many ways to realize such a scheduling algorithm; we will see one particular method in the next chapter.

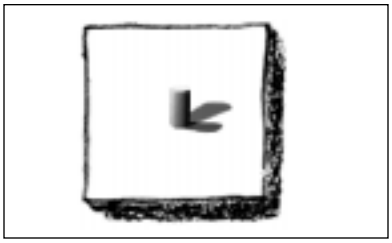


FIGURE 7-4. The Action Scheduler Layer contains {1} a lexicon of behaviors and behavior morphologies (cylinder), as well as {2} scheduling mechanisms for requests to perform these (not shown).

4. Complex behaviors that span long stretches of time need to be interruptible, as well as computed incrementally to allow relevant, unexpected events to be taken into consideration.

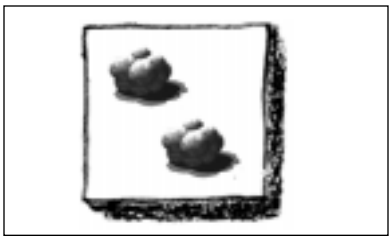


FIGURE 7-5. The Content Layer contains several knowledge bases (shown as cloud-like blobs).

Content Layer (CL)

The role of the CL is to host the processes that make sense of the *content* of the input and generate acceptable responses based on this. For example, given the multimodal input "Delete [gesture] those", the CL should be able to combine the verbal and gestural actions, and come up with a correct action in the topic domain (i.e. remove a set of objects).

The Content Layer contains one or more Topic Knowledge Bases (TKBs), which contain information about how to interpret a person's multimodal acts, how to generate responses to those acts, and how to communicate its status to other parts of the system, particularly the Process Control Layer. The CL also contains a Dialogue Knowledge Base (DKB), which stores meta-knowledge about all other knowledge bases in the layer. A meta-knowledge DKB allows the system to select the most relevant TKB at any point in time.[5] To take an example, if the agent knows the two topics of music and computer graphics, and hears the utterance "Turn the blue box sideways" the DKB will recognize that this utterance probably refers to the computer graphics topic, and will notify the TKB containing the knowledge necessary to interpret computer graphics-related utterances, which will in turn interpret the input and generate some actions that will make the user's wish come true.[6]

Once a Topic Knowledge Base has generated usable output, it will notify the DKB, which has the necessary knowledge to know *when* to execute this action in the dialogue.

5. Alternatively, we can run a DKB in parallel with any one (or more) TKBs which is considered relevant at any point in time. Output from the KB that produces the best interpretation will be selected. This has two consequences: {1} The KBs have to give a measure of their success and{2} equally good interpretations from different KBs have to arbitrated. This can be done in many ways, including asking the speaker a question, and because the conflict happens at the meta-level, that question would be composed in the DKB.
6. If the DKB is uncertain which KB to pipe the input to, it might pipe it to more than one TKB and choose the outcome that is rated as a "good recognition" by the TKB.

In Ymir, the DKB is considered a central part of the system's psychosocial skills. In fact, any knowledge that has to do with dialogue, such as knowledge about participants, their body parts, instruments used in interaction (mouths, hands, eyes, etc.), greetings, good-byes, etc., rightfully belongs in the DKB and are considered a topic (albeit a meta-topic) in and of itself. The argument behind this view is the same as behind the push to embody the computer: knowledge about interaction is intrinsic to the interaction and necessary to conduct it correctly. While history about the task—be it excavation or moon landings—is stored in the relevant TKB, history about the interaction, and references to the interaction ("Go back to when I told you ..."), are stored and treated in the DKB. The big win in this modular approach is that it allows for the one-time creation of dialogue knowledge ("Look over here", "Listen to me"), with domain-dependent knowledge being "plugged in" by the agent's designer (and by the system at run-time) in a modular fashion.[7]

Summary of Layers

Placing these four parts of Ymir within Coordination Theory (see Figure 5-2 on page 69) [Malone & Crowston 1991, Crowston et al. 1988, Malone et al. 1988], the Reactive Layer and the Process Control Layer are product-oriented hierarchies: they contain a complete set of heterogeneous processes to produce a product—the product being external and internal actions. The Content Layer is also a product-oriented hierarchy: its job is to produce descriptions from the user's commands that can be executed in the agent's world. (For example, upon hearing the words "Delete the blue box", the system should be able to locate the I.D. of the blue box, find the correct command to remove items, and apply that command to the I.D. of the requested object.) The Action Scheduler, however, repetitively carries out the same kind of process, and is thus what Coordination Theory calls a functionally-oriented hierarchy.

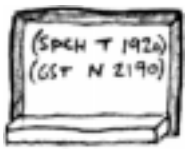


FIGURE 7-6. Blackboards contain information accessible to a certain set of modules.

7.2.2 Blackboards

How do all these processes talk to each other? Psychological research has shown that in perceptual processing, different information becomes available at different times: for example, low-frequency visual information and motion becomes available sooner than higher-frequency information [Card et al. 1983]. A person can select how long to wait before reacting to a particular stimulus, depending on the current trade-off between cost of delay and cost of errors. This points to a process where information that has already been computed is made available to the rest

7. I would like to thank Richard A. Bolt for pointing me to the issue of modularity; see also Walker & Whittaker [1990].

of the system. In A.I. a blackboard is a metaphor for a global information exchange [Selfridge 1959]. Any process with access to a blackboard can look for information relevant to its own processing. In Ymir we use the idea of more than one blackboards, each with limited access. There are three main blackboards. The first one is for information exchange primarily between the Reactive Layer and the Process Control Layer. This blackboard is called the *Functional Sketchboard* (FS). It stores intermediate and final results of low-level (high-speed) perceptual processes such as motion, whether the agent hears something or not, and first-pass multimodal descriptions.

The second is the *Content Blackboard* (CB), servicing communication between the PCL and the Content Layer. This blackboard is the key to the separation of process control and content analysis in Ymir. Here results are posted that are less time-critical than those on the Functional Sketchboard.

The third blackboard in Ymir is the *Motor Feedback Blackboard* (MFB), where the Action Scheduler posts the progress of behaviors currently morphing and executing. In the MFB the PCL and CL can read status of formerly initiated behaviors and replace those that are cancelled or have failed for some reason.

7.2.3 Perceptual Modules

This section explains input processes of an agent, and input data representation. There are currently two kinds of perceptual processes in Ymir, *virtual sensors* and *multimodal descriptors*. But before describing these, let's look at the philosophy behind the approach.

Background

The organization of low-level perceptual processes in Ymir follows the so called purposive, qualitative, or animate perception approaches [Aloimonos 1993] in that it is purpose-directed and ego-centric. It is based on the general idea that a situation is an important factor in selecting the perceptuo-motor skills of an animal, to maximize attentional and mental faculties for the particular tasks that situation calls for. This is closely related Agre & Chapman's [1990, 1987] *indexical-functional* representation in their *Pengi* system, where objects and dependencies are represented in terms of the effect they have on the agent's goals, in ego-centered terms, e.g. instead of representing a flying bee with the symbol `BEE-23` it uses labels like `the-bee-i-am-now-chasing`, defining the bee in direct relationship to the perceiver [Lyons & Hendriks 1992].

At higher levels the system's perception becomes more complex, slower and more general. Here we can expect the agent to be recognizing objects, faces, body parts, and relating them to its lexical and relational knowledge bases, to generate verbal output for instance. These kinds of processes are not specified in Ymir (this is quite a research topic), but general ideas about how these could be handled in a dialogue system will become apparent as we continue.

The perceptual abilities of an agent are assumed to be grounded in knowledge about the *interaction*, such as knowledge about participants, body parts, turn taking, etc., by *situational indexing*. Thus, an agent created in this architecture is not expected to be trying to avoid obstacles, prevent itself from getting killed, etc., while it engages in interaction with humans, and therefore does not require any non-communication based perception. For instance, upon hearing a voice in a given (perceived) location, an orienting response toward that voice could be triggered. The architecture could of course be expanded to include perceptions of other things than only those relating to communication. How broad the ego-centered approach can be made is an open question. Now, let's take a look at the two perceptual elements in Ymir, Virtual Sensors and Multimodal Descriptors.

Virtual Sensors

The simplest processes in Ymir's perceptual system are the virtual sensors, which process simple features of the dialogue and output Boolean values. These sensors are considered to lie at least one level above the energy transducer layer, such as a retina or cochlea, or, in the case of the Gandalf system (Chapter 9.), the space sensing cubes, eye tracker and microphone. A virtual sensor is usually associated with a single mode: an example would be a vocalization sensor that turns on or off depending on whether the user is making sounds with his or her throat.

The virtual sensors in the Ymir system fall into the following categories:

1. *Prosodic*
2. *Speech*
3. *Positional*
4. *Directional*

Prosodic sensors track the intonation of the speech, pauses and volume of vocalization; *speech* sensors are a general class of processes that look at the speech content. They could for example be sensitive to certain words that play a role in dialogue orchestration such as cue phrases; they would also track the global functional aspects of speech, such as determining whether an utterance is syntactically correct, whether it makes sense pragmatically, what the topic is, etc., as much as these can be gleaned from just looking at the speech (more extensive analysis of

these is done at higher levels, albeit at a slower pace). Examples of each of these classes will be given in the chapter on Gandalf. *Positional* sensors track the absolute position of objects (position in real-space) or the relative position of two or more objects, e.g. displacement of the eyebrows from a resting position, and *directional* sensors track the direction of objects (e.g. gaze, trunk or head). Two fundamentally different kinds of virtual sensors are postulated:

1. *Static sensors*, which report a current static state to be true or false, and
2. *dynamic sensors*, which track the change of a certain feature of a single mode over time and report on the conditions over a given duration of time.

For example, a static sensor could report whether a person was looking at the agent's face or not. A dynamic sensor could report whether the person glanced away quickly and then back. The theory behind this is that short patterns of activity may have significance for the interaction [cf. Argyle, Lefebvre & Cook 1974].

Multimodal Descriptors

Processing the output provided by the virtual sensors is a net of what I call *Multimodal Descriptors*. The descriptors aggregate information from the Virtual Sensors to compute intermediate, multimodal "functional sketches" of the user's behavior. An example is a descriptor that tries to determine whether the user is giving the turn. Another might combine information from a spatial sensor and a speech sensor to provide a {SOUND, LOCATION} pair that can be used by the agent for orienting itself toward a person. Two kinds of descriptors are proposed,

1. *static descriptors* and
2. *dynamic descriptors*.

As with Virtual Sensors, *static* descriptors simply respond to a static situation, whereas *dynamic* descriptors detect patterns over time intervals—as reported by the virtual sensors—such as a specific combination of arm and eye movements for a given interval. For example, bringing up your hand and uttering something ("ahhh") while the agent is talking may constitute a wish to interrupt. This could be detected by a static descriptor sensitive to the conditions of either hand in gesture space and vocalization present. A head nod and a particular vocalization ("aha") combines into a single "back channel feedback" report, detected with a single dynamic descriptor.

Summary of Virtual Sensors and Multimodal Descriptors

We can now summarize the perceptual system. The virtual sensors receive data from the sensing equipment and do initial computations to prepare it. Multimodal descriptors monitor the status of the virtual sensors through a blackboard and change states. In the implementation of Ymir, this is based on first-order logic combinations of these, as well as the states of other descriptors. Later versions of Ymir might use Fuzzy Logic [c.f. Kacprzyk 1992] for this purpose, or other methods, provided they are fast and flexible enough.

7.2.4 Decision Modules

Decision modules look at the state of the agent's knowledge, which includes a representation of the outside world as well as the state of its own processing, and make decisions about what to do from moment to moment. These decision can affect both the outward behavior of the agent or the internal processing inside the agent's "mind", and thus fall broadly into two categories:

1. *External Decision Modules*—those that initiate overt actions, and
2. *Internal Decision Modules*—those that only change the internal state.

Granularity of the modules varies according to their task. Each decision module contains knowledge about where to look for data (which blackboard), what to do with it and how to communicate its status to other modules by posting information to the blackboards.

Decision modules in the Reactive Layer search for specific conditions in the Reactive Layer's Functional Sketchboard; the Process Control Layer's decision modules can look for conditions in both the Functional Sketchboard and the Content Blackboard (see "Blackboards", above).

7.2.5 Representation of Behaviors

We are now ready to look in detail at the fourth and last layer in Ymir, the Action Scheduler.

Background

Research on errors in human and animal locomotion have supported a model in which distinct levels of representation are at work for any motor act [Rosenbaum et al. 1992]. For example, levels activated earlier provide information spanning longer stretches of time, e.g. the global act of moving your arm/hand/finger to enter the expression "27 + 9 + 3" into a calculator. Levels actuated later provide smaller and smaller constituents for that behavior, e.g. individual key presses. This model

would indicate that *information* needed to execute an act like that would also need to be represented at multiple levels: locating the calculator in real-space is more coarse than locating its individual buttons. Rosenbaum et al. [1992] have proposed what they call the Knowledge Model: Motor control is performed by autonomously functioning modules that compete for execution, and that these modules carry information about postures. This model, and similar ones [Rosenbaum et al. 1991, Albus et al. 1987] are aimed at explaining complex motions like those of the arm moving the hand to press a button in an elevator, all the way down to the feedback provided from the muscles.

Behaviors & Behavior Morphologies

The approach taken here is in some ways similar to Rosenbaum et al.'s [1992]. The idea of stored postures is used in the Action Scheduler, as is the idea of hierarchical storage of increasingly smaller units. However, choosing between alternative actions is done by a monolithic algorithm, not competing individual modules. In Ymir, action is split into two phases: an action request (or intentional, decision) phase and a composition/execution phase. As discussed in the section on the Reactive and Process Control layers, the first phase is based on a collection of decision modules that can request specific actions, specified at varying levels of detail. The second phase happens here, in the AS. This method for representing behavior leads to a database where functional and morphological definitions co-exist in the same space, with no distinct division lines between the two classes. An example of what such a database could look like is given in the chapter on Gandalf (Section 9.7.1 on page 153).

Behaviors are indexed at various levels of detail: `smile`, `pull-corners-of-mouth-up`, `move-motor-x-to-position-y`. Obviously, there must be hundreds of ways a person could smile, fewer ways in which one could pull the corners of mouth up, and perhaps only one way to move a muscle to a particular location. We can make a tree, where particular morphologies are given as the tree's leafs (Figure 7-7). As we travel up the tree, the flexibility for various implementations of a particular act, like smile, or show-taking-turn, increases. Of course, given more options, it takes longer to choose the best one. Decision modules in the Reactive Layer related to external behavior generally request highly specific actions; those in the Process Control Layer usually make a more general specification.

Action requests issued by the RL are generally specified at a lower level than those in the PCL, since these are under tighter time constraints (with the effect that the AS—see below—doesn't have to spend valuable time composing a set of motor commands for the action involved, but simply looks up the default motor schema and sends it to the anima-

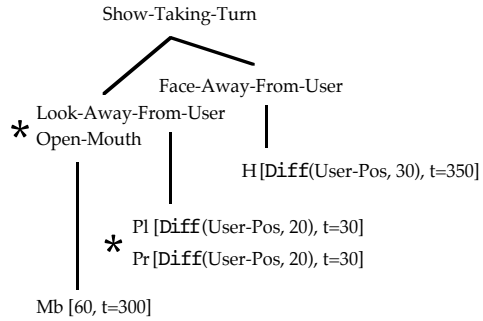


FIGURE 7-7. The hypothetical behavior *Show-Taking-Turn* has two possible instantiations, *Turn-Away-From-User* and the parallel pair {*Look-Away-From-User, Open-Mouth*}. Each of these point to low-level motor commands with degrees and time in milliseconds. The function Diff returns a setting that is guaranteed to not include the user's position in the agent's line of sight. Parallel actions are marked with a star.

H = agent's head, P = pupil (left and right), t = time in milliseconds, other numbers represent degrees (and relative position in the case of motor Mb), Mb = bottom mouth motor (see Figure 8-11 on page 123).

tion module). Examples of actions issued by the RL are `show-taking-turn` and `look-at-person`. Examples of the more extensive actions included in the PCL are `indicate-response-delay` and `express-confusion`. To determine which layer a potential action should belong to, one can use time-specificity (i.e. those that typically have expectied lifetimes under 1 second are likely to belong in the RL than the PCL; see "Temporal Constraints" on page 69), and/or the time the actions spans (RL if less than 1 second). Another criteria is the kind of perceptual data the behavior needs to be executed reliably; those requiring complex data are less likely to belong to the RL. Guidelines for determining which layer a particular behavior belongs to are likely to emerge out of further research on this topic.

Generating Manual Gesture

The dialogue management system, by itself, can support the generation of four classes of dialogue-related manual gesture [Rimé & Schiaratura 1991], independent of the topic knowledge base(s) used[8]. These are {1} *emblem gestures* related to the dialogue (e.g. holding up a hand to signal "Stop speaking!"), {2} *deictic gestures* (involving objects in the dialogue knowledge base), {3} *beats* and {4} *butterworths* (see Figure 3-2

8. Since iconic, pantomimic and deictic gestures related to the topic of discussion cannot be generated without reference to knowledge of the topic, and the knowledge residing in the dialogue system contains no topic knowledge, these would be generated in the corresponding knowledge base.

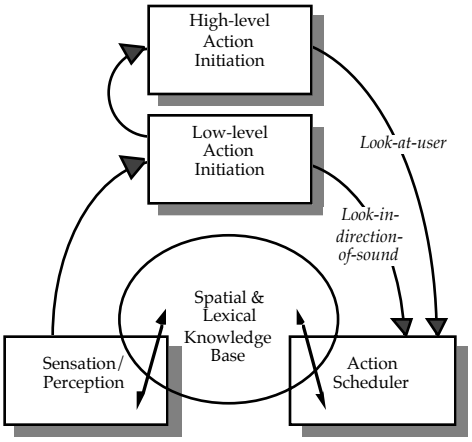


FIGURE 7-8. The Action Scheduler has access to a spatial knowledge base that is kept updated by the sensory and perceptual mechanisms. Examples of messages sent to the AS from the Reactive and Process Control layers are shown in italic letters.

on page 44). These are all requested from the RL and PCL by calling the appropriate type of gesture with the optional parameters (such as a 3-D vector for deictic gestures) and treated in the same way as other actions in the Action Scheduler.

Any gesture related to the topic should be generated in the corresponding Topic Knowledge Base.

Spatio-Motor Skills

To allow an agent to move in relation to surrounding objects such as a person or a task area, the AS needs access to a spatial knowledge base. Examples of such actions would be `Look-at-User` and `Turn-to-Area-[X]`[9]. I propose that this should be done with access to a common spatial knowledge base that is fed with information from the sensors[10] (Figure 7-8).

9. This behavior, unlike the other examples, contains a variable. There is nothing in Ymir that excludes such modules—in fact, for complex behaviors they will prove essential.

10. A simplified version of this approach has been implemented in Gandalf (Chapter 9., page 129).

When do we go Ballistic?

When, in the process from intention to execution, does an action, or part of an action, become impossible to cancel? This question about where to go ballistic is an important one. As we established in Chapter 5. the incrementality and reactive nature of dialogue allows participants to interrupt each other at a moment's notice. In Ymir, once the AS has sent the commands out, the agent's behavior is ballistic, i.e. there is no way to cancel their effect after this point. This last part of the path should therefore be kept very short, typically less than a second. For actions longer than a second, one would expect them to be composed—or at least *executed* [Kosslyn & Koenig 1992]—incrementally so that they can be cancelled at any time.

Creating Behavior Classes with Cascaded Decision Modules

The idea behind cascaded decision modules is this: Suppose you're walking along a narrow trail in the woods and you've decided to put your food down somewhere, when suddenly you realize it looks like a puddle and you don't want to get wet. So you cancel your original motion goal and replace it with a new one. The new goal results in a motion that moves your foot to a different location along a different path. The new goal in this example is one decision module of a group of cascaded decision modules, all aimed at placing your foot in various ways on the ground. By cascading a number of decision modules, corresponding to a number of behavior morphologies, each triggered in the case of another's cancellation, inappropriateness or failure, whole classes of behaviors can be built up.

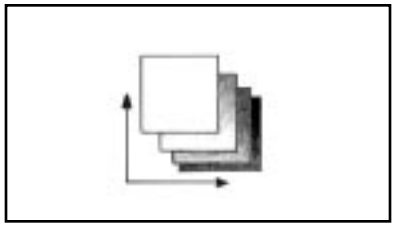


FIGURE 7-9. Cascading decision modules with slightly different triggering timing (abscissa) and conditions (mantissa) allows us to cover classes of behaviors.

For example, if the agent performs the action `Show-Taking-Turn` but the user continues to speak, the `Show-Taking-Turn` behavior must have failed somehow, either in execution or in indicating to the user what it was intended to show—or perhaps the user just decided to ignore it. The outcome in any case is the same: the user simply continues to speak, as measured by the virtual sensors. But how should the agent respond? It has already decided to take the turn, and may have stopped listening, yet has been unsuccessful in showing this to be the case. And its `Show-Taking-Turn` module has already fired (and won't fire again unless it is reset). The solution to this apparent deadlock lies in designing a second module, called `Show-Take-Turn-2`, that activates if the two states of user-speaking and `Agent-Has-Turn` occur simultaneously for a longer-than-normal period, e.g. 500 ms. The outward behavior resulting from this second decision might be more exaggerated than that of the first, including perhaps a manual gesture to try to show the user that the agent really wants the turn.

Another example of using cascaded behaviors is the situation where you know have been asked a question, and you have already shown that you're taking the turn, but you haven't come up with the answer yet. To deal with this delay, people engage in various verbal and non-verbal activities; they look up, say "ahhh" or fill in with more extensive elaborations like "I know this...hang on a second." With cascaded behaviors, delays as these can be taken care of automatically depending on the duration from the time you took the turn. The decision module/behavior `Produce-Filler` could produce an initial "ahh", `Produce-Filler-2` might be designed to take care of an additional delay. Yet other decision modules could take care of pauses that hadn't been filled but should have been—we can imagine a collection of 10-20 such modules, each with slightly differing conditions for triggering. This puts the creation of the behaviors in the hand of the agent designer, but the selection of each option in the hands of the run-time system.

Using cascaded decision modules (Figure 7-9), whole classes of condition-action situations can be addressed. How this is done for a particular agent—and this applies to all modules in the system— is largely an empirical task that depends on a number of factors including the potential users, their diversity, and cultural background.

7.2.6 Knowledge Bases: Content Interpretation & Response Generation

A knowledge base provides a system's ability to "understand" language, gesture, facial expressions and gaze. The topic of an interaction could revolve around car mechanics, architecture, plumbing, or the solar system. The knowledge base will provide the ability to key together actions such as pointing at a wall and the utterance "remove that wall" so that the multimodal act can be understood and an appropriate reaction to the command generated.

Ymir contributes to our understanding of interpretation in that it provides a framework for better specifying the *limitations* that interpretive processes have to work under—regardless of how they are exactly implemented. The major requirements the TKBs used in a real-time dialogue system have to fulfill:

1. Processes have to be *incremental*.
2. Processes have to be *reportable*.

Incrementality means that interpretation happens at a fine enough granularity to allow meaningful communication between knowledge bases and the PCL to happen during a user's utterance. *Reportability* refers to the system's ability to report on its own status and progress. These principles are the basis for allowing a modular approach to knowledge

representation. Other benefits are also expected. If these principles are adhered to, the system's ongoing topic interpretation of a user's multimodal act can be constrained by the information posted to the functional sketchboard. For example, if the Sketchboard has an indication that a communicative gesture was made during the user's turn, gestural analysis of the segment can be started up in the TKB even before the user has stopped speaking. If the interpretation process encounters problems, an ordered list of progressively less likely functional roles for the multimodal act in question can be used to provide a next viable candidate. The Process Control Layer will take care of any delays that such a disruption may introduce into the interaction.

For each TKB, and the DKB, non-real time blackboards could be used for more exact functional analysis[11], using a high accuracy/speed tradeoff, but still working along the same lines as the sensor/descriptor system. An important requirement of this design is that the sketches provided by the fast analysis match more than 50% (= chance) of the analysis performed at the higher levels, because otherwise any real-time feedback based on these sketches is not correlated with the higher-level functioning of the system. To be sure, people often make mistakes in their functional analysis (e.g. mistaking a deictic nod for an acknowledgment) which they effectively mend in the process of the dialogue. But such mistakes are disruptive and hence should be avoided in a computer agent. We will come back to this issue in Chapter 9.

If achieving good correlation between low-level analysis and high-level interpretation proves to be a difficult task, there is at least one reason to be optimistic about this being a problem: If the user is following Grice's maxim [Grice 1989] about mutual cooperation in dialogue, over time, she is likely to modify her own behavior to make the two consistent in the agent. This modification would probably take only a few hours—certainly less than days—of interaction. So, if the agent consistently displays the same inconsistency between the real-time feedback and higher-level analysis—i.e. as long as there exists a morphology of user behavior that results in the same analysis on both levels—we may expect such problems to disappear over time.[12]

11. The exact implementation of the KBs is not specified in our architecture; this is one of the issues that should be left to later research.

12. In fact, data in the evaluation experiment of the Gandalf system (Chapter 10.) seemed to support this claim.

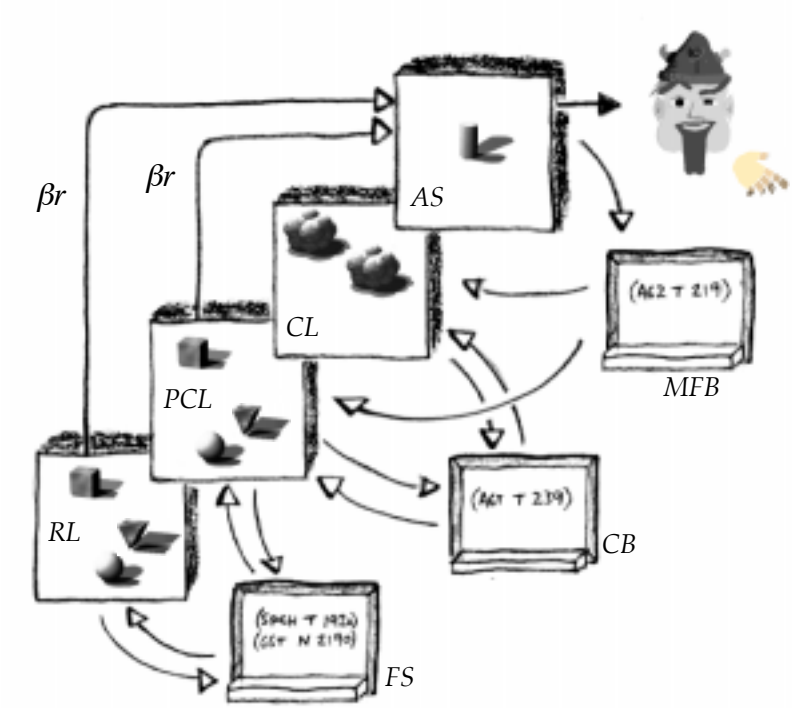


FIGURE 7-10. Overview of the Ymir architecture, showing the four layers, types of processes in each layer (square, sphere, prism, blobs, cylinder), communication links (hollow arrows) and motor control (black arrow), as well as the blackboards used for communication between the layers (βr = behavior requests, RL = Reactive Layer; PCL = Process Control Layer; CL = Content Layer; AS = Action Scheduler; FS = Functional Sketchboard; CB = Content Blackboard; MFB = Motor Feedback Blackboard). Not shown are the direct control links from the decision modules in the PCL to the Multimodal Descriptors in the RL. Multimodal information is assumed to be streaming in to each of the RL, PCL and CL as needed.

7.3 Ymir: Summary of all Elements

Figure 7-10 summarizes the layers in Ymir and provides an overview of their interconnections. Hollow arrowheads show the flow of information; the black arrowhead indicates absolute commands. Notice that flow between layers is not deterministic; each layer decides what to do with the data received from elsewhere according to time-constraints and the type of data. Layers further down the page are generally slower layers, with the exception of the Action Scheduler, which uses time as a variable to determine its processing. Not shown is the ability of State Decision Modules to turn the activity of perceptual modules in the PCL and RL on and off.

7.4 A Notation System for Face-to-Face Dialogue Events

To facilitate the discussion about the model presented above, and the way it treats dialogue, it would help to have some kind of simplified, clear way of presenting the complexities of face-to-face events.

We start by defining the following concepts:

- **A.** *Dialogue Condition* (C_d)
- **B.** *Action* (A)
- **C.** *Action Trigger* (A_t)
- **D.** *Expected Lifetime of an Action* (A_{el})
- **E.** *Action Execution Time* (A_{et})
- **F.** *Dialogue Participant* (P_d)

Any dialogue condition (C_d) could be a situation appropriate to respond to. It consists of a collection of necessary states of the dialogue participants' modes (hands, face, gaze, etc.), dialogue (e.g. who is speaking), mental state, etc. A C_d of significance to a particular dialogue participant, P_d, is indexed in this participant with an Action Trigger (A_t). A response A to a condition C_d gets initiated upon the A_t event, if the condition is detected.

$$\textbf{Detect}\,(C_d) \Leftrightarrow A_t \tag{7.1}$$

The action's execution time (A_{et}) determines how long after the A_t the action starts to take effect. The expected lifetime of an action (A_{el}) is used to determine the likelihood that an action is outdated when it comes time to execute it (move the "muscles").

$$\textbf{Execute}\,(A) \text{ IF } A_{et} < A_{el} \tag{7.2}$$

For example, a ball comes flying in your direction ($EVENT_1$, or E_1). You decide to [catch the ball]-(A'), but at a critical moment you [see that you won't be able to]-(E_2), and you decide to [give up]- (E_3) on the action, before it's been fully executed. In this example, $A_t = E_1$. Action A' gets initiated at **time**[E_1]. The A_{et} of A was too long, and the action's A_{el} allowed you to cancel the action before it was over. In this example, A_{el} is computed *during* execution of the action: you compare the progress of the action (distance between hand and ball) to its goal (ball in hand); before the goal is reached you decide, based on your prediction, to cancel it. For very fast, reactive responses, A_{el} has to be precomputed, because the perception necessary to assess the progress of such short actions would take too long. For an intention spanning a long period, A_{el} can be computed during execution. We will see an imple-

mentation of these mechanisms for reactive behaviors in the next chapter.

7.5 Summary

We presented Ymir—a generative model of psychosocial dialogue skill. The model supports the necessary framework for the creation of a computer controlled character capable of real-time orchestration of seamless, multimodal input and output. The model proposes the distinction between dialogue management and topic expertise; emphasis is on the former rather than the latter. As a result, the model underspecifies characteristics of topic knowledge but makes instead some specifications for the interaction protocol between the administrative tasks of multimodal dialogue on the one hand and topic knowledge on the other.

The model is multi-layered, with each layer providing a specific set of processes. These processes provide certain computational services, with the ability to communicate their results to other modules via blackboards. Actions in the system offer various degrees of "reactiveness", from very reflex-like to highly "intelligent". The morphology of actions is not fixed for any but the lowest level actions: the form of behaviors can be modelled at various levels of detail, with various combinatorial options. Execution of actions is prioritized according to where the "intention" to perform them originated, in the Reactive Layer, the Process Control Layer, or the Reflective Layer. The model as described here is not fixed; it is presented with enough flexibility as to be able to meet various demands on the implementation of its parts. Hopefully, this will allow it to become a guide to various approaches within the topic of face-to-face dialogue and real-time interaction. In the next chapter we will se one example of implementation of Ymir.

Ymir: An Implementation in LISP

8.

The Ymir architecture described in the last chapter has been implemented in Common Lisp [Steele 1990] and its object-oriented extension, CLOS (Common Lisp Object System) [Lawless & Miller 1991, Steele 1990, Keene 1989]. It allows for the desing of rules, modules and control structures to create a character that can interact face-to-face with a human. This chapter details the implementation of the "bare-bones" foundation for such character design: object classes, methods, and other software constructs, as well as the hardware setup.

8.1 Overview of Implementation

First we will give a short overview of the implementation and then go into further detail, showing particular algorithms and examples of software.

8.1.1 Simplifications

In this implementation of Ymir, which we will refer to as "Alpha", several simplifications have been made from the general model presented in the last two chapters. The main ones are:

1. Dialogue automatically takes precedence over any other task the agent may be involved with.
2. Only two conversing parties are assumed (computer character and person).
3. The dialogue is centered around a task where the computer character is the expert; the interaction is driven solely by the human.

Other smaller simplifications of the Ymir architecture will become apparent as we get into the details of the implementation.

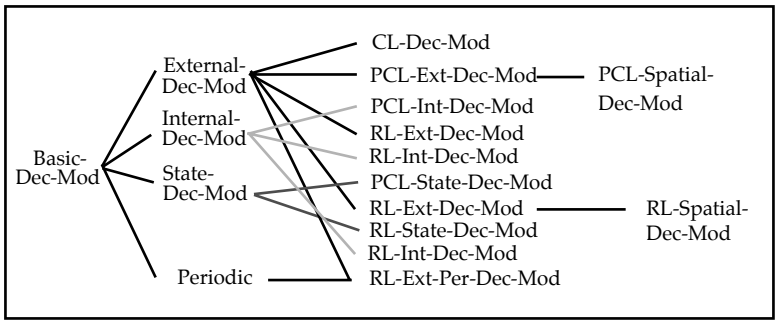


FIGURE 8-1. Types of behavior modules used in Ymir. Each level of behavior module inherits characteristics from the ones above and adds some of its own. (Dec-Mod = decision module.) See text for details on each type.

8.1.2 Hardware Overview

Two computers are used for implementation Alpha:

1. A Digital Equipment Corporation 3000/300 runs most of Ymir: Reactive Layer, Process Control Layer and Knowledge Base).
2. A Digital Equipment Corporation 5000/240 runs the Action Scheduler.

In addition, six peripheral computers are used for data collection and graphics; these are presented in the next chapter. (More detail on the hardware and software used can be found in Appendix A2 on page 213.)

8.1.3 Software Overview

The main elements of Ymir are implemented in Lisp [Steele 1990]. Supporting software, such as graphics [Thórisson 1996, see Appendix A1] and body tracking [Bers 1996], is implemented in C [Kernigan & Ritchie 1988] and C++ [Stroustrup 1991].

Perceptual and decision modules have been implemented as object classes, using the features of CLOS. Blackboards are simply lists of sublists—each sublist being a posting containing information on the form [MSGS STATE TIMESTAMP], where STATE is either TRUE or FALSE, MSGS is the name of a message from the perceptual modules and TIMESTAMP comes from a global clock, timed in centiseconds.

Several types of decision modules in the RL and PCL have been implemented (Figure 8-1). Modules inherit characteristics from their superclass and add some of their own. Basic operators such as POST and UPDATE are implemented as CLOS methods and specialized for each object type.

The Action Scheduler in Ymir Alpha runs on its own UNIX machine with a socket connection to the PCL. A prioritizing scheduler on the PCL side is used to ship out actions to the Action Scheduler. The AS uses an identical prioritizing scheme to send motor commands to the animation system (animation system is described in Appendix A1 on page 203). The AS also has a scheme for selecting between behavior morphologies and an interrupt feature which will allow it to provide output even if it hasn't looked at all the possible morphologies for a particular behavior. This happens if the expected lifetime of a behavior has been reached (see "A Notation System for Face-to-Face Dialogue Events" on page 108).

8.1.4 Top-Level Loop

Although Ymir is intended for a distributed implementation, the current version runs the RL, PCL and the CL on the same processor. Pseudocode for the top-level loop on this machine is shown in Algorithm 8-1, along with processes next-level down. Let's now take a closer look at each of the layers.

8.2 Reactive Layer

The RL contains perceptual modules (Virtual Sensors and Multimodal Descriptors) and Decision Modules.

8.2.1 Perceptual Modules

Perceptual modules have been implemented as a collection of Virtual Sensors and Multimodal Descriptors, also refered to as a Logic Net, because of their use of logic gates and Boolean output.

Operators for Multimodal Descriptors and Virtual Sensors

Two operators are defined for Virtual Sensors, UPDATE and POST. The UPDATE operator simply feeds the module with the required data. Each perceptual module has a destination for posting its state, msgs-dest. Multimodal Descriptors have four operators: UPDATE, POST, ACTIVATE and DEACTIVATE. If a module is ACTIVE and all conditions in its pre-conditions are TRUE (these are ANDed), then it is POSTed as TRUE to msgs-dest. When it becomes FALSE (one or more of its conditions are FALSE), it is POSTed as FALSE. Decision Modules in the Proces Control Layer determine when descriptors are ACTIVATEd and DEACTIVATEd.

```
TOP-LEVEL-LOOP
  UPDATE {ALL SENSORS}
  UPDATE {ALL ACTIVE DESCRIPTORS}
  UPDATE {ALL ACTIVE RL DECISION MODULES}
  UPDATE {ALL ACTIVE PCL DECISION MODULES}
  SEND {ALL ACTION-REQUESTS IN *BEHAVIOR-REQUESTS*}
     TO ACTION SCHEDULER

UPDATE [SENSOR]
  Evaluate-Raw-Input {SENSOR}
  if {different (CURRENT-STATE {SENSOR})(LAST-STATE {SENSOR})}
     then POST {SENSOR} TO FUNCTIONAL SKETCHBOARD

UPDATE [DESCRIPTOR]
  Evaluate-Conditions {DESCRIPTOR}
  if {different (CURRENT-STATE {DESCRIPTOR})(LAST-STATE {DESCRIPTOR})}
     then POST {DESCRIPTOR} TO FUNCTIONAL SKETCHBOARD

UPDATE [RL-EXTERNAL-DECISION-MODULE]
  Evaluate-Conditions {RL-EXT-DEC-MOD}
  if {different (CURRENT-STATE {RL-EXT-DEC-MOD})(LAST-STATE {RL-EXT-DEC-MOD})}
     then POST {EXT-RL-DEC-MOD} TO  *BEHAVIOR-REQUESTS*

UPDATE [PCL-EXTERNAL-DECISION-MODULE]
  Evaluate-Conditions {PCL-EXT-DEC-MOD}
  if {different (CURRENT-STATE {PCL-EXT-DEC-MOD})(LAST-STATE {PCL-EXT-DEC-MOD})}
     then POST {DEC-MOD} TO  *BEHAVIOR-REQUESTS*

UPDATE [RL-INTERNAL-DECISION-MODULE]
  Evaluate-Conditions {RL-INT-DEC-MOD}
  if {different (CURRENT-STATE {RL-INT-DEC-MOD})(LAST-STATE {RL-INT-DEC-MOD})}
     then EXECUTE {INTERNAL-ACTION {RL-INT-DEC-MOD}}

UPDATE [PCL-INTERNAL-DECISION-MODULE]
  Evaluate-Conditions {PCL-INT-DEC-MOD}
  if {different (CURRENT-STATE {PCL-INT-DEC-MOD})(LAST-STATE {PCL-INT-DEC-MOD})}
     then EXECUTE {INTERNAL-ACTION {PCL-INT-DEC-MOD}}
```

ALGORITHM 8-1. Top-level loop that handles all events in the Reactive, Process Control and Content layers. Notice that this loop is a serial implementation of a largely parallel system. *BEHAVIOR-REQUESTS* contains all actions that should be sent to the Action Scheduler.

Virtual Sensors

Virtual Sensors consist of two elements: {1} a custom-made function that pre-digests the data needed for the sensor, and {2} the module itself, a CLOS object, with pointers to where to find the necessary data, a pointer to the custom-made function which takes that data as arguments, its own state (TRUE or FALSE), as well as a time stamp for when the module was last **POST**ed (Figure 8-2).

```
(defclass body-sensor (var-ref)
 ((msgs :initform nil)
 (func :initform nil)
 (data1 :initform (vector 1 1 1))
 (data2 :initform (vector 1 1 1))
 (state :initform nil)))
```

FIGURE 8-2. A *body-sensor* class. The content of its data slots will be refreshed in each call to **UPDATE**.

The sensor classes implemented are:

1. Body-sensor-with-fixed-reference
2. Body-sensor-with-variable-reference
3. Prosody-sensor
4. Speech-sensor

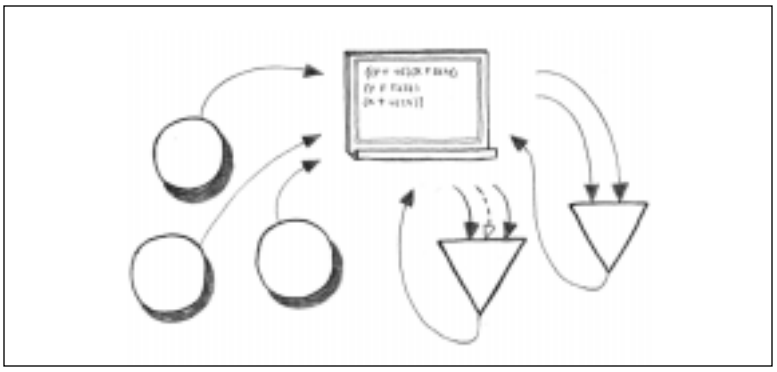


FIGURE 8-3. Virtual Sensors (circles) post their status on the Sketchboard, while Multimodal Descriptors read the Sketchboard to compute their own states, subsequently to be posted on the Sketchboard as well. A large collection of sensors and descriptors makes up a Logic Net (LN).

```
(defclass mm-descriptor ()
 ((msgs :initform nil)
 (pos-conds :initform '())
 (neg-conds :initform '())
 (state :initform nil)
 (stamp :initform 0)
 (active :initform T)
 (thresh :initform 1)))
```

FIGURE 8-4. The multimodal descriptor class. The `thresh` slot contains a value that determines how many of the conditions in neg-conds and pos-conds are needed to make the descriptor `POST` as TRUE to the blackboard.

Body sensors with a fixed reference track an object relative to a stationary object, such as the user's left hand in relation to the monitor. Those with variable reference track the relative spatial aspects of two objects in relation to each other, e.g. the position of the left hand in relation to the trunk. As explained in the last chapter, prosody sensors track some aspect of speech that has nothing to do with its propositional content, such as speech-on-off, intonation, etc.; speech sensors are related to the pragmatic and semantic aspects of speech. We will see example implementations of these sensor classes in the next chapter.

Multimodal Descriptors

Multimodal integration is handled by what can be thought of as a net of first-order, Boolean logic gates (see "Appendix: Logic Net" on page 126), which I will simply refer to as a logic net (LN, see last section in this chapter). The system uses a special syntax developed for easy construction and modification (by the agent designer as well as the run-time environment). The basic element of this net is the Multimodal Descriptor.

```
(looking-at-hands (pos-conds
 (looking-at-r-hand 0.5)
 (looking-at-l-hand 0.5))

(addressing-me (pos-conds
 (turned-to-me 1.0)
 (facing-me 1.0)
 (facing-domain 1.0)))
```

FIGURE 8-5. Examples of POS-CONDS lists of two multimodal descriptors. The first is a simple aggregator of two virtual sensors; the second is a heuristic for determining if the user's utterance is meant for the agent. The condition's score is added up and compared to the module's threshold to determine if the module is posted as true.

Only static descriptors have been implemented in Ymir Alpha. The descriptors have a set of positive and negative pre-conditions (Figure 8-4). Each condition has a value associated with it (Figure 8-5). When the descriptor is updated, each of the values for those conditions that are TRUE are added up; if they add up to more than the descriptor's pre-set threshold, the descriptor is set to TRUE; otherwise it is FALSE. There are two functionally distinct groups of descriptors, *static* and *dynamic*. Static descriptors simply respond to a static situation, whereas dynamic

descriptors detect patterns over time intervals, such as a specific gaze pattern or a combination of arm and eye movements for a given interval. The operators UPDATE, POST, ACTIVATE, DEACTIVATE are implemented as CLOS methods. To check a blackboard for a specific state being TRUE or FALSE, the operator Call-BB is used. It takes a single condition and returns TRUE if that condition was last posted as TRUE, and FALSE otherwise.

Communication via the Sketchboard

The sensors and descriptors communicate with each other via the Functional Sketchboard, where their states are POSTed with a time stamp every time they change. The messages, contained in the MSGS slot of the sensor or descriptor, its state (TRUE or FALSE) and a time stamp are included (e.g. [ADDRESSING-ME T 35415]). The Sketchboard thus accumulates a history of node states, which can be related to other time stamped events in the system, such as turn state or words spoken.

8.2.2 Decision Modules

The general model of Decision Modules is this: Each Decision Module has an associated intention and a condition list. If the conditions become true, the intention "fires". In terms of Ymir, this means that it either results in some internal process running or some outward behavior being executed. A module can be ACTIVE or INACTIVE. If it is ACTIVE, it will fire when the conditions are met; if it is INACTIVE it cannot fire.

Operators for Decision Modules

Four decision module operators are defined: UPDATE, POST, ACTIVATE, and DEACTIVATE. UPDATE supplies a module with access to all data it needs to make its decision. If a module is ACTIVE and its state is TRUE (i.e. all conditions in its FIRE-CONDS lists are met—these are ANDed), then its messages is POSTed to msgs-dest and the module subsequently DEACTIVATEd, and its STATE reset to FALSE. If the module is INACTIVE, the conditions in its reset lists (POS-RESTR-CONDS and NEG-RESTR-CONDS) are checked, and, if all of them are met (these are also ANDed), the module is ACTIVATEd.

```
(defclass basic-dec-mod ()
 ((msgs :initform nil)
 (state :initform nil)
 (active :initform nil)
 (el :initform 100)
 (stamp :initform 0)
 (pos-conds :initform '())
 (neg-conds :intiform '())
 (pos-restr-conds :initform '())
 (neg-restr-conds :initform '())
 ))
```

FIGURE 8-6. The decision-module class. (See also Figure 8-1.)

Decision Module Structure

Decision Modules have been implemented as CLOS classes. Specializations for decision modules are (Figure 8-1):

1. Internal Decision Modules
2. External Decision Modules
3. State Decision Modules

4. Periodic Decision Modules

Each decision module is associated with one intention (I), whose name is kept in the MSGS slot (Figure 8-6). A module has a STATE slot that shows whether it has been fired or not, i.e. whether all the conditions in the POS-CONDS and NEG-CONDS lists have been met simultaneously. If they have, the module's STATE is set to TRUE. In this state, the module is not executable—the conditions in POS-RESTR-CONDS and NEG-RESTR-CONDS LISTS determine whether we restore the module's state to executable (Figure 8-6).

The following general model captures the design philosophy of the decision modules: A module can have four boolean states: ACTIVE or INACTIVE (mutually exclusive), and TRUE or FALSE (also mutually exclusive). It has seven slots: [1] POS-FIRE-CONDS and [2] NEG-FIRE-CONDS, two lists of conditions that, when all conditions in the first are TRUE and those in the second FALSE, will make the module fire (turn its own state to TRUE), [3] POS-RESET-CONDS and [4] NEG-RESET-CONDS, two lists of conditions that, when all conditions in [3] are TRUE and all in [4] are FALSE will make the module active, [5] STATE, containing the state of the module (either TRUE or FALSE), [6] MSGS, containing the message that is posted when the module changes state, and [7] MSGS-DEST, containing a pointer to `msgs-dest`, the destination for its messages (either a blackboard or the Action Scheduler).

To deal with unpredictable time delays from the time when a decision is made to execute an act until it reaches its final stage of being sent to the "muscles", the Action Scheduler uses an action's expected lifetime (A_{el}). This time-out specifies the total time the behavior can stay in the system before reaching the motors (or muscles).

Time stamping is performed whenever a module's action is `POST`ed, and can be used by any of the other decision modules for activating behaviors based on the age of messages reported on the blackboard. The operator `Call-BB-Time` gives the last posting time for a given module, and can be put in any of a decision module's condition lists.

Periodic Decision Modules

Periodic modules simply `POST` their MSGS at a fixed frequency whenever their conditions are met. This is useful for recurring behaviors such as blinking.[1] One could, for example, define conditions that "make the character sleepy", and during these conditions it blinks at a slower frequency than when engaged in conversation.

Internal and External Decision Modules

External behaviors POST messages to a buffer, subsequently to be shipped to the Action Scheduler; internal behaviors execute various internal-acting procedures that are run inside the top-level loop (see above). Behaviors on the action request list are therefore visible actions; internal ones are invisible to the user.

External modules contain the name of a behavior in their MSGS slot, which gets put on a *BEHAVIOR-REQUESTS* cue list and sent to the Action Scheduler. The fact that these are all of type reactive makes their priority in the Action Scheduler (and on the cue list for getting shipped to the AS) the highest.

Internal modules contain a *function* name in their MSGS slot; when the module is POSTed this function is Funcalled. This is implemented by using two separate methods for the UPDATE operator, which take each kind of module.

State Decision Modules

```
(defclass state-dec-mod
                (basic-dec-mod)
 ((next :initform '())
 (active-mm-descrs :initform '())
 ))
```

FIGURE 8-7. The *state* specialization of the basic decision module.

A problem with the above modules is that they can't cause internal conditions as a function of being in a particular state. This would be useful for conditions that are mutually exclusive, such as Dialogue-On/Dialogue-Off, yet come with different requirements for perceptual activities and sub-processes. To solve this, state decision modules are made for keeping track of things such as dialogue state, turn state, etc., and switching on and off the right kinds of perceptual processing. They can be thought of the transition rules in an ATN (augmented transition network) with the difference that they can lead to more than one new state (Figure 8-6). To take an example, the module Dialogue-Off can be made to transition to two states, Dialogue-On and User-Has-Turn. State modules switch multimodal descriptors between being ACTIVE and INACTIVE (the ones that are to be active during the state are stored in a list in the ACTIVE-MM-DESCR slot). Since not all descriptors should be ACTIVE during all states, this provides a mechanism for a sort of "narrowing of attention" for the agent, as well as freeing up processor cycle time.

1. A more intelligent solution to controlling blinking is representing the moisture of its eyes with an evaporation constant and a decision module that reads this simulated moisture and decides to blink when it goes below a certain threshold. This would also be more in keeping with the philosophy of the whole system, the "event-driven" model of control, but is not necessary for the purposes of communicative dialogue.

8.3 Process Control Layer

The PCL is for the most part identical to the Reactive Layer, except that no special perceptual processes are implemented at this level. Interesting future candidates for advanced perceptual processes would be ones that monitor the performance of groups of decision modules and, in conjunction with other decision modules, can modify the ones that don't perform well. Such systems have been called B-Brains [Minsky 1985].

8.3.1 Decision Modules

Decision modules in the PCL are the same types as those in the Reactive Layer. The only difference is that these can look for conditions in both the Functional Sketchboard and the Content Blackboard.

8.3.2 Communication via the Content Blackboard

In the current implementation, messages from the PCL to the CL and from the CL to the PCL are posted to the Content Blackboard. Currently the Content Layer posts more messages to the PCL than vice versa (Figure 8-8). As we established in the last chapter, the PCL posts messages to the Content Layer about the status of the interaction. These are messages about the turn status, for example: A short utterance would be more likely to be a back channel if the agent was speaking—this knowledge could be used in real-time by the CL by weighting the vocabulary toward back channel feedback utterances, thus increasing the probability of correct recognition. Such a setup has not been tested yet in Ymir, but is being investigated.

FROM KB TO PCL
Rcv-Speech
Speech-Data-Avail
KB-Succ-Parse
KB-Exec-Act
CL-Act-Avail
KB-Exec-Act
TKB-Act-Avail
TKB-Exec-Speech-Act
TKB-Exec-World-Act
DKB-Exec-Act
Exec-Done

FIGURE 8-8. A basic set of communication primitives between the Process Control Layer and a Topic Knowledge Base.
Rcv = received, Act = action; Exec =executing; succ = succsessful.

8.4 Content Layer

The Content Layer contains what can be thought of as a combined DKB and TKB. No experiments have yet been done with multiple knowledge bases or multiple topics. The Content Blackboard, which is used by the CL and the PCL, was discussed above.

8.4.1 Dialogue Knowledge Base

A very minimal knowledge for interaction has been implemented. It only contains knowledge about greetings and good-byes. Its small size made it simplest to integrate directly with a topic knowledge base in the case of Gandalf. Gandalf's topic knowledge (section 9.1, page 105),

```
(defclass behavior ()
  ((name   :accessor name   :initarg :name   :initform nil)
   (acts   :accessor acts   :initarg :acts   :initform nil)
   (delay  :accessor delay  :initarg :delay  :initform 0)
   (execution-time :accessor exec-time  :initform nil)))
```

FIGURE 8-9. The generic *behavior* class.

revolves around simple facts about the solar system, such as how big planets are, how many moons they have, etc. It will be described in the next chapter.

8.4.2 The Topic Knowledge Base

The outcome of any interpretation-response generated by a TBK is sent directly to the virtual world (initiated by decision modules in the PCL), without going through the agent's motor mechanisms. This makes it very easy to accommodate multiple knowledge bases for diverse virtual environments (e-mail, graphics, audio, movie clips, etc.) without having to encode these skills in terms of complex end-effector actions such as would be the case if we wanted an all-purpose physical robot

Communication between the TKB and the PCL is handled with a set of pre-determined communication primitives (Figure 8-8) as mentioned above.

8.5 Action Scheduler

The AS keeps track of the facial state and uses knowledge about which layer initiated the action request, as well as the age of the action request, to compose a viable motor scheme for satisfying it. The process of going from a high-level "intention" to an actual motor act is called "morphing" for lack of a better term. Because there is no feedback mechanism between the AS and the decision making layers in the current implementation, feedback about the decision's success has to be gotten by sensing the state of the user. This scheme seems to work well with very simple knowledge bases, but will probably break down for more complex characters.

8.5.1 Behaviors

Behaviors are implemented as CLOS objects (Figure 8-9). When designed, a list of these is created in a general format (Figure 8-10 on page 123), and a make function then called on the list. We refer to the

collection of behaviors generated as CLOS objects as the agent's *Behavior Lexicon* (Figure 8-10).

Behaviors in Ymir Alpha are of two kinds:

1. `act` behaviors and
2. `mot-lev` behaviors.

The former contain leaf nodes, i.e. motors, to be moved for generating a particular behavior. These have a direct mapping to a particular motor configuration (dynamic or static), e.g. a facial expression. The latter are abstractions of behaviors that generally have more than one way to be realized. Together, these form a tree. A behavior that subsumes a `mot-lev` behavior is always one level above the leaves of the tree.

Behavior modules above the motor-level (i.e. `act`) have a [1] NAME, and [2] OPTIONS—a list of behaviors that can be used in morphing the behavior; each option is a list of behaviors, which is a list of the form [NAME, EXEC-TIME, DEALY], where NAME is the behavior's name, EXEC-TIME is the execution time for that behavior, and DELAY is a time-delay that offsets this behavior's execution from the execution of the behavior that subsumes it.

Each `mot-lev` action has a [1] NAME—the behavior's unique name, [2] MOTOR-LIST—a list of the motors involved. Each motor in this list contains [1] MOTOR-NAME—the motor's unique name, [2] EXEC-TIME—its default execution time, and [3] REL-POS—the motor's goal position, relative to its range of motion.

`act` behaviors can contain replacement execution times for the behaviors they subsume. Thus, when the behavior `eyes-neutral` is executed as part of the higher-up behavior `face-neutral`, it takes 400 ms for the motors to get to their final position, but when `eyes-neutral` is called directly it takes 100 ms. When the execution times differ for each motor, the longest motor would take the *max* execution time (400 ms in the example above), while the others would be a percentage of that. Using this scheme, a behavior's EXEC-TIME could be recalculated when composing a final action, e.g. in light of time-constraints, but this feature has not been implemented.

8.5.2 Behavior Requests

In Ymir Alpha, behavior requests are received in the Action Scheduler over a socket connection. They are put on a buffer, *PENDING*[2], where

2. This buffer has the corresponding *BEHAVIOR-REQUESTS* output buffer on the PCL side.

```
(defun prioritize-incoming (PEND)
 (let* ((return nil)
        (RL        1)
        (layer    RL)
        (satisfied nil))
  (loop while (not satisfied) do
   (dolist (act PEND)
     (if (eq (second act) layer)
         (setf satisfied t
           return act)))
   (setf layer (1+ layer)))
  (setf PEND
   (remove return PEND))
  return))
```

ALGORITHM 8-2. Lisp-code for scheduling actions in the Action Scheduler. The procedure `prioritise-incoming` receives the *PENDING* list, returns an act to work on, prioritized by the system that initiated it.

they are serviced based on who—RL, PCL or CL—created the request (Algorithm 8-2). The algorithm simply services all RL requests until there are none left, then it executes one PCL request, if there is one. If there are no RL or PCL requests to execute, it executes a content-related (CL) request.

Requests are received in the form [ACTION-NAME TIME-STAMP EXPECTED-LIFETIME WHO], where WHO is one of RL, PCL or CL. If the expected lifetime has been reached, and no `mot-lev` behavior has been found for it yet, the action is cancelled:

$$\textbf{Cancel}\ \{A\}\ \text{IF (TIME-NOW} > \text{TIME-STAMP}_A + \text{EL}_A). \qquad \{8.1\}$$

No feedback is sent back to the originator of the action whether this action was executed or not. This information is expected to flow back through the virtual sensors as a particular reaction of the user to the lack of behavior. Since there is a tight loop of functional analysis going in to the agent, any problem in the global aspects of dialogue should show up there instantaneously and a new action would be triggered.[3] Thus, the need for complex book-keeping protocols *within* the system—as opposed to through the *outer* feedback loop by way of the effect the agent's behavior has on the user's behavior—should be diminished, if not eliminated.

8.5.3 Generating Behavior Morphologies

When a behavior request is selected to be executed, its name is looked up in the Behavior Lexicon (Figure 8-10). The AS will look at the options available for an action and select one that interferes the least with the ongoing actions of the agent's communicative features such as brows, gaze, hands, mouth, etc. This recursive algorithm works as follows: First it checks in the lexicon if the behavior to be morphed has more than one option. If it has, it takes the first option and goes down one level, again checking for options. Once it reaches the leaves, it pops back up and finds the leaves (motors) for the next option. When it has found motor actions for two options, it selects. If there are more options for the current level, it repeats this until the best one is left. Then it pops up one more level and continues.

This process can only be terminated after at least one option has been traced down to the leaves. It terminates under two conditions: {1} if there are no more options to select between, and {2} if the expected lifetime of the action has been exceeded. The latter is checked every time a

3. By designing cascaded decision modules for reactive behaviors, failures in the interaction are met with "pre-compiled" reactions. See "Creating Behavior Classes with Cascaded Decision Modules" on page 104.

```
(setf *Behavior-Lexicon*
  ;ACT TEMPLATE:
   ;(name class (((act-name-of-option-1 delay exec-time)
   ;              (act-name delay exec-time) etc*)
   ;              (etc*)))
  ;MOTOR TEMPLATE:
   ;(motor-name class delay exec-time pos/data)
   '(
     ; MORPHOLOGICAL DEFINITIONS
       ;Features
         ;neutral
     (face-neutral act
                      (((mouth-neutral 100 400)
                        (eyes-neutral 0 300)
                        (brows-neutral 0 500))))
     (brows-neutral act
                      (((left-brow-neutral 0 400)
                        (right-brow-neutral 0 400))))
     (left-brow-neutral mot-lev
                      (((Bll 0 400 30)
                        (Blc 0 400 30)    ;Brow, left, central
                        (Blm 0 400 30)))) ;Brow, left, medial
     (right-brow-neutral mot-lev
                      (((Brm 0 400 30)
                        (Brc 0 400 30)
                        (Brl 0 400 30))))
     (eyes-neutral act
                      (((upper-lids-neutral 0 100)
                        (lower-lids-neutral 0 100))))
     . . . .
```

FIGURE 8-10. A short segment of the behavior lexicon for Gandalf (see next chapter). Behaviors that are above the leafs are marked as "act"; behaviors that contain only motor commands are marked "mot-lev". (Figure 8-11 shows the names of the facial motors.) A list like *Behavior-Lexicon* is given as an argument to a function that automatically creates CLOS behavior objects. (A full listing of Gandalf's behavior modules is given on page 153.)

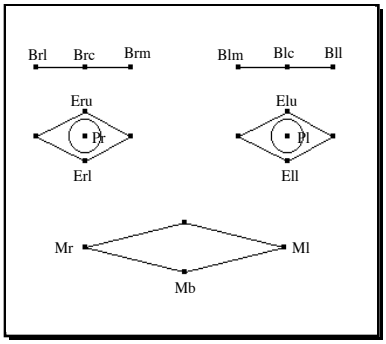


FIGURE 8-11. Movable control points—or motors—are coded as shown: Bll = brow, left, lateral; Blc = brow, left, central; Blm = brow, left, medial; Elu = eye, left, upper; Ell = eye, left, lower; Pl = pupil, left; Ml = mouth, left; Mr = mouth, right; Mb = mouth, bottom. Brow, pupil and eye are mirrored on the right side of the face. Head motion is coded as H. All motors are referenced with an absolute position from 0 to 100. Motors with two degrees of freedom are addressed by either h or v, for horizontal and vertical motion, respectively. All motors can be addressed and run in parallel. (See "Character Animation" on page 203.)

selection between two options has been performed. Upon termination, the leaves (motors commands) get sent to the animation unit, and the character behaves.

8.5.4 Motor Control in the Action Scheduler

The animation unit can be thought of as the character's muscles: it makes sure that the behavior takes the time it is supposed to take, as defined in the EXEC-TIME. Normally all motor specifications of a behavior get sent to the animation unit at the same time. An example of such a behavior is the behavior **Smile**: corners of the mouth are moved upward and outward, while the lower eyelids are moved slightly up from their resting position, all at once. For sequential actions, certain motor commands may have to wait for others to finish. The DELAY of a

motor determines how long after the whole action started it should begin execution. An example of a behavior that uses this feature is the behavior Blink: first the eye is closed, then opened.

Speech

Just as motor commands are ballistic once they leave the AS, speech leaving the AS is also ballistic. It is therefore important that the speech is segmented correctly to allow for cancellations in case the user interrupts the agent. Currently, this is done at natural boundaries larger than the word but shorter than the sentence. Noun phrases, verb phrases and fillers are all sub-components that give useful (albeit not *always* appropriate) boundaries. How the AS controls the incremental execution of long actions, and what its communication with the Topic Knowledge Bases would be remains an issue of further research.

8.5.5 Motor Programs: Animation Unit

The animation unit provides the agent with muscles. It receives commands from Action Scheduler in the form [MOTOR, POSITION, TIME], where motor is the motor to move, position is the new (absolute) position it should move to and time is the absolute time it should take to get there. The commands received by this unit are ballistic (except for manual gesture—see below).

Manual Gesture Control

The ideal way to animate a hand would make use of a representation of the hand that incorporated motors in all finger joints, as well as those of the arm. This is in fact a serious research area in the robotics industry, gesture recognition and gesture generation [Cassell et al. 1994, Wexelblatt 1994, Sparrell 1993, Cutkosky 1992] and will not be dealt with here in any depth. The problem is simplified in Ymir Alpha by representing separately the hand's *position* and *shape*, and by giving the hand two states, *at-rest* and *active*. Whenever the animation module receives a command for a manual gesture it will execute the given type of gesture for the requested time period, after which it moves it back to its *at-rest* position.[4]

The gestures also have some controllable parameters, such as a *pitch* and *yaw* for deictic gestures, and *duration* for beat gestures. Gestural interruptability has been implemented: If a gesture is executing when a new hand gesture command arrives, the current action will be cancelled, and the new command will take over. The shape of the hand is interpo-

4. Thanks to Hannes Vilhjálmsson for his contributions to and implementation of the hand and the gesturing mechanism.

lated from its current state to the shape associated with the first position in the new gesture, while the hand is moved linearly from its current position to the first position of the new command. This scheme works surprisingly well considering its simplicity. (See also "Character Animation" on page 203.)

8.6 Appendix: Logic Net

8.6.1 Syntax

As mentioned above, the logic of the virtual sensors and multimodal descriptors are implemented in a custom-designed syntax. Figure 8-1 shows how common logic gates can be built from this syntax. The advantages to this syntax, as opposed to for example using the logical operands of Lisp, are mainly related to ease of manipulation, by the developer and the run-time system itself. Possible future enhancements that are facilitated by the approach are:

1. Weights could be used to change thresholds of both condition lists and nodes at run-time.

TABLE 8-1. Common logic gates and their equivalents in Logic Net Syntax. Values of the conditions are added; threshold is assumed to be 1.0.

LOGIC GATE	TRUTH TABLE	LOGIC NET SYNTAX
AND	A B |OUT L L | L L H | L H L | L H H | H	POS[A 0.5][B 0.5] NEG[]
NAND	A B |OUT L L | H L H | H H L | H H H | L	POS[] NEG[A 1.0][B 1.0]
OR	A B |OUT L L | L L H | H H L | H H H | H	POS[A 1.0][B 1.0] NEG[]
NOR	A B |OUT L L | H L H | L H L | L H H | L	POS[] NEG[A 0.5][B 0.5]
XOR	A B |OUT L L | L L H | H H L | H H H | L	POS[A 0.6][B 0.6] } D1 NEG[A 0.4][B 0.4] POS[D1 0.4] NEG[A 0.6][B 0.6]
XNOR	A B |OUT L L | H L H | L H L | L H H | H	POS[A 0.6][B 0.6] } D1 NEG[A 0.4][B 0.4] POS[A 0.6][B 0.6] NEG[D1 0.4]

2. Continuous-value sensors and descriptors could be integrated more easily, while the Boolean nature of the network could still be preserved at the highest descriptor level.
3. It is easier to implement a learning mechanism for number-based logic gates.
4. Time-based descriptors can more easily be added.

8.6.2 Logic Net: Any Alternatives?

A choice was made to use a custom-designed logic net (LN) as the backbone of the perceptual system. Other contenders for the task of the logic net include fuzzy logic [Kacprzyk 1992] and Fuzzy Cognitive Maps (FCMs) [Dickerson & Kosko 1994]. LNs do not make use of membership functions like fuzzy logic, or graded feature vectors like FCMs, and are therefore simpler to develop and modify. Whether LNs are a sufficiently powerful mechanism for extracting functional aspects of dialogue is an empirical question beyond the scope of this thesis, and will hopefully be settled in future psychological research. One important advantage of using Boolean nodes at the sensory stage is that we can use the states of the modules to track the history of the functional interpretive process: Each time a sensor or descriptor node changes, its state is posted to a blackboard where the other nodes can subsequently read its current state. This allows us to track the progress and path of a particular interaction sequence, as well as to summarize and store the history for future reference. These are key elements in allowing the system to learn over time and to allow the user or the agent to reference past dialogue events. With a continuous (non-Boolean) system, where features are detected to a certain degree and cannot be treated as crisp events that either happen or not happen, post-analysis of internal history becomes problematic, or at least much more complex.[5]

Another advantage of this approach is the possibility of a parallel implementation. Since the nodes are modeled as objects that communicate asynchronously via a common blackboard, parallel implementation could speed up the execution of the system and increase reliability of overall system performance, as well as make more complex sensors and descriptors a viable option, all without slowing computation.

A third advantage is the ease with which new sensors and descriptors can be developed and added without disrupting the ones that are already in place. But first and foremost, it is fast.

5. An interesting question here is how we remember events to have either occurred or not occurred—is a vague memory of an event an indication that continuous-scale memory indices are at work?

Gandalf: Humanoid One

9.

Using Ymir as a foundation, a character can be built by specifying perceptual, decision and behavior modules, plus the specific procedures needed for data and internal computations. In this chapter we will present the first character designed in Ymir, *Gandalf*. Gandalf (lower right corner, this page) is a "straw-cartoon"—it has been given the minimal set of modules necessary for face-to-face interaction.[1] It shows that Ymir is a sufficient and appropriate platform for developing communicative characters. It also points to issues that need to be addressed further in future research. These will be discussed in the conclusion chapter, page 185.

The computer ... can give you the exact mathematical design, but what's missing is the eyebrows.

—Frank Zappa

Why "Gandalf"?

As told in the Icelandic Sagas [Sturluson 1300~1325], after the gods Ó›inn, Vili and Vé had killed the giant Ymir (pronounced e-mir), and used his carcas to make the heaven and the earth, worms sprung to life in the newly created soil. The gods subsequently changed these worms to dwarfs, 63 of them to be exact. Gandalf—or *Gandalfr*—was one of these worms-turned-to-dwarf creatures (predating J.R.R. Tolkien's [1937] character of the same name by about a millenium :-)

9.1 The Gandalf Prototype: Overview

Gandalf is a collection of control rules implemented in Ymir Alpha, in Lisp (Chapter 8.), plus the necessary hardware and support software for sensing, acting and embodiment.

9.1.1 Prototype Setup

Gandalf appears to users on its own monitor (Figure 9-1). A model of the solar system (Figure 9-2) appears on a large screen in front of the user. Gandalf is an expert in the solar system and can tell users facts about the planets. It can also travel to the planets, zoom in and out, and start and stop the planets' moons in their orbits. The speech recognition used is a speaker-independent, continuous speech recognizer from BBN called HARK [BBN 1993]. This recognizer is grammar based and has

1. See also Chapter 10.3.2, page 162 about analysis and action schemes for responding to deictic gestures.

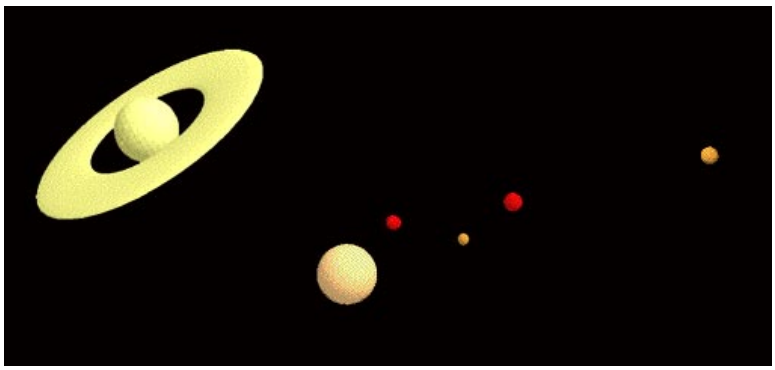
FIGURE 9-1. A user gets ready for interacting with Gandalf.

been programmed to recognize utterances like "Take me to Jupiter" and "Tell me about Saturn". Intonation analysis is performed by a custom-built analyzer. Gandalf "sees" the user via a body-tracking suit that uses magnetic space-sensing cubes, and an eye tracker that the user wears [Bers 1995, 1996]. By mapping out Gandalf's monitor in real-space he knows his own position in real space. Mapping the big-screen display allows him to know the real-space position of the planets.

Eight computers are used to run Gandalf's software:

1. A Digital Equipment Corporation Alpha 3000/300 workstation runs most of Ymir: Reactive Layer, Process Control Layer and Knowledge Base.
2. A Digital Equipment Corporation 5000/240 workstation runs the Action Scheduler.

FIGURE 9-2. Gandalf knows general facts about our solar system. It commands a graphical model of the planets with logarithmically scaled distances and moon sizes.



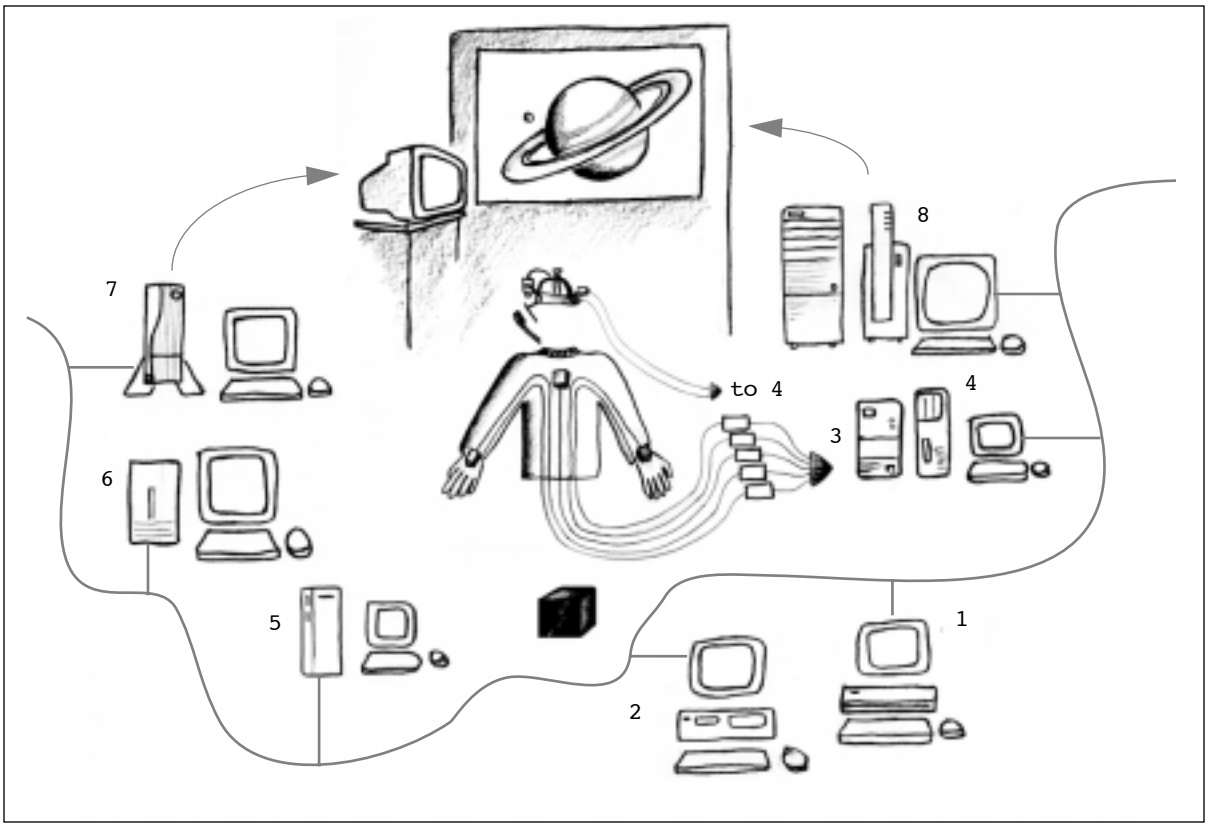


FIGURE 9-3. Gandalf system layout. Grey arrows show display connections; grey line is Ethernet. Eye tracker is connected to computer 4 via a serial port and the data is fed to computer 3 via another serial connection; output from the jakcet is connected to computer 3 via serial connections (dark arrowheads). Black cube (connected to computer 3) generates the magnetic field for sensing the posture of user's upper body. Microphone signal is split to two computers (6 & 7) via an audio mixer (connections not shown).

Legend
1. DEC 5000/240
2. DEC Alpha 3000/300
3. PC (Intel 486)
4. PC (Intel 386)
5. Macintosh Quadra 950
6. SGI Iris
7. SGI Indigo
8. HP Apollo 9000/750

3. A PC is used to collect data from a body tracking suit.
4. A PC is used to collect data from an eye tracker.
5. A Macintosh Quadra 950, with a Roland CP-40 pitch-to-MIDI converter is used to track and analyze intonation.
6. A Silicon Graphics Iris is used for speech recognition.
7. A Silicon Graphics Indigo2 is used to animate the character.
8. A Hewlett-Packard Apollo 9000/750 animates a virtual solar system.

The configuration is presented graphically in Figure 9-3. An Ethernet connection provides data flow between all computers (a serial line con-

nects computers 3 & 4). Further detail on the hardware and software can be found in Appendix A2 on page 203.

9.2 Gandalf: Technical Description

Gandalf consists of a collection of virtual sensors, decision modules and a behavior lexicon. Its knowledge base, whose mechanisms were mentioned in the last chapter, is also presented here in full detail, along with each module and communication primitives. Most of Gandalf's modules aimed toward generating appropriate turn taking and maintaining the flow of the interaction: very little has been done in terms of providing it with sophisticated perceptual mechanisms or extensive topic knowledge. Needless to say, Ymir provides the necessary hooks to integrate such mechanisms into the control of the agent's behavior. To give the reader a complete picture of what it takes to design a (minimal) agent in Ymir, in this chapter we will look at all the modules needed to get Gandalf working.

9.2.1 Where do Gandalf's Control Rules Come From?

Gandalf's modules are designed with the single purpose of supporting intelligent dialogue behavior on behalf of the agent: enabling it to look where you're pointing[1], glance at you when you're done speaking, turn its head back to you when it's finished doing what you asked it to do, etc. For the most part, this was done by data-mining the psychological literature (Chapter 3., page 37). Most of its Decision and Behavior modules, therefore, trace their origin to one or more papers reviewed in that chapter. Some fine-tuning and modification of modules was also performed after user testing ("Human Subjects Experiment" on page 147).

9.2.2 Virtual Sensors

Gandalf has a total of 16 virtual sensors (Table 9-1), of which 3 are prosody sensors, 9 are body sensors and 3 are speech sensors. One sensor is specialized for monitoring body data input, and is called vision-sensor (sensor number 10, Table 9-1). It allows Gandalf to know whether its vision is working properly. It POSTs FALSE if any of the data from the body tracking stops updating normally.

The prosody sensors are added for homogeneity; the features they represent are computed on a separate machine (see above) and sent over a

1. See Table 10-1 on page 172 for perception and action modules designed for deictic gestures.

NAME: facing-work-screen **1** TYPE: body-sensor-fix-ref DATA-1: nil DATA-2: work-screen INDEX-1: get-head-direction INDEX-2: nil FUNC: facing?	NAME: looking-at-me **2** TYPE: body-sensor-fix-ref DATA-1: nil DATA-2: agent-screen INDEX-1: get-gaze-direction INDEX-2: nil FUNC: u-looking-at-me?
NAME: facing-me **3** TYPE:body-sensor-fix-ref DATA-1: nil DATA-2: agent-screen INDEX-1: get-head-direction INDEX-2: nil FUNC: facing?	NAME: r-hand-in-gest-space **4** TYPE: body-sensor-var-ref DATA-1: nil DATA-2: nil INDEX-1: get-r-wrist-position INDEX-2: get-trunk-direction FUNC: hand-in-gest-space?
NAME: turned-to-me **5** TYPE: body-sensor-fix-ref DATA-1: nil DATA-2: agent-screen INDEX-1: get-trunk-direction INDEX-2: nil FUNC: turned-to?	NAME: l-hand-in-gest-space **6** TYPE: body-sensor-var-ref DATA-1: nil DATA-2: nil INDEX-1: get-l-wrist-position INDEX-2: get-trunk-direction FUNC: hand-in-gest-space?
NAME: complete-synt **7** TYPE: speech-sensor DATA-1: nil INDEX-1: index-1 FUNC: syntax-complete?	NAME: speaking **8** TYPE: prosody-sensor DATA-1: nil INDEX-1: speech-on-idx FUNC: u-speaking?
NAME: looking-at-l-hand **9** TYPE: body-sensor-var-ref DATA-1: nil DATA-2: nil INDEX-1: get-gaze-direction INDEX-2: get-l-wrist-position FUNC: u-looking-at-hand?	NAME: see-user **10** TYPE: vision-sensor DATA-1: *socket-object1* DATA-2: nil INDEX-1: nil INDEX-2: nil FUNC: body-socket-monitor
NAME: facing-domain **11** TYPE: body-sensor-fix-ref DATA-1: nil DATA-2: work-screen INDEX-1: get-head-direction INDEX-2: nil FUNC: facing?	NAME: looking-at-r-hand **12** TYPE: body-sensor-var-ref DATA-1: nil DATA-2: nil INDEX-1: get-gaze-direction INDEX-2: get-r-wrist-position FUNC: u-looking-at-hand?
NAME: complete-gram **13** TYPE: speech-sensor DATA-1: nil INDEX-1: index-2 FUNC: grammar-complete?	NAME: complete-pragm **14** TYPE: speech-sensor DATA-1: nil INDEX-1: index-3 FUNC: pragmatics-complete?
NAME: intonation-up **15** TYPE: prosody-sensor DATA-1: nil INDEX-1: nil FUNC: inton-direction?	NAME: intonation-down **16** TYPE: prosody-sensor DATA-1: nil INDEX-1: nil FUNC: inton-direction?

TABLE 9-1. Virtual sensors used in the Gandalf prototype. The "agent-screen" variable holds the plane of the Gandalf's monitor (substituting for the plane of its face).

socket. These sensors provide a simple way to receive the intonation data and POST it to the Sketchboard.

Sensor compl-pragm is currently always true—there is no checking for pragmatic correctness yet in Gandalf. In addition to sensors 13 and 14, a Multimodal Descriptor called complete-utter POST TRUE immediately after the user stops speaking, unless either 13 or 14 are false.

9.2.3 Multimodal Descriptors

Gandalf has a total of 10 Multimodal Descriptors (Table 9-3, page 136). Module 7, is-active, is an example of a descriptor that takes the state of another Descriptor as well as that of a Sensor into account. Module 9, complete-utter, works in a very primitive way: if a template in either the Dialogue Knowledge Base or the Topic Knowledge Base has been filled completely, it POSTs as TRUE. The full implementation of this scheme is currently precluded by the inability of the speech recognition to recognize words incrementally. In an ideal system, this module would allow a character to respond with appropriate facial expression or body language—even verbally—before actually generating a complete response to the content of an utterance.

9.2.4 Decision Modules

Gandalf has a total of 35 Decision Modules. Of these, 16 belong to the Reactive Layer and 19 to the Process Control Layer. The three types of Decision Modules are: State Decision Modules, Internal Decision Modules and External Decision Modules.

State Decision Modules

Six Decision Modules are used to keep states (Figure 9-7 on page 144). These are based on a generalized finite state machine approach, where one or more State Decision Modules are associated with a single state. Each state module has an associated list of perceptual modules (Multimodal Descriptors) that should be active while that module is true. When two or more modules are activated, all the perceptual modules associated with the old state are turned off, and the perceptual modules in the newly entered state modules are turned on (meaning they are called on every update in the main loop). This way various internal processes can be turned on and off depending on the particular conditions of a collection of state modules, which in turn are only active during their associated state.

NAME: giving-turn POS-CONDS: (looking-at-me 0.3)(facing-me 0.3) NEG-CONDS: (gesturing 0.3)(speaking 0.4) THRESH: 1.0	**1**
NAME: taking-turn POS-CONDS: (gesturing 1.0)(speaking 1.0) NEG-CONDS: (looking-at-me 0.5) THRESH: 1.0	**2**
NAME: wanting-turn POS-CONDS: (speaking 0.5)(hand-in-gest-space 0.6) NEG-CONDS: nil THRESH: 1.0	**3**
NAME: want-back-ch-feedb POS-CONDS: (looking-at-me 0.5)(speaking 0.5) NEG-CONDS: nil THRESH: 1.0	**4**
NAME: looking-at-hands POS-CONDS: (u-looking-at-r-hand 0.5)(u-looking-at-l-hand 0.5) NEG-CONDS: nil THRESH: 1.0	**5**
NAME: hand-in-gest-space POS-CONDS: (l-hand-in-gest-space 0.5)(r-hand-in-gest-space 0.5) NEG-CONDS: nil THRESH: 1.0	**6**
NAME: is-active POS-CONDS: (hand-in-gest-space 1.0)(speaking 1.0) NEG-CONDS: nil THRESH: 1.0	**7**
NAME: gesturing POS-CONDS: (hand-in-gest-space 0.5)(speaking 0.6) NEG-CONDS: nil THRESH: 1.0	**8**
NAME: complete-utter POS-CONDS: (complete-pragm 0.5)(complete-syntax 0.5) NEG-CONDS: nil THRESH: 1.0	**9**
NAME: addressing-me POS-CONDS: (turned-to-me 1.0)(facing-me 1.0)(facing-domain 1.0) NEG-CONDS: nil THRESH: 1.0	**10**

TABLE 9-2. Multimodal Descriptors used in the Gandalf prototype. In descriptor 10, any one of its conditions will trigger an `addr-me` message to get **POST**ed to the Functional Scketchboard.

NAME: take-turn-1 TYPE: RL-State-Dec-Mod MSGS: take-turn NEXT-STATES: (give-turn dial-on) POS-CONDS: (addressing-me wanting-turn) NEG-CONDS: (KB-Exe-Act Spch-Data-Avail) ACTIVE-DESCR: (taking-turn wanting-turn gesturing addressing-me saying-goodbye concluding hand-in-gest-space is-active looking-at-hands)	**1**
NAME: take-turn-2 TYPE: RL-State-Dec-Mod MSGS: take-turn NEXT-STATES: (give-turn dial-on) POS-CONDS: (addressing-me wanting-turn) NEG-CONDS: (KB-Exe-Act) ACTIVE-DESCR: (taking-turn wanting-turn gesturing addressing-me saying-goodbye concluding hand-in-gest-space is-active looking-at-hands)	**2**
NAME: give-turn-1 TYPE: RL-State-Dec-Mod MSGS: give-turn NEXT-STATES: (take-turn dial-on) POS-CONDS: ((`FS-time-since` 'speaking 50) giving-turn) NEG-CONDS: (taking-turn) ACTIVE-DESCR: (taking-turn giving-turn gesturing addressing-me saying-goodbye concluding is-active looking-at-hands want-back-ch-feedb complete-utter)	**3**
NAME: dial-on-1 TYPE: PCL-State-Dec-Mod MSGS: dial-on NEXT-STATES: (dial-off) POS-CONDS: (saying-goodbye) NEG-CONDS: (dial-off) ACTIVE-DESCR: (saying-goodbye)	**4**
NAME: dial-off-1 TYPE: PCL-State-Dec-Mod MSGS: dial-off NEXT-STATES: (give-turn dial-on) POS-CONDS: (saying-my-name) NEG-CONDS: (dial-off) ACTIVE-DESCR: (addressing-me gesturing)	**5**
NAME: dial-off-2 TYPE: PCL-State-Dec-Mod MSGS: dial-off NEXT-STATES: (give-turn dial-on) POS-CONDS: (addressing-me) NEG-CONDS: (dial-off) ACTIVE-DESCR: (addressing-me)	**6**

TABLE 9-3. State Decision Modules used in the Gandalf prototype. These are part of the Reactive Layer. The ACTIVE-DESCR slot contains the names of all Multimodal Descriptors that should be operative during the state. They are the descriptors that are essential to determine transitions to the next state, as specified in the NEXT-STATES slot. When a state node gets posted its MSGS value is put on the Functional Sketchboard.

NAME: exe-DKB-act TYPE: PCL-Dec-Mod EL: 2000 MSGS: (post-DKB-act) POS-CONDS: (dial-on TKB-act-avail take-turn) NEG-CONDS: nil POS-RESTR-CONDS: (CL-act-avail) NEG-RESTR-CONDS: nil	**1**
NAME: exe-TKB-act TYPE: PCL-Dec-Mod EL: 2000 MSGS: (post-TKB-act) POS-CONDS: (dial-on TKB-act-avail take-turn) NEG-CONDS: nil POS-RESTR-CONDS: (CL-act-avail) NEG-RESTR-CONDS: nil	**2**

TABLE 9-5. External Decision Modules in the *Process Control Layer* involved in deciding the timing of content delivery.

Internal Decision Modules

Five internal Decision Modules are used to deal with speech data (Table 9-4), four of which are PCL modules, one a reactive. Currently the only internal processes have to do with "cleaning"—such as deleting old speech from the parse buffer, etc.

External Decision Modules

Of the twenty-six external Decision Modules in Gandalf, fourteen belong to the PCL and have to do with content delivery and process control (Table 9-8 & Table 9-7 on page 140); twelve are dedicated to reactive behaviors (Table 9-6, page 139 & Table 9-8, page 141). Two of the PCL decision modules decide execution of content-related material (Table 9-8). These POST available acts—that is, acts that were successfully generated in response to a user's multimodal input—to the *BEHAVIOR-REQUESTS* queue (see "Behavior Requests" on page 118). Two different periodic decision modules were made in the Reactive Layer to produce eye blinks (Table 9-8, page 141); one is active during interaction, the other before and after the dialogue.

9.3 Spatial Data Handling

"...gesture is not simply a way to display meaning but an activity with distinctive temporal, spatial, and social properties that participants not only recognize but actively use in the organization of their interaction."

—Charles Goodwin (1986, p. 47)

The body of a user is represented geometrically. This is not a requirement of Ymir, and is obviously not the only way to represent a user's body in a multimodal system. Certain computations, however, are simple to deal with in geometric terms, such as gaze direction and deictic

(continued on page 141)

NAME: parse-speech TYPE: PCL-Int-Dec-Mod EL: 200 MSGS: (parse-speech) POS-CONDS: (giving-turn spch-data-avail) NEG-CONDS: nil POS-RESTR-CONDS: (give-turn) NEG-RESTR-CONDS: nil	**1**
NAME: rem-spch-addr-to-others TYPE: PCL-Int-Dec-Mod EL: 200 MSGS: (remove-speech-to-others) POS-CONDS: ((**FS-time-since** 'speaking 50)) NEG-CONDS: (addressing-me KB-exe-act) POS-RESTR-CONDS: (speaking) NEG-RESTR-CONDS: nil	**2**
NAME: remove-partial-parses TYPE: PCL-Int-Dec-Mod EL: 100 MSGS: (rem-part-parses) POS-CONDS: (KB-succ-parse KB-exe-act) NEG-CONDS: (addressing-me) POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (KB-exe-act)	**3**
NAME: report-no-words TYPE: PCL-Int-Dec-Mod EL: 50 MSGS: (compose-DKB-trouble-report 'no-words) POS-CONDS: ((**CB-time-since** 'rcv-spch 350) addressing-me giving-turn) NEG-CONDS: (CL-act-avail KB-succ-parse KB-exe-act speaking) POS-RESTR-CONDS: (give-turn) NEG-RESTR-CONDS: nil	**4**
NAME: clear-spch-buffers TYPE: RL-Int-Dec-Mod EL: 10000 MSGS: (clear-spch-buffs) POS-CONDS: (give-turn) NEG-CONDS: nil POS-RESTR-CONDS: (take-turn) NEG-RESTR-CONDS: nil	**5**

TABLE 9-4. Internal Decision Modules used in Gandalf's *Reactive* and *Process Control* layers. The function **CB-time-since** returns true if the time since the passed message (e.g. **rcv-spch**) was posted to the **C**ontent **B**lackboard exceeds the passed time (e.g. 350 centiseconds). EL = Expected Lifetime; FS = Functional Sketchboard.

NAME: show-take-turn **1** TYPE: RL-Ext-Dec-Mod EL: 500 MSGS: show-take-turn POS-CONDS: (take-turn) NEG-CONDS: nil POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (take-turn)	NAME: show-give-turn-1 **2** TYPE: RL-Ext-Dec-Mod EL: 200 MSGS: show-give-turn POS-CONDS: (give-turn) NEG-CONDS: nil POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (give-turn)
NAME: show-addr-me-1 **3** TYPE: RL-Ext-Dec-Mod EL: 20 MSGS: smile POS-CONDS: (TKB-exe-speech-act) NEG-CONDS: nil POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (turned-to-me)	NAME: show-give-turn-2 **4** TYPE: RL-Ext-Dec-Mod EL: 200 MSGS: show-give-turn POS-CONDS: (give-turn) NEG-CONDS: nil POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (give-turn)
NAME: show-addr-me-2 **5** TYPE: RL-Ext-Dec-Mod EL: 20 MSGS: eyebrow-greet POS-CONDS: (saying-my-name turned-to-me facing-me) NEG-CONDS: nil POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (turned-to-me)	NAME: show-listen **6** TYPE: RL-Ext-Dec-Mod EL: 20 MSGS: brows-in-pensive-shape POS-CONDS: (saying-my-name turned-to-me facing-me) NEG-CONDS: nil POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (turned-to-me)
NAME: initialize **7** TYPE: RL-Ext-Dec-Mod EL: 20 MSGS: face-neutral POS-CONDS: (dial-off) NEG-CONDS: nil POS-RESTR-CONDS: (dial-on) NEG-RESTR-CONDS: nil	NAME: show-not-addr-me **8** TYPE: RL-Spatial-Dec-Mod EL: 100 MSGS: (`turn-to` 'work-space) POS-CONDS: (dial-off) NEG-CONDS: (turned-to-me) POS-RESTR-CONDS: (taking-turn) NEG-RESTR-CONDS: nil
NAME: look-puzzled **9** TYPE: RL-Ext-Dec-Mod EL: 100 MSGS: look-puzzled POS-CONDS: (turned-to-me facing-me (`FS-time-since` 'facing-me 400)) NEG-CONDS: nil POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (turned-to-me)	NAME: look-aloof **10** TYPE: RL-Ext-Dec-Mod EL: 100 MSGS: look-aloof POS-CONDS: (turned-to-me facing-me (`FS-time-since` 'facing-me 800) dial-off) NEG-CONDS: (speaking KB-parsing) POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (turned-to-me)

TABLE 9-6. External Decision Modules used in Gandalf's *Reactive Layer*. Expected Lifetime (EL) values are in centiseconds. These modules control Gandalf's reactive behavior.

NAME: hesitate-1 **1** TYPE: PCL-Ext-Dec-Mod EL: 100 MSGS: hesitate POS-CONDS: (dial-on take-turn spch-data-avail (FS-time-since 'speaking 70)) NEG-CONDS: (CL-act-avail speaking) POS-RESTR-CONDS: (give-turn) NEG-RESTR-CONDS: nil	NAME: show-done-exe-1 **2** TYPE: PCL-Ext-Spatial-Dec-Mod EL: 2000 MSGS: (Look-At 'user) POS-CONDS: nil NEG-CONDS: (TKB-exe-world-act) POS-RESTR-CONDS: (TKB-exe-world-act) NEG-RESTR-CONDS: nil
NAME: turn-to-user **3** TYPE: PCL-Ext-Dec-Mod EL: 2000 MSGS: (Turn-To 'user) POS-CONDS: (TKB-exe-speech-act) NEG-CONDS: nil POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (TKB-exe-speech-act)	NAME: show-done-exe-2 **4** TYPE: PCL-Ext-Spatial-Dec-Mod EL: 2000 MSGS: (Turn-To 'user) POS-CONDS: nil NEG-CONDS: (TKB-exe-world-act) POS-RESTR-CONDS: (TKB-exe-world-act) NEG-RESTR-CONDS: nil
NAME: show-content-delivery **5** TYPE: PCL-Ext-Spatial-Dec-Mod EL: 2000 MSGS: (Look-At 'user) POS-CONDS: (DKB-exe-act looking-at-me) NEG-CONDS: nil POS-RESTR-CONDS: (take-turn) NEG-RESTR-CONDS: nil	NAME: show-know-addressing **6** TYPE: PCL-Ext-Spatial-Dec-Mod EL: 2000 MSGS: (Turn-To 'user) POS-CONDS: (TKB-exe-world-act looking-at-me facing-me speaking) NEG-CONDS: nil POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (TKB-exe-world-act)
NAME: pay-attention-to-act **7** TYPE: PCL-Ext-Spatial-Dec-Mod EL: 2000 MSGS: (Turn-To 'work-space) POS-CONDS: (TKB-exe-world-act) NEG-CONDS: nil POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (TKB-exe-world-act)	NAME: show-idle **8** TYPE: PCL-Ext-Dec-Mod EL: 2000 MSGS: restless POS-CONDS: nil NEG-CONDS: (facing-me) POS-RESTR-CONDS: (facing-me) NEG-RESTR-CONDS: nil
NAME: show-listening-1 **9** TYPE: PCL-Ext-Dec-Mod EL: 2000 MSGS: (Turn-To 'user) POS-CONDS: (speaking addressing-me) NEG-CONDS: nil POS-RESTR-CONDS: (take-turn) NEG-RESTR-CONDS: nil	NAME: look-at-domain **10** TYPE: PCL-Ext-Spatial-Dec-Mod EL: 2000 MSGS: (Turn-To 'work-space) POS-CONDS: (facing-domain take-turn) NEG-CONDS: (speaking) POS-RESTR-CONDS: (taking-turn) NEG-RESTR-CONDS: nil
NAME: look-relaxed **11** TYPE: PCL-Ext-Dec-Mod EL: 2000 MSGS: face-neutral POS-CONDS: (TKB-exe-world-act) NEG-CONDS: nil POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (TKB-exe-world-act)	NAME: show-listening-2 **12** TYPE: PCL-Ext-Spatial-Dec-Mod EL: 2000 MSGS: (Look-At 'user) POS-CONDS: (speaking addressing-me) NEG-CONDS: nil POS-RESTR-CONDS: (take-turn) NEG-RESTR-CONDS: nil

TABLE 9-7. External Decision Modules used in Gandalf's *Process Control Layer*. Expected Lifetime (EL) values are in centiseconds.

(continued from page 137)

gestures. All spatial features in the system are computed from data supplied by a body model server [Bers 1996, 1995a] which provides a geometric model of a person's upper body. We will now look at the representation of geometric elements derived from this model.

9.3.1 Spaces & Positional Elements

Space is divided into *volumes*, *planes* and *points*. The hope is that one can get away with this simplification without sacrificing too much of the agent's perceptuo-motor skills. Three spatial features are of crucial importance to conversants in multimodal dialogue:

1. *Work volume*
2. *Gesture volumes*
3. *Face planes*

These are important because they mark the boundary (albeit in a somewhat fuzzy manner) that events, objects or information can be located in. But knowing the size and shape of these is not enough; one needs to know the positions of these volumes and planes, and, in particular, objects within these. To point your eyes at any particular location in space, you need to be able to define that point relative to yourself. Coarse body movements can be generated on the basis of coarse knowledge—e.g. the boundaries of a volume. This can help you to point your head in the general direction (e.g. if you know someone is standing to your left, you turn your head to the left to get your eyes pointing in the right direction.) But to fixate on an object, its position needs to be defined more precisely. Accurate positional information allows one to apply the necessary force on the muscles of one's eyes to move them into the particular configuration that points them at the spot. In conversation, these positional data are essential to the information exchange:

TABLE 9-8. Periodic Decision Modules used in the Gandalf prototype. These belong to the *Reactive Layer*.

NAME: blink TYPE: RL-Per-Dec-Mod EL: 1000 MSGS: blink PERIOD: 300 POS-CONDS: (dial-on) NEG-CONDS: nil	**1**
NAME: blink-slowly TYPE: RL-Per-Dec-Mod EL: 1000 MSGS: blink-slowly PERIOD: 300 POS-CONDS: (dial-off) NEG-CONDS: nil	**2**

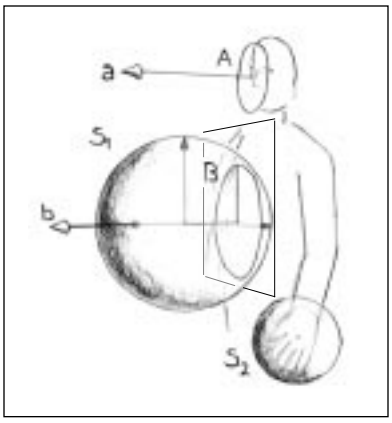


FIGURE 9-4. Geometric definitions of gesture space (S1), face space (A) and hand space (S2), along with normals showing direction of head (a) and trunk (b).

1. Position of *work volume*
2. Position of *gesture volumes*
3. Position of *face planes*
4. Position of *hands* (or *hand volumes*)

Volumes are mapped out as shown in Figure 9-4. Work space and faces are simply three-dimensional planes—a circular one for the face and a square one for the work space display (not shown). Position is given by the planes' centers.

The gesture space's primary role is to be an indicator of intentional gesturing, which mainly happens in the space right in front of the speaker's body [Rimé & Schiaratura 1991]. Self-adjusters [Ekman & Friesen 1969], which seldom have a communicative function, are automatically excluded by lifting the gesture space approximately 10 cm from the body of the user (plane B in Figure 9-4) because these happen close to the body. McNeill's research [1992] has indicated that the type of gesture and its place of articulation may be correlated. If this turns out to be the case, a finer division of gesture space would be useful for determining the function of manual gestures.

The user's hands are surrounded by a 20 cm diameter sphere, in order to give their position a larger margin, making it into a volume. This is especially useful when computing whether the user is looking at his or her hands. The following method is used for checking if a hand is inside gesture volume:

$$inside(S_g, h) = \begin{pmatrix} 0 if\ (C_{Sh} - C_{Sg} > R_{Sg})\ or\ (\text{acos}\ (|C_{sh} - C_B| \bullet b) > 90°) \\ 1 otherwise \end{pmatrix} \quad \{9.1\}$$

where h is a hand, S_g is gesture space (S_1 in Figure 9-4), C_{Sh} defines the center of hand space (S_2 in Figure 9-4), C_{Sg} defines the center of gesture space, R_{Sg} is the radius of gesture space, C_B is the center of plane B, and • is dot product (refer to Figure 9-4).

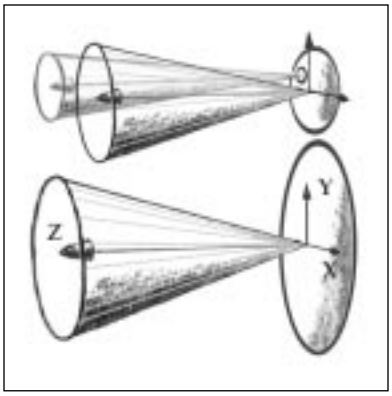


FIGURE 9-5. Directional elements of the user's upper body are treated as cones (gaze not shown) whose overlap on objects in the environment (such as the agent's face and the workspace screen) constitutes "facing" the objects in question.

9.3.2 Directional Elements

When is a person "facing" someone or something? When is a person "turned" in a given direction? These are questions that need to be answered by any multimodal agent, because their answers are required for successful participation in a face-to-face conversation. There are undoubtedly numerous ways to answer it. Here, as before, we use geometric definitions. The directional features extracted in Gandalf are:

1. Direction of *gaze*.
2. Direction of *head*.
3. Direction of *trunk*.

Absolute direction is computed after the position and relative orientation of relevant body parts is known. Figure 9-4 shows the two normals used for head and trunk (gaze direction not shown). The head, gaze and trunk normals are further modified by making them cones (Figure 9-6) so that their interception with other spaces, such as the agent's face space, is broader: interception happens if the inside of the cone overlaps the area or point in question. The angle of the cones is graded such that gaze has the narrowest (a 20° cone), then the head (35°), and lastly the trunk (40°).

The user is facing point p if point p falls within the boundary of head cone. More generally, to find if plane A is facing a point p in space, the following method is used:

$$facing(A, p, T_A) = \begin{pmatrix} 1 \, if \, \text{acos}((p - C_A) \bullet n_A) < T_A \\ 0 \, otherwise \end{pmatrix} \quad \{9.2\}$$

where C_A defines the center of plane A, n_A is plane A's normal and T_A is the angular threshold of plane A's cone (Figure 9-6).

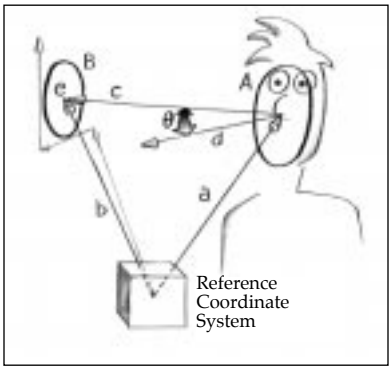


FIGURE 9-6. Geometry defining the "facing" function (see Equation 9.2). Center of Face Plane A is defined by vector a, d is plane A's normal. Center of plane B is defined by vector b. θ defines plane A's cone. By comparing the angle betwen vectors d and c to a threshold, one can determine whether the person on the right is "facing" plane B, which couldfor example represent the agent's face.

9.4 Prosody

Methods have been suggested for automatic analysis of prosody [Wang & Hirschberg 1992], but very few have tried to do analysis in real-time. I use a real-time intonation analyzer that I designed, that detects the following boolean, time-stamped events:

1. Speech on/off.
2. Intonation going up.
3. Intonation going down.

Notice that although detecting a feature like "speech on/off" may seem trivial, this is only true if we use a dedicated microphone which is unlikely to pick up anything besides the dialogue participant's speech. Using a signal processing approach with artificial ears, this may be a significantly more difficult task.

The intonation analysis is performed using a windowing technique, where a window is 300 ms. Each new window starts where the last one ended. The slope of the intonational contour is tracked in each window and checked against a threshold. If over the threshold and different from last window, a time stamped status report is given about intonational direction.

To analyze direction, a time-dependent algorithm is used whose output depends on the prior output, one window back in time. The robustness

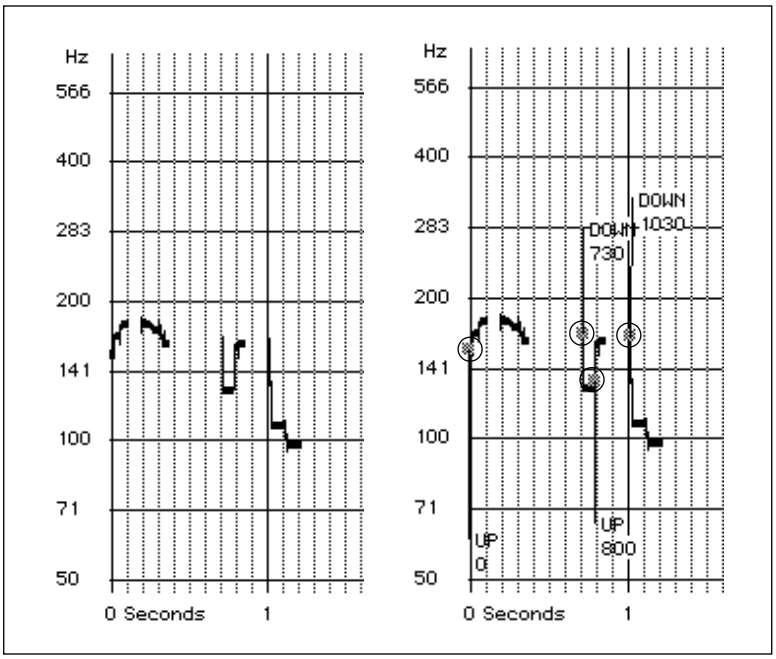


FIGURE 9-7. Example of intonation for the utterance "Take me to Jupiter" plotted to a logrithmic frequency scale. On the right we see the result of the real-time intonation analysis. Segmentation of pitch direction is marked with vertical bars, giving timing (in msec) and direction of the audio stream. Slanting lines from the grey dots on the graph to the markers (e.g. "DOWN / 730") indicate time delay. Where no slant is seen, the analysis took less than 10 ms to compute. The last "DOWN" marked may have taken about 10-15 ms.

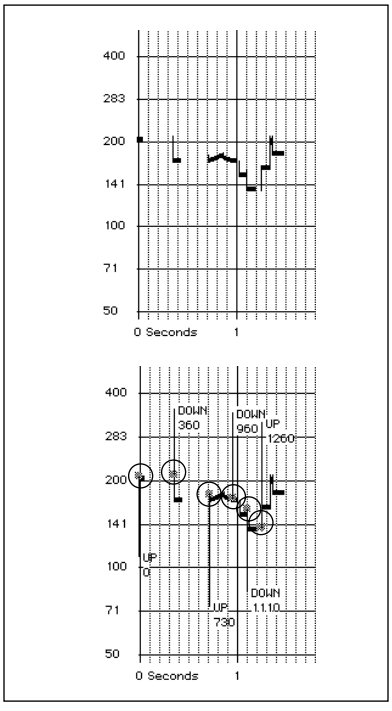


FIGURE 9-9. Results of analysis of the question "What planet is that?".

of intonation analysis is relatively high, considering that this is a real-time system. (We estimate that in normal interaction, about every tenth utterance is impossible to analyze. This reliability is high enough for an interactive system—keep in mind that the agent can always ask the user a question when the data doesn't make "sense".) Other modes of course help correct for occasional failures in the analysis process.

By running this intonation system on a dedicated machine, real-time response can be assumed. To give the reader a feel for the performance of the algorithm, several examples are given (Figures 9-7, 9-8 & 9-9).

9.4.1 Future Additions

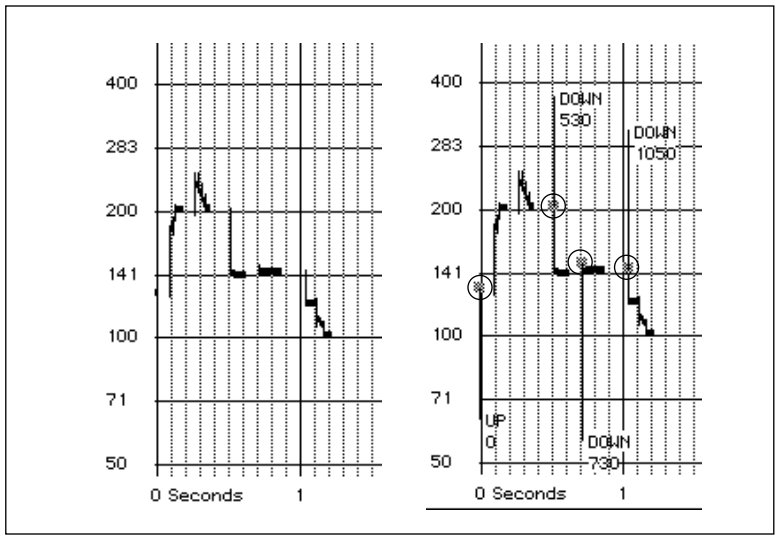
Future work includes using temporal multimodal descriptors to extract information regarding the intonation pattern over a full utterance, for determining whether an utterance could be a filler (relatively short and flat pitch pattern), question (final rise) or command (final fall) [Pierre-

```
(defun compute-direction (datapoints)
 (let* ((beginning (first-half datapoints))
        (ending (last-half datapoints))
        (pitch-a (compute-average-pitch beginning))
        (pitch-b (compute-average-pitch ending)))
        (slope 0)
        (snap-anlge   0) ;degrees
        (up-thresh   20) ;degrees
        (down-tresh -20) ;degrees
        (duration 200))  ;ms
   (setf slope (compute-angle pitch-a pitch-b duration)))
   (cond  ((eq *last-direction* 'UP)
             (setf snap-angle (expt slope 1.5)))
          ((eq *last-direction* 'DOWN)
             (setf snap-angle (expt slope 2.5)))
          (t (setf snap-angle (expt slope 0.8)))))
   (cond  ((> snap-angle up-thresh)
             (setf *last-direction* 'UP))
          ((< snap-angle down-thresh)
             (setf *last-direction* 'DOWN))
          (t (setf *last-direction* 'FLAT)))))
```

ALGORITHM 9-9. This function sets the global variable *last-direction* to the current direction. If *last-direction* is either UP or DOWN, it is trasmitted, along with frequency and a time-stamp, to Ymir.

humbert & Hirschberg 1990]. More extensive analysis would of course be preferable, including volume of speech and absolute high/low point recognition [Pierrehumbert & Hirschberg 1990] instead of relative. With such data one could more easily find pitch accents that could be synchronized with the words provided by the speech recognizer, and thus distinguish between the theme and rheme of an utterance [Clark &

FIGURE 9-8. Another example of pitch contour for the utterance "Take me to Jupiter". On the right is the result of the real-time analysis of the direction of intonation.



Brennan 1990, Prevost 1996]. This is a more difficult problem, but one that should be solvable in the near future. We are testing a method that adds absolute pitch (Hz) to the UP/DOWN markers and are hoping that this representation of the intonational contour will make it relatively straight forward to find pitch accents.

One problem with absolute scales is that pitch range varies between individuals and even highs and lows may vary highly between utterance for the same individual. A third source of difficulty is obtaining accuracy in the pitchtracking itself. These issues will have to be addressed in future work.

9.5 Topic & Dialogue Knowledge Bases

9.5.1 Speech Recognition

As mentioned in the beginning of this chapter, the current prototype uses a beta version of the HARK system from BBN [1993]. Time stamping is a necessity in any real-time system where different pieces of the same puzzle are analyzed separately, to piece them back together again. HARK provides time stamps (using additional in-house developed post-processing) for each word. Incremental interpretation is crucial for not disrupting the natural flow of multimodal behavior. Ideally the speech recognition would be continuous, although in reality it doesn't happen until a significant pause (250 msec[1]) is found in the audio stream.

As discussed in the section on interpretive knowledge, and as supported by user testing (page 155), more robust results would be achieved by using multiple speech recognizers: one that spots keywords, one that spots transitional cues (e.g. *cue phrases* [Grosz & Sidner 1986, Cahn 1992]), one that has grammar and vocabulary for a particular topic and one that recognizes fillers. Transitional cues would be used to weight various parts of the topic grammar (or turn them on or off) according to what the system thought the current topic is. Ymir is very well suited for this kind of "distributed" speech recognition scheme: We are currently designing Internal Decision Modules for this task, interfacing with the built-in features of HARK.

1. This number is user-definable.

9.5.2 Natural Language Parsing & Interpretation

The natural language parser/multimodal output generator used in Gandalf is based on a continuous speech recognition model (in spite of the current speech recognition being a "batch processing" recognizer), i.e. if fed with one word at a time, it will try to fit the words together even before all of them have arrived. A selected sample of utterances it recognizes are shown in Figure 9-10. It uses semantic templates to parse the utterances. This works quite well for a small domain, and can be extended to handle multimodal interpretation. The parser is indifferent to word order (i.e. it makes no distinction between "That planet—tell me about it" and "Tell me about that planet"). While it may not be the most sophisticated way to parse natural language, there is little reason at the current state of technology to apply complex grammar rules when parsing in a real-time, face-to-face multimodal system, since word order and actions in such an interaction are much more loosely connected by grammar than for example words written on a page.

Show me X
Take me to X
Tell me about X
What else?
Tell me more
Is that [deictic gesture] X?
What planet is that [deictic gest]?
Zoom [in | out]
Tilt it this way [wrist gesture].
Stop [the] animation
Start [the] animation
Tell me about the moon[s]
Hello [Gandalf]
Goodbye [Gandalf]
Gandalf?

FIGURE 9-10. A sample of the utterances Gandalf recognizes. X stands for the name of any planet in the solar system, and the sun. Brackets show options.

Outline of Parsing Process

For each word received, the parser tries to fit it into a semantic template. If a template has already been activated that can accommodate the word, it puts it there, otherwise it finds all templates that could possibly fit the word and marks them as active. A template gets a score depending how many of its slots have been filled. Whenever this score is higher than the template's pre-set threshold, the knowledge base tries to produce a response based on the content in the template. If it does, it posts this fact to the Content Blackboard (see section 8.3.2, page 98). If it doesn't, time passes and no message is posted to the Content Blackboard. This condition is monitored by decision modules in the PCL, which will then initiate a "problem report" generated by the Dialogue Knowledge Base. The DKB will look at the messages posted by the TKB and generate an appropriate response, e.g. "I'm sorry, I didn't get that". For partially filled templates, the DKB could generate more intelligent responses such as "Which planet did you want to go to?" or "Please repeat the name of the planet". The response itself is kept in a list in the knowledge base, and executed when Decision Modules in the PCL decide to.

When the agent takes the turn, a module in the PCL fetches the response and sends it to the {1} virtual world if it is an action (TKB-world-act) or to the {2} Action Scheduler if it is a speech act (TKB-speech-act or DKB-speech-act). If the TKB-world-act contains complimentary facial expressions, speech or manual gesture, the action is split to each destination.

9.5.3 Multimodal Parsing & Interpretation

The above scheme was intended to be extended to multimodal parsing. Ongoing work involves extending it by adding multimodal *meta*-templates with slots for each mode. A deictic meta-template could for example contain slots for sentences involving use of deictic phrases ("That one", "...these two") and a slot for deictic gestures. Extensive rules about how to fill the template's slots can make this scheme quite flexible. While for example Sparrell's VECIG system [Sparrell 1993] was completely driven by speech, my approach combines top-down with bottom-up processing in each mode, as well as a across modes: The idea is that maximum information be gleaned from a bottom-up approach (e.g. morphology, combinations of mode-dependent information) and that this will guide top-down hypotheses regarding the content of the multimodal acts. An intonation pattern typical for questions can for example activate a speech-parsing template for questions; a gesture that looks very much like an iconic gesture could activate a template for spatial information. Likewise, a speech template could be marked for probability of co-verbal gesture, thus initiating gesture analysis, even if the bottom-up approach fails. This would of course not be useful unless the speech recognition is incremental.

A nice feature of this approach is that a template activated by events in one mode can indicate what kinds of multimodal actions could be expected. Another big advantage is that interpretation is not solely driven by speech content: over the course of a multimodal action, any mode can contribute to the hypothesis-building about its content and meaning. Information posted to the Functional Sketchboard will obviously play a large part in this extension.

9.5.4 Topic: The Solar System

The system's knowledge is based on a lexico-spatial database of planets. Each planet, represented as a CLOS object, is defined as a point in the virtual world, and is accessed through its unique name. Four functions provide the Gandalf with the ability to generate responses to action-related queries, {1} `Go-To`, {2} `Zoom` {3} `Freeze-Anim`, and {4} `Tilt`. The first takes a planet object as an argument, the second takes a direction as an argument (zoom "in" or "out"). `Freeze-Anim` takes a boolean state and stops or starts the clock of the world animation engine. `Tilt` takes a direction, given with a wrist gesture. A number of utterances lead to the same use of these functions, for example "Show me Jupiter" and "Let's go to Jupiter" both result in the `Go-To` function being called with the "Jupiter" object.

9.6 Action Scheduler

9.6.1 Behaviors

Gandalf's Behavior Lexicon contains 83 behaviors, specified at various levels of detail. Below is a full listing of the specification of the lexicon. Notice that only a subset of this lexicon is used so far by the decision modules. A number of these will have to wait for future extensions of Gandalf.

9.6.2 Motor System

Gandalf's motor system, the ToonFace Animator (Appendix A1), runs on a Silicon Graphics Indigo2. Currently the loop time for a complete redraw of the face and hand is 150 ms.

9.6.3 Behavior Lexicon

At the end of the chapter (page 153) is the list used to generate CLOS behavior objects for Gandalf, alias the *Behavior Lexicon*, by calling the function `Make-Behaviors` with the list `*behavior-lexicon*`. The list, albeit minimal, is sufficient for rudimentary dialogue skills.

Behaviors come into two main groups: {1} *Morphological* and {2} *Functional*. Morphological behaviors are named after the way they *look*, for example, the behavior `brows-in-u-shape` specifies a shape for the brows to take. Nothing is said about what circumstances such a behavior should or could be used in, nor what possible meanings such a behavior could carry. On the other hand, the behavior `show-taking-turn` specifies a dialogue *function*. There are many ways for showing that you are going to say something, one being opening the mouth slightly, another is glancing away briefly [Kleinke 1986, Goodwin 1981, Duncan 1972]. Within these classes, various sub-classes of facial and manual gesture have been implemented.

9.7 Examples of System Performance

Now that we have shown how a complete character is built in Ymir, let's look at some run-time data from this prototype to get a better idea of how it performs. All modules shown in these graphs can be found in the tables in this chapter.

Figure 9-11 shows the internal events of Gandalf in its interaction with the actor Alan Alda during a visit from the television series "Scientific

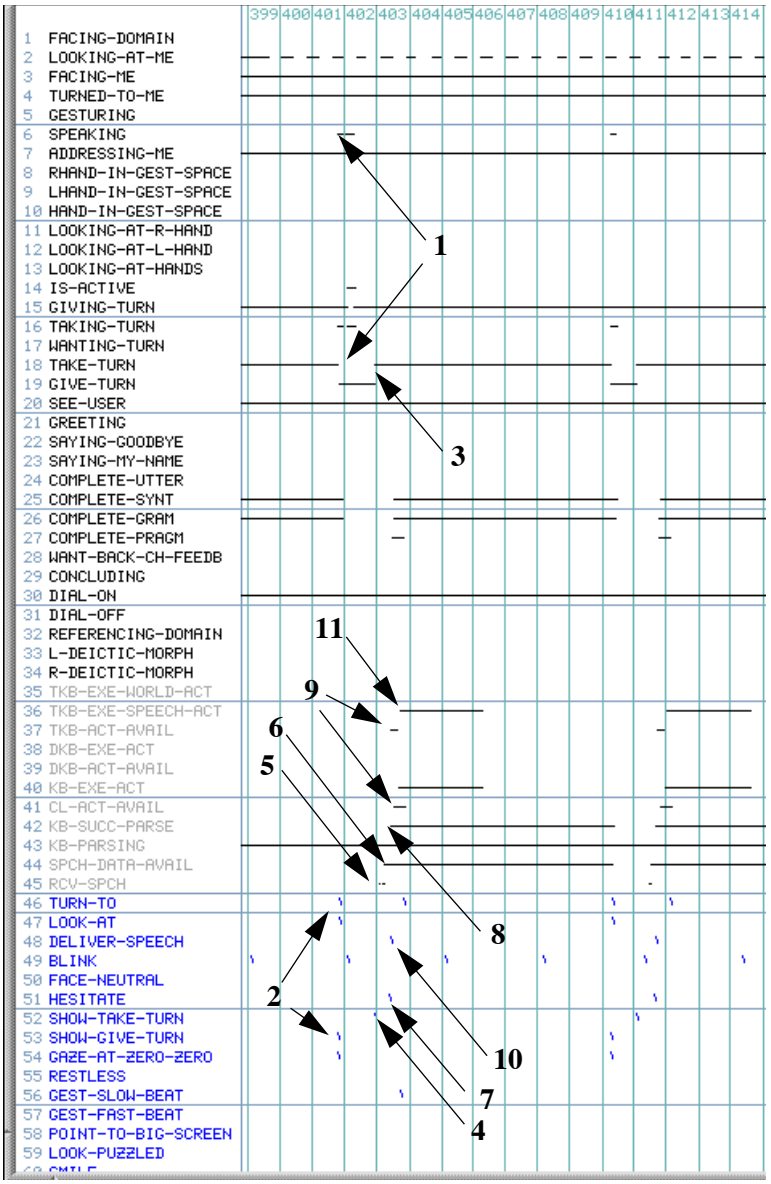


FIGURE 9-11. Graph showing internal states of Gandalf during interaction over a 16 second interval (each vertical line marks a second). When the person starts speaking Gandalf gives turn [1], turns to the user and shows that he is giving turn [2]. When the person falls silent Gandalf takes the turn [3], and shows that it is doing so [4]. At about the same time something is recieved from the speech recognizer [5] and shortly thereafter they are reported as available words [6]. These are then parsed and reported as successfull parse [8] (meanwhile Gandalf hesitates because he has taken the turn but has nothing to say as of yet [7]). When a response is available [9], Gandalf delivers this [10] and this event is posted internally [11]. (The highly rythmic gaze pattern observed at the top of the graph indicates a bad eye calibration.) Notice that modules 1 through 34 all relate to the *user's* behavior.

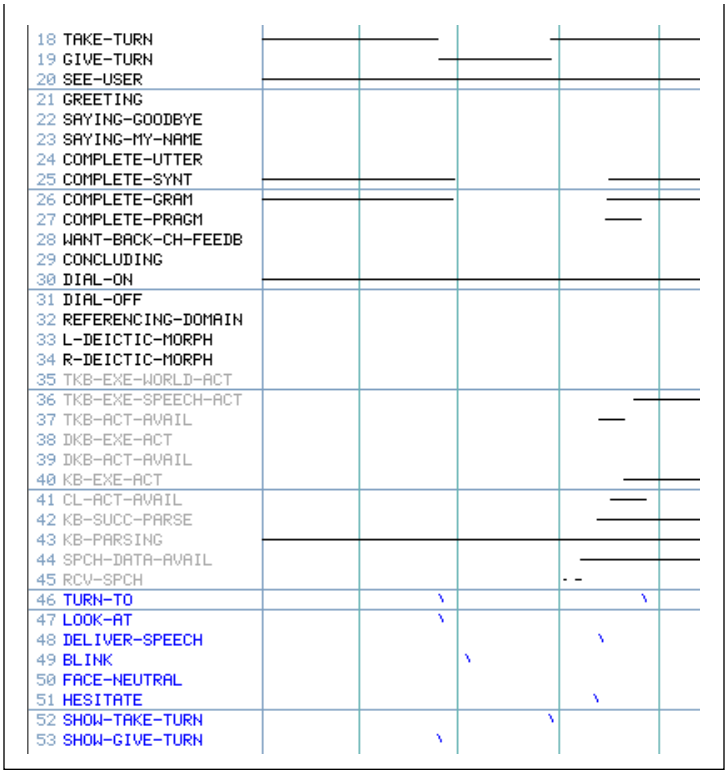


FIGURE 9-12. Interaction example in Figure 9-11 plotted at a high resolution. Each vertical line marks a second.

American Frontiers". Several features of the system are displayed in this example, as explained in the Figure text. The request made was "Tell me more about Mars".

Figure 9-13 shows yet another example of internal events during interaction with a user. This example spans 23 seconds, during which time the user makes three different requests, "Take me to the Sun", "Take me to Jupiter" and "What planet is that". Notice that although everything seems to have worked correctly internally in the first request, Gandalf does not execute the action (there is no line drawn for TKB-EXE-ACT after the request). This is because the Sun was already on the screen at the time of the request, and instead of executing the action, the DKB produces the utterance "This is Sun, dude".

Figure 10-11 on page 170 and Figure 10-13 on page 171 show another example of Gandalf's internal events when interacting with a human.

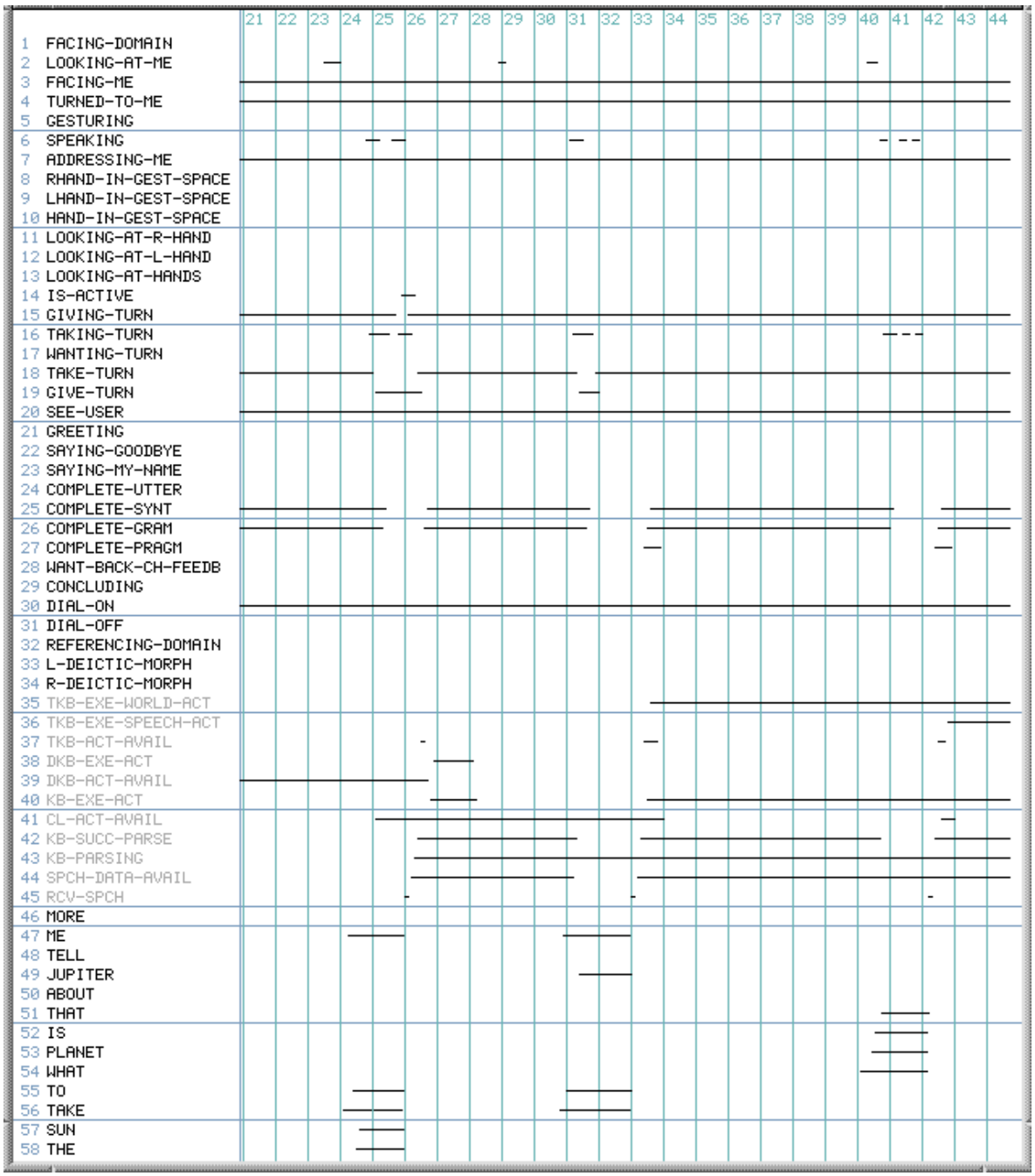


FIGURE 9-13. Example of internal events during an interaction betwen a user and Gandalf. Words spoken by the user are shown in lines 46 through 58 (beginning of line marks the time the word was uttered; end of line maks the time when Gandalf received the word.) See text for more details.

9.7.1 Behavior Lexicon Listing

```
(setf *behavior-lexicon*
  ;GENERAL LAYOUT: (<list-of-acts> (<first-act>(<first-act-element>)(<second-act-element>))
  ;                                (<second-act (<...>))
  ;                                )
  ;ACT TEMPLATE: (name class (((act-name-of-option-1 delay exec-time)(act-name delay exec-time) etc*)
  ;                           (etc*)))
  ;MOTORS:  (motor-name class delay exec-time pos/data)

 '(
  ; MORPHOLOGICAL DEFINITIONS[1]
   ;Features
    ;neutral
 (face-neutral act (((mouth-neutral 100 400)
                     (eyes-neutral 0 300)
                     (brows-neutral 0 500))))
 (brows-neutral act (((left-brow-neutral 0 400)(right-brow-neutral 0 400))))
 (left-brow-neutral mot-lev (((Bll 0 400 30)
                              (Blc 0 400 30)    ;Brow, left, central
                              (Blm 0 400 30)))) ;Brow, left, medial
 (right-brow-neutral mot-lev (((Brm 0 400 30)
                               (Brc 0 400 30)
                               (Brl 0 400 30))))
 (eyes-neutral act (((upper-lids-neutral 0 100)(lower-lids-neutral 0 100))))
 (upper-lids-neutral mot-lev (((Eru 0 100 75)(Elu 0 100 80))))
 (upper-lids-open-wide mot-lev (((Eru 0 100 89)(Elu 0 100 94))))
 (lids-neutral act (((upper-lids-neutral 0 300)(lower-lids-neutral 0 200))))
 (mouth-neutral mot-lev (((Mb  0 200 15) ;Mouth, bottom
                          (Mlv 0 200 60) ;Mouth, left, vertical
                          (Mlh 0 200 40)
                          (Mrv 0 200 60)
                          (Mrh 0 200 40))))
 (mouth-in-n-shape   mot-lev (((Mb 0 200 15)(Mlv 0 40)(Mrv 0 40)(Mlh 0 50)(Mrh 0 50))))
 (head-at-zero-zero  mot-lev (((Hh 0 800 0)(Hv 0 150 0))))
 (head-diag-up-left  mot-lev (((Hh 0 1000 20)(Hv 0 1000 20)))) ;for debugging
 (gaze-at-zero-zero  mot-lev (((Plv 0 50 0)(Plh 0 50 0)(Prv 0 50 0)(Prh 0 50 0))))

 ;actions
 (raise-brows         mot-lev (((Bll 0 400 90)(Blc 0 300 100)(Blm 0 400 90)
                                (Brl 0 400 90)(Brc 0 300 100)(Brm 0 400 90))))
 (lower-brows         mot-lev (((Bll 0 400 5)(Blc 0 400 5)(Blm 0 400 5)
                                (Brl 0 400 5)(Brc 0 400 5)(Brm 0 400 5))))
 (brows-in-v-shape    mot-lev (((Bll 0 400 90)(Blc 0 300 40)(Blm 0 400 10)
                                (Brl 0 400 90)(Brc 0 300 40)(Brm 0 400 10))))
 (brows-in-roof-shape mot-lev (((Bll 0 400 10)(Blc 0 300 50)(Blm 0 400 90)
                                (Brl 0 400 10)(Brc 0 300 50)(Brm 0 400 70))))
 (brows-in-n-shape    mot-lev (((Bll 0 400 50)(Blc 0 300 90)(Blm 0 400 50)
                                (Brl 0 400 50)(Brc 0 300 90)(Brm 0 400 50))))
 (brows-in-pensive-shape mot-lev (((Bll 0 400 95)(Blc 0 300 40)(Blm 0 400 40)
                                   (Brl 0 400 50)(Brc 0 300 10)(Brm 0 400 5))))
 (squint mot-lev (((Elu 0 300 60)(Eru 0 300 60)(Ell 0 300 20)(Erl 0 300 20))))
 (half-closed-eyes mot-lev (((Elu 0 500 50)(Eru 0 500 60))))
 (lower-lids-neutral mot-lev (((Erl 0 300 30)(Ell 0 300 50))))
 (lower-lids-up mot-lev (((Ell 0 300 0)(Erl 0 400 0))))
 (pull-l-mouth-corner mot-lev (((Mlh 0 500 90))))
 (quickly-glance-sideways-and-back act (((gaze-right 0 100)(gaze-at-zero-zero 100 100))))

 ;emblems
 (shake-head act (((turn-head-left      0  50)
                   (turn-head-right    50 100)
                   (turn-head-left    100 100)
                   (head-at-zero-zero 150 50))))
 (nod        mot-lev (((Hv 0 105 -15)(Hv 105 100 0))))
```

1. Morphological behaviors are behaviors that are named after the way they *look*. Contrast with behaviors that are named after what they *do*—i.e. functional definitions.

```
 (wink       mot-lev (((Elu 0 100 0)(Elu 300 100 90))))
 (say-ahh    mot-lev (((Sp 0 250 "[_<,110>aa<550,100>]"))))
 (gaze-up    mot-lev (((Plv 0 100 30)(Plh 0 100 40)(Prv 0 100 30)(Prh 0 100 40))))
 (gaze-away  mot-lev (((Plv 0 100 20)(Plh 0 100 -30)(Prv 0 100 20)(Prh 0 100 -30))))
 (gaze-right mot-lev (((Plv 0 100 20)(Plh 0 100 40)(Prv 0 100 20)(Prh 0 100 40))))

 ;emotional emblems
 (smile mot-lev (((Mlh 0 400 99)(Mrh 0 400 99)(Mlv 0 200 99)(Mrv 0 200 99))))
 (smile-a-little mot-lev (((Mlh 0 400 99)(Mrh 0 400 99)(Mlv 0 200 88)(Mrv 0 200 88))))
 (grin-broadly   mot-lev (((Mrh 0 1000 80)(Mrv 0 500 60)(Mlh 0 1000 80)(Mlv 0 500 60))))
 (grin-a-little  mot-lev (((Mrh 0 1000 75)(Mrv 0 500 55)(Mlh 0 200 40)(Mlv 0 200 50))))

 ;self adjustors
 (blink act (((close-eyes 0 50)(open-eyes 50 50))))
 (blink-slowly act (((close-eyes 0 300)(open-eyes 300 200))))

 ; FUNCTIONAL DEFINITIONS
  ;Back channel feedback / turn control
 (say-aha   mot-lev (((Sp 0 250 "[_<,110>aahxaa<250,130>]"))))
 (look-pensive act (((gaze-away 0 100)(pull-l-mouth-corner 300 500))))
 (look-aloof   act (((gaze-away 0 50)(turn-head-left 200 1000)(raise-brows 800 800))))
 (look-puzzled act (((squint 200 200)(brows-in-roof-shape 0 400))))
 (look-drowsy  act (((half-closed-eyes 0 800)(lower-lids-neutral 0 400))))
 (show-give-turn act (((face-neutral    0  200)(gaze-at-zero-zero 0 100)
                       (head-at-zero-zero 0 600)(raise-brows     0  200))))
 (show-take-turn act (((open-mouth-wide 0  100)
                       (quickly-glance-sideways-and-back 0 300)(blink-slowly 300 400))  ;option 1
                     ((eyebrow-greet   0  500)(quickly-glance-sideways-and-back 0 200)))) ;option 2
 (hesitate act (((say-ahh      0 400))   ;option 1
                ((gaze-up      0 200))   ;option 2
                ((look-pensive 0 600)))) ;option 3

; Notice that show-give-turn is controlled from the DKB, but should be composed completely here.
; Below action for illustrative purposes only - 2/19/96
;(show-give-turn act (((look-at user))))

 (show-listening act (((blink-slowly  0 500))))
 (back-ch-feedb-normal act (((say-aha 0 100)) ;option 1
                            ((nod     0 200)))) ;option 2

 ;other
 (happy act (((raise-brows 0 400)(brows-in-n-shape 400 200)(lower-lids-up 0 300)
              (open-eyes-wide 0 300)(smile 0 200))))
 (greet act (((eyebrow-greet 0 1500))))
 (eyebrow-greet act (((raise-brows 0 200)(upper-lids-open-wide 0 200)
                      (brows-neutral 900 200)(eyes-neutral 1000 300))))
 ;acknowledge
 (ack-normal    act (((say-ok-normal 0 250))))
 (say-ok-normal mot-lev (((Sp 0 250 "[_<,120>ow<,130>kehiy<250,95>]")))) ;speech - sent to DecTalk
 (say-ok-bored  mot-lev (((Sp 0 250 "[ow<,130>k<100,100>ehiy]"))))       ;speech - sent to DecTalk
 (say-all-right-normal mot-lev (((Sp 0 250 "[<,120>ow<,130>lraet<250,95>]"))))

 ;other
 (close-eyes         mot-lev (((Eru 0 300 10)(Elu 0 300 10))))
 (open-eyes          mot-lev (((Eru 0 300 75)(Elu 0 300 80))))
 (open-mouth-wide    mot-lev (((Mb  0 400 60))))
 (close-mouth-tight  mot-lev (((Mb  0 300 15))))
 (open-eyes-wide     mot-lev (((Elu 0 100 99)(Eru 0 100 95)(Ell 0 100 95)(Erl 0 100 95))))

 ; WILDCARD SPEECH
  ;Star is replaced by a value from the TKB [topic knowledge base] or the DKB [dialogue knowledge
base]
 (deliver-speech     mot-lev (((Sp 0 250 *))))



 ; SPATIAL BEHAVIORS
  ;gaze ;Star is replaced by a value from the spatial knowledge base
 (gaze-at-user spatial-mot-lev (((Plh 0 250 *)(Plv 0 250 *)(Prh 0 250 *)(Prv 0 250 *))))
 (look-at spatial-mot-lev (((Plh 0 250 *)(Plv 0 250 *)(Prh 0 250 *)(Prv 0 250 *))))
 (turn-to spatial-mot-lev (((Hh  0 1000 *)(Hv  0 800 *))))
```

```
 ;head (mostly for debugging)
(turn-head-toward spatial-mot-lev (((Hh 0 900 *)(Hv 0 900 *))))
(turn-head-left  mot-lev (((Hh 0 150 -20))))
       (turn-head-right mot-lev (((Hh 0 150  20))))
(turn-head-up       mot-lev (((Hv 0 300  20))))
(turn-head-down     mot-lev (((Hv 0 300 -20))))
(head-at-zero-horiz mot-lev (((Hh 0 500   0))))
(head-at-zero-vert  mot-lev (((Hv 0 500   0))))
(turn-head-90-left  mot-lev (((Hh 0 150 -90))))
(turn-head-90-right mot-lev (((Hh 0 150  90))))
(turn-head-45-right mot-lev (((Hh 0 500  45))))
(turn-head-45-left  mot-lev (((Hh 0 500 -45))))

; MANUAL GESTURE
 ;morphological defs
(gest-rest              mot-lev   (((Gr 0 1000 0))))
(hand-raise-palm-fwd    mot-lev   (((Gw 0 600  0))))
(drum-with-fingers      mot-lev   (((Gd 0 600  0))))
(point-to-big-screen    mot-lev   (((Gp 0 1000 0))))

 ;functional defs
(gest-slow-beat         mot-lev   (((Gb 0 1500 0))))
(gest-fast-beat         mot-lev   (((Gb 0 600  0))))
(manual-hold-it-signal  act (((hand-raise-palm-fwd 0 900))))
(gest-greet      act (((hand-raise-palm-fwd 0 600))     ;option 1 - fast
                     ((hand-raise-palm-fwd 0 1000))))  ;option 2 - slower
(restless act (((drum-with-fingers 0 400))))
)) ;End all
```

Ymir / Gandalf: An Evaluation in Three Parts

10.

In this chapter we will be asking several questions. Having spent all this effort designing and implementing a computer controlled character, a key question is, *Does this system really behave like a human in a conversation?* The answer to that is "yes and no": On the one hand, people seem to give Gandalf a very favorable rating in comparison to humans. (For example, on a scale from 0 to 10 for language capabilities, humans getting a perfect 10, naïve computer users gave the humanoids a mean score of 7.59; SD=1.45.) On the other, the system's limitations are usually obvious to anyone after only 1-3 minutes of interaction. I will answer this question in three ways: {1} By comparing the performance of Ymir/Gandalf to the *Model Human Processor*—a predictive model of human perceptual, motor and cognitive performance [Card et al 1983, 1986], by {2} presenting the results of an experiment with 12 subjects interacting with 3 different characters, and by {3} careful observation from a designer's perspective. The question of concern in the last issue is how easy it is to construct an agent in Ymir. Once the foundation had been laid, it took only between 3 and 5 weeks[1] to construct a minimal set of perceptual, decision and behavior modules for Gandalf. We will look at this at the end of the chapter.

10.1 Evaluating Gandalf with the Model Human Processor

The Model Human Processor [Card et al., 1986; 1983] is a general engineering model of cognition, designed specifically to predict human performance and reaction times. By comparing Gandalf to this model, it

1. This is a rough estimation since some of Gandalf's modules were created in parallel with the development of Ymir.

gives us not only the ability to compare outward behavior of the character to the way humans behave, it allows us to compare the internal components of the Ymir/Gandalf system to something that has proven to be a good predictor of human performance.

The Model Human Processor can be described as a collection of *storages* and *processors* together with a set of *operating principles*. The perceptual system consists of sensors and short-term memories. The cognitive system receives symbolic information from the perceptual system and uses information in long-term memory to make decisions about how to respond. The motor system carries out the responses. Each of these subsystems has a hypothetical processor running at its own clock speed. A number of parameters characterize the behavior of each of these systems. For our situation, the relevant parameters would be:

- τ_c = cycle time of cognitive processor: 70 [30~100] msec
- τ_m = cycle time of motor processor: 70 [25~170] msec
- τ_p = cycle time of perceptual processor: 100 [50~200] msec
- fix = duration of a fixation-saccade pair: 230 [70~700] msec.

The brackets indicate the extremes for a given parameter (*typical value [lower bound*, *upper bound]*, respectively). Card et al. [1986] give a concise explanation of the origin of these numbers. They come from various psychological literature on reaction time and human performance. In this model, all activity in the cognitive system is a result of a discrete number of processing cycles.

For a task such as multimodal dialogue, a person would need to use all of the three subsystems of the MHP, the perceptual system, the cognitive system and the motor system. The sequence of actions in the MHP is *perceive*, *think*, *act*. Thus, cycle times (τ_p, τ_c, and τ_m) occurring within a single step should added; the maximum values for each step are then added together to get a final RT prediction.

10.1.1 Perceptual Processes

For reactive actions, such as perceiving whether the user has stopped speaking, the Model Human Processor would predict 100 [50~200] msec. We could take the lower end of this range to apply to the reactive perceptual processes, using Card et al.s' notational system, analysis of the current implementation of Gandalf shows this to be:

- Read visual (body) data: 8 [5 ~ 10] msec
- Read prosody (intonation) data: < 1 msec
- Read speech (word tokens) data: 2 [0 ~ 5] msec
- Update perceptual processes in Reactive Layer: 8 [5 ~ 10] msec.

These are serial events, giving us *18 [10 ~ 25] msec*. In addition, feeding the above processes with data is done over a fiber-optic net:

- Transmission delay for body data: 10 [3 ~ 30] msec
- Transmission delay for intonation data: 10 [3 ~ 30] msec
- Transmission delay for speech data: 10 [5 ~ 30] msec.

The transmission delays are parallel, giving us 10 [5 ~ 30] msec. This adds up to a total of 28 [13 ~ 55] msec. To answer the question about the speed of determining that a user has stopped speaking, we have to look at the delay from stimulus onset (when the user starts speaking) until the information is available to other processes. In Gandalf, this goes through the intonation tracking system, which has the additional delay:

- Speech on/off filtering: 10 [0 ~ 20] msec
- Silence inertia (constant): 50 [50 ~ 50] msec.

The silence inertia filters out pauses shorter than 50 msec. Taken together, we have *88 [63 ~ 125] msec* to detect that the speaker is silent. For other features, such as detecting that the hands are in gesture space or that the user is looking at the agent, we get a somewhat lower number of *28 [13 ~ 55] msec*. The MHP predicts 100 [50 ~ 200] msec.

The above numbers may seem pretty good, however, no time-dependent perceptual processes have been implemented, and the features detected are relatively simple (for example, no processes are devoted to determining the reliability of the data, which surely must be part of the 100 msec attributed to humans). The only perceptual process at the Process Control level is the deictic-gesture detector, which means that the numbers are likely to be different for a more capable agent.

10.1.2 Cognitive Processes

For an action such as deciding to act on a set of stimuli, the MHP would predict 70 [30~100] msec. Since the Decision Modules only look for logical combinations of conditions to compute their state, the performance for each module surpasses this prediction. Even taken as a group (running on a fast serial machine), the modules take < 0.0 [0.0 ~ 5.0] msec to execute one loop. Adding to that the delay to transmit the decision to the Action Scheduler, which we take to be 10 [3 ~ 30] msec, we get a total of 10 [3 ~ 35] msec. Since no time-dependent decision modules were implemented, it is difficult to predict how this would change for a fully-fledged decision system, although the numbers look like there is room for much more computationally intensive computations.

Adding to the above the time needed to parse incoming speech, update the knowledge bases, and monitor real-world acts, 3 [0 ~ 10] msec, giving a total of *13 [3 ~ 45] msec*.

10.1.3 Motor Processes

For an action such as moving the eyes, the Model Human Processor would predict 230 [70~700] msec. The gaze of Gandalf is updated only about every fourth second, falling short of the observed human performance. However, symbolic gaze events, such as turn signals, are still performed at the right transitions. Other motor responses should fall along the lines of 70 [25~170] msec, according to the MHP. Combining the behavior morphology selection and action scheduling, which is 30 [20 ~ 150] msec, and net transfer, 10 [3 ~ 130] msec, we get *40 [23 ~ 280] msec*. Another limiting factor in the motor system is the performance of the motor system itself, which is close to the upper limit of the Model Human Processor: ToonFace's (Appendix A1, page 203) smallest unit of execution in the current implementation is a constant of *150 msec*.[2]

10.1.4 Full-Loop Actions

The MPH predicts that a "closed-loop" motor task with visual feedback should be limited to 240 [105 ~ 470] msec [Card et al. 1983, p. 35]. Taking together the sequential events in Gandalf, we get

- Perceiving: 28 [13 ~ 55][3] msec
- Deciding: 13 [3 ~ 45] msec
- Acting: 190 [173 ~430] msec
- *TOTAL:* *231 [189 ~ 530] msec*.

These numbers are surprisingly close to those predicted by the MHP. However, a main issue in making a reactive conversant seems not to be reactivity in a closed-loop visuo-motor task, such as the above, but in making the right predictions about where, when and why events happen. In other words, top-down hypotheses must be at work in human-human dialogue to enable turn transitions with 0 msec overlaps in speech [c.f. Sacks et al. 1974], among other things. Another confounding factor is the slowness of the speech recognition, taking between 1.5 - 2 seconds to provide the content of the speech. Gandalf has to do an awful lot of

2. This number was determined empirically for a wide range of motor commands and scheduling loads, and is completely dependent on the speed of the computer responsible for the animation.
3. The lower values were selected since these represent a much larger set of events in Gandalf than the values for speech onset-offset detection.

	Gandalf / Ymir Alpha	MHP
Reactive Perception	28 [13 ~ 55] msec	100 [50 ~ 200] msec
Decision Making	13 [3 ~ 45] msec	70 [30~100] msec
Motor Actions	40 [23 ~ 280] msec	70 [25~170] msec
Full-Loop Actions	231 [189 ~ 530] msec	240 [105 ~ 470] msec

TABLE 10-1. Summary of comparison between Gandalf/Ymir Alpha and predictions of the Model Human Processor.

filling in with nonverbal behaviors to justify to the user this long pause before he responds. We will take a closer look at this in Section 10.2.

10.1.5 Conclusion

The intention here was to compare the current implementation of Gandalf to human performance, as modelled in the Model Human Processor. With the exception of speech recognition, Gandalf's performance stands fairly well up to human performance as predicted by the MPH, for the limited actions it was designed to do. If these performance numbers can be kept when adding increasingly sophisticated processes and modules to the system, one should expect a very reasonable model of a human conversant.

10.2 Human Subjects Experiment

The purpose of this experiment is to evaluate characters constructed in Ymir as they perform in real-time face-to-face interaction with a person, and to evaluate user attitudes toward humanoid interfaces as a function of the type of feedback given by the system. Three prototype humanoids were video-taped in their interaction with the subjects. Subjects' evaluation of the system was subsequently collected with a questionnaire. Subjects' speech patterns and behaviors were scored along the dimensions of relative number of user utterances (number of subject contributions[4] to the discourse over the number of a character's contributions) and relative number of subjects' hesitations and expressions of frustration (over the total number of their contributions).

4. A "contribution" is defined here as any speech utterance that is meant to elicit information, achieve an action, or be a response providing the information or achieving the action.

10.2.1 Background & Motivation

Research intended to answer questions about the various features of agent-oriented systems—that is, systems that employ an embodied, humanoid characters—has to date been hampered by the lack of real computer systems capable of sustaining and supporting spoken dialogue with a human user. To assess topics such as believability, trust, effectiveness of communication, users' likability of the interaction, as well as the question of whether to employ human-like figures to represent the system, these studies have turned to Wizard-of-Oz techniques [Maulsby et al. 1993, Hauptman 1989], mixed automation/Wizard-of-Oz [Thórisson 1992], typed natural language [Neal & Shappiro 1991, Wahlster 1991], iconic embodiments of various types [King & Ohya 1996, c.f. Maes 1994], or simply ignored the issue of embodiment [Sparrell 1994, Thórisson et al. 1992]. As a result, one cannot justifiably generalize the results of these studies and/or systems to future systems employing computer-controlled characters capable of real-time dialogue—tempting as it may have been to many researchers.

Prior efforts have often tried to assess the value of the *very idea of the agent metaphor* using a grab-bag of interaction methods. Because interaction method may be expected to interact strongly with users' perceptions of a system, such methodology is suboptimal at the best, unacceptable at worst. Various research has also intended to evaluate numerous *representations* of agents—humanoid, iconic, animal-like, etc.—by using collections of arbitrary behaviors, or simply ignoring behavior. Instead of trying to evaluate the inherent value of the humanoid agent metaphor, or the value of various visual and auditory representations for computer-imparted agency, I propose to turn these approaches on their heads, using a real computer-controlled humanoid to study communicative behaviors that *require* a humanoid representation.

Since attempts to evaluate full-duplex multimodal systems that employ artificial agents (fully or partially automated) have been virtually nonexistent, no data exists yet on important features of dialogue such as back channel feedback, mixed representations (e.g. spatial gestures + speech), and flexible turn taking, in the natural manner they combine to sustain and support face-to-face dialogue. Yet these are arguably the strongest reasons to employ an embodied, humanoid agent in a co-spatial, co-temporal communications system that uses spoken natural language.

In this experiment, we are interested in features that cannot be reproduced in any other way but by the use of embodied, social actors: spontaneous manual gesture and speech. If creating complex characters with multi-layered input analysis and output generation is to be justified, how else to justify it than with hard data from real interaction? "But why not

compare Gandalf to a condition using no embodiment?" you might ask. One cannot use the conventions of face-to-face dialogue (e.g. "If I'm looking at you while speaking, I'm probably speaking to you") if the conversants are not co-present. "But how about comparing Gandalf to a keyboard condition?" If the goal is to look at *natural speech* and/or *full-duplex multimodal* as interaction, one cannot introduce a keyboard or mouse into the system without compromising its naturalness. This is a different question—one that has been investigated by other researchers [e.g. Seu et al. 1991]—and will not be addressed here.

10.2.2 Goals

One claim that is often heard is that there is no need for multiple modes since the speech channel carries all the necessary data [Ochsman & Chapanis 1974]. If this is true, there is little reason to put in the effort to embody the system, all that is needed is speech synthesis and recognition. The main objective of this experiment is determining the importance of what we refer to here as *envelope* feedback to the effectiveness of dialogue. Envelope feedback includes back channel feedback [Yngve 1970], attentional feedback and other process-related feedback. Included in envelope feedback are reactive behaviors—behaviors that are very quick and people normally don't think about when performing during conversation. Examples include blinking, determining fixation points from moment to moment, saying "aha" at the right times, etc. We also group manual beat gestures in this category. The claim here is that these kinds of behaviors are the strongest argument for using an embodied agent in speech-based human-computer interaction, and, unless shown to somehow be important to dialogue, would be dismissed as yet another useless hog of processor cycle time.

Another kind of feedback that is often mentioned in relation to embodiment are *emotional emblems*. Emotional emblems are facial expressions that reference a particular emotion, without requiring the person showing the expression to feel that emotion at the moment of expression [Ekman 1979]. In the literature on anthropomorphism, emotional emblems have been held again and again to be a feature that an embodied agent-based interface could—and should—add to human-computer interaction [cf. Hasegawa et al. 1995, Nagao & Takeuchi 1994, Takeuchi & Nagao 1993, Britton 1991].

To investigate these questions, we compare three conditions. The basic condition contains content related feedback only. Content feedback is any uni- or multimodal actions that pertain to the topic of the dialogue, such as answers to questions or responses to requests. The second condition adds envelope feedback to the content responses, as defined below. The third condition combines emotional emblems with content responses.

Definitions of Behaviors

The following agent behaviors were used in the experiment:

I. Response to content:

1. Executing commands & answering questions.

II. Emotional emblems:

2. Confused expression when it doesn't understand an utterance.
3. Smiles when addressed by the user and when responding to a multimodal act.

III. Envelope feedback:

— Attentional:

4. Appropriate head turning and deictic gaze when listening to user and executing commands in the domain.

— Back channel:

5. Averting gaze and lifting eyebrows when taking turn.
6. Gazing back at person when giving turn.

— Status:

7. Eye blinks and tapping fingers to show that it is "alive".

— Content-related:

8. Manual gesture when providing verbal content.
9. Verbal acknowledgment when having understood a multimodal act.

We can take behavior 1 as given in any purposeful, communicative system: without appropriate response to content, there is little point to dialogue. But what about the latter two? In an anthropomorphic interface, which is more important: providing the system with the ability to provide (a) emotional emblems, or (b) feedback which is related to the process of the communication? We are claiming that the importance of anthropomorphism lies first and foremost in its power as a unifying concept for simplifying discourse. If this is true, feedback that relates directly to the process of the dialogue should be of utmost importance to both dialogue participants, while any other variables, such as emotional displays, should be secondary.

10.2.3 Experimental Design

Three conditions were tested: The Content Feedback (CONT) condition includes behavior 1 only, thus excluding emotional and envelope feedback. The Envelope Feedback (ENV) condition included all behaviors except II, excluding emotional feedback. The Emotional emblems (EMO) condition includes behaviors I & II, thus excluding envelope feedback. Examples of neutral, smiling and puzzled expressions for each character are given in Figure 10-1.

Hypotheses

Eight hypotheses were tested:

{H1} *No difference will be found for relative contributions from users between conditions CONT than in condition EMO.*

FIGURE 10-1. The three faces used in the experiment. Rows, from top to bottom: Gandalf, Roland, Bilbo. Columns, left to right: neutral expression, puzzlement, and smile.

{H2} *Relatively fewer subject contributions will be found in condition ENV than conditions CONT and EMO.*

{H3} *No difference in hesitations will be found between conditions CONT and EMO.*

{H4} *Relatively fewer hesitations will be found in condition ENV than in conditions CONT and EMO.*

{H5} *No difference in overlaps in speech will be found between conditions CONT and EMO.*

{H6} *Relatively fewer overlaps in speech will be found in condition ENV than in conditions CONT and EMO.*

{H7} *No difference will be found in subjects' rating of the agent between conditions CONT and EMO.*

{H8} *Subjects in condition ENV will rate the agent higher than those in condition CONT and EMO.*

Data for hypotheses 1 and 2 was collected by analyzing video tape recordings of the subjects. Relative number of contributions, as well as hesitations and frustration responses were scored according to pre-determined scoring schemes (Appendix A3, page 215). Data for hypotheses 3 and 4 was collected with questionnaires (Appendix A3.3, page 201).

Variables & Statistical Procedure

The dependent variables of concern are:

1. Relative number of contributions.
2. Relative number of hesitations.
3. Subjects' rating of agent on numerous scales.

The independent variables of concern are:

1. Amount of multimodal feedback (groups ENV, CONT and EMO).
2. Computer character.

The difference between conditions CONT, ENV and EMO on all dependent variables was tested with a repeated-measures MANOVA.

Procedure

Three different characters (face[5] + voice) are used to represent the computer in each condition, each of which was presented equally often in each position, and equally often for each of the conditions:

5. I would like to thank Hannesi Vilhjalmssyni and Roland Paul for designing the faces of Bilbo and Roland, respectively.

Condition		Character	
Content	(CONT)	Gandalf	(G)
Emotional	(EMO)	Bilbo	(B)
Envelope	(ENV)	Roland	(R)

and then varying the order of these conditions for each participant, creating the following presentation order for the 12 subjects, for the three conditions:

Subject	Order of Characters X / Y / Z	Order of Conditions 1st / 2nd / 3rd
1	G / B / R	ENV / CONT / EMO
2	B / R / G	CONT / EMO / ENV
3	R / G / B	EMO / ENV / CONT
4	R / G / B	ENV / CONT / EMO
5	G / B / R	CONT / EMO / ENV
6	B / R / G	EMO / ENV / CONT
7	B / R / G	ENV / CONT / EMO
8	R / G / B	CONT / EMO / ENV
9	G / B / R	EMO / ENV / CONT
10	G / B / R	ENV / CONT / EMO
11	R / G / B	CONT / EMO / ENV
12	B / R / G	EMO / ENV / CONT

The procedure for each subject was as follows:

1. Subject read the instructions for interacting with the characters (Appendix A3.2, page 215), and
2. read and signed a Declaration of Consent. Then they
3. answered a Background Questionnaire.
4. Subject put on the input devices (microphone, jacket & eye tracker) and went through a calibration procedure [Bers 1996].
5. Subject interacted for 2-4 minutes with character X (pilot).
6. Subject interacted with character X for 7-10 minutes and subsequently
7. answered Evaluation Questionnaire for character X (Appendix A3.3, page 216).
8. Subject interacted with character Y and subsequently
9. answered Evaluation Questionnaire for character Y.
10. Subject interacted with character Z and subsequently
11. answered Evaluation Questionnaire for character Z.

12. Subject answered Prior Beliefs Questionnaire (Appendix A3.4, page 219).
13. Subject read debriefing statement.

All three Evaluation Questionnaires are identical, except for the name of the character last interacted with.

Experimenters & Subjects

K.R. Th. acted as experimenter. A convenience sample of 12 volunteers between the ages of 22 and 37, both male and female, were tested. The Background Questionnaire confirmed that the subjects were naïve computer users, with no visual problems or other handicaps. All were native English speakers. Video tapes were scored independently by two scorers, in a double-blind design[6]. Scoring reliability for the variables obtained from the videos, overlaps, hesitations, and number of contributions, was > .95 [Pearson's correlation coefficient, $p < .001$, one-tailed].

Hypothesis	Means	t (pared)	Significance	Conf.
{H1} Contributions: EMO = CONT	EMO=1.52 CONT=1.33	-1.45	n.s. (two-tailed)	√
{H2} Contributions: ENV < CONT, EMO	ENV=1.23 C+E/2=1.42	2.49	$p < .016$ (one-tailed)	√
{H3} Hesitations: EMO = CONT	EMO=0.022 CONT=0.023	.07	n.s. (two-tailed)	√
{H4} Hesitations: ENV < CONT, EMO	ENV=1.0 C+E/2=0.02	-2.86	$p < .008$ (one-tailed)	no
{H5} Overlaps: EMO = CONT	EMO=0.036 CONT=0.015	-1.55	n.s. (two-tailed)	√
{H6} Overlaps: ENV < CONT, EMO	ENV=0.42 C+E/2=0.03	-2.05	$p < .033$ (one-tailed)	no
{H7} Agent Rating (Q1): EMO = CONT	EMO=40.67 CONT=44.83	1.86	n.s. (two-tailed)	√
{H8} Agent Rating (Q1): ENV > CONT, EMO	ENV=46.83 C+E/2=42.75	-3.99	$p < .003$ (one-tailed)	√
{H7} Helpfulness (Q3): EMO = CONT	EMO=3.23 CONT=3.02	-1.13	n.s. (two-tailed)	√
{H8} Helpfulness (Q3): ENV > CONT, EMO	ENV=3.85 C+E/2=3.13	-4.29	$p < .002$ (one-tailed)	√

TABLE 10-2. Results from paired t-tests for each of the hypotheses. Rating hypotheses were tested with two questions from the Evaluation Questionnaire (Appendix A3.3, page 216). DF = 11 for all t-values. EMO and CONT are pooled for all comparisons with ENV. Leftmost column lists which hypotheses were confirmed.

6. My thanks to Katrín Elvarsdóttir and Roland Paul for their precise scoring.

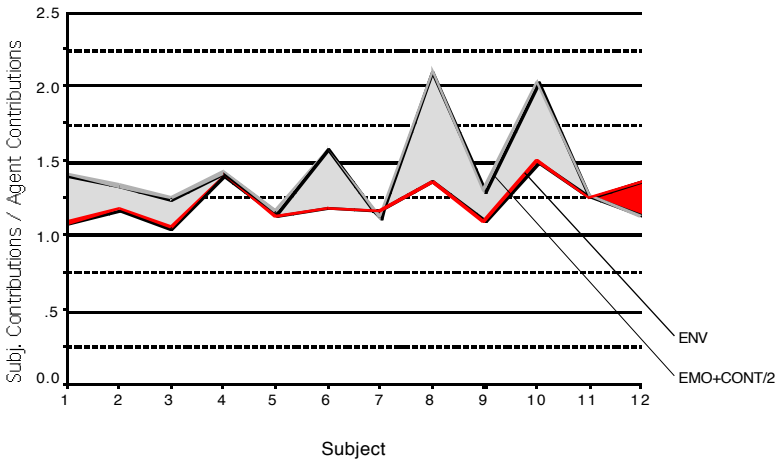
10.2.4 Results

All but two of the eight hypotheses were confirmed. The null hypotheses—that all numbers came from the same pool—was tested with a repeated-measures multiple analysis of variance (MANOVA) with all variables[7], and was rejected [F = 2.742, DF = 24, p < .02] (α is set at .05 for all hypotheses). Overall, the results supported the significance of envelope feedback over emotional emblems and content only feedback: Comparisons between individual means was done with paired t-tests, and are summarized in Table 10-2. No effects were found for order of character [F = 1.86, DF = 6, n.s.] or order of conditions [F = 1.85, DF = 6, n.s.], or interactions between these.

Relative Number of Contributions

Figure 10-3 shows the distribution of relative contributions for each condition in the experiment. This difference was significant at the .01 level [F = 2.74, DF = 10, p < .01, repeated-measures MANOVA]. Figure 10-2 shows a comparison between the number of contributions with CONT and EMO pooled ((CONT+EMO)/2).

FIGURE 10-2. Difference in relative number of contributions for all subjects between condition ENV (dark line) and (CONT+EMO)/2. This difference was significant. The amount of difference between the two conditions is filled with grey; the dark tail at the right indicates a reversal of the overall pattern for subject 12.



7. Set includes Evaluation Question 1, number of hesitations, relative number of contributions, and number of overlaps.

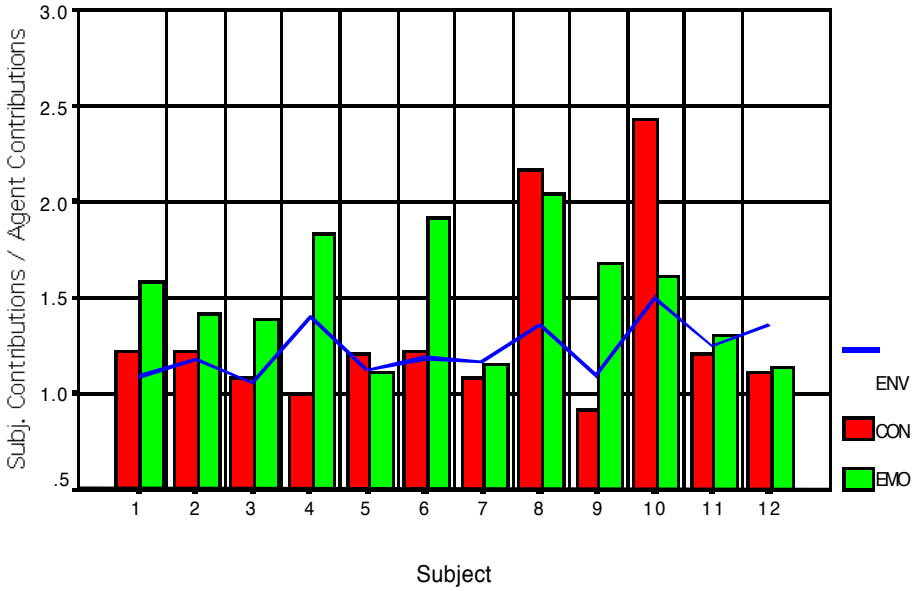


FIGURE 10-3. Relative contributions of subject (Csubject/Cagent) for each of the three conditions (left bar = CONT; right bar = EMO; line = ENV). The difference between the three conditions is significant at the .035 level [F = 2.74, DF = 10, $p < .035$, repeated measures ANOVA]. The difference between EMO and CONT is not significant, however, but the difference between ENV and (CONT+EMO)/2 is (see Table 10-2). In the ENV condition the ratio of user to agent contributions is 1.23.

Figure 10-4 and Figure 10-5 show the means for the two questions that were used to test the hypotheses for subjects' attitudes toward the agents.

Subject Evaluation of Conditions

The subjects' rating of the characters' language abilities are interesting: On a scale from 0 to 10, humans getting a perfect 10, subjects gave agents in the ENV condition a mean of 7.25 (SD=1.86) for language understanding and 7.92 (SD=1.83) for language use. These numbers are surprisingly high, and unless they simply indicate the user's satisfaction with the language part of the system—which, with naïve computer users could be the case—may point to a lack of grounding the lower end of the spectrum (i.e. stating that a dog should get a 1 might have resulted in different numbers).

The sub-questions in question 1 that were significantly different between ENV and the other two (pooled; CE=(CONT+EMO)/2) condi-

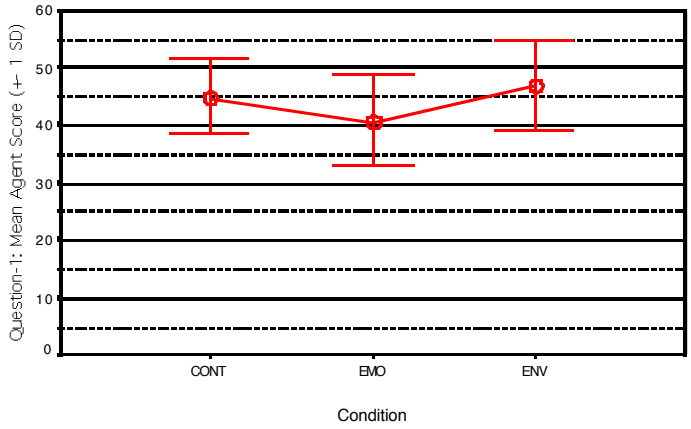


FIGURE 10-4. Means for each of the three conditions for the subjects' rating of the quality of interaction (question 1, Evaluation Questionnaire (page 216).

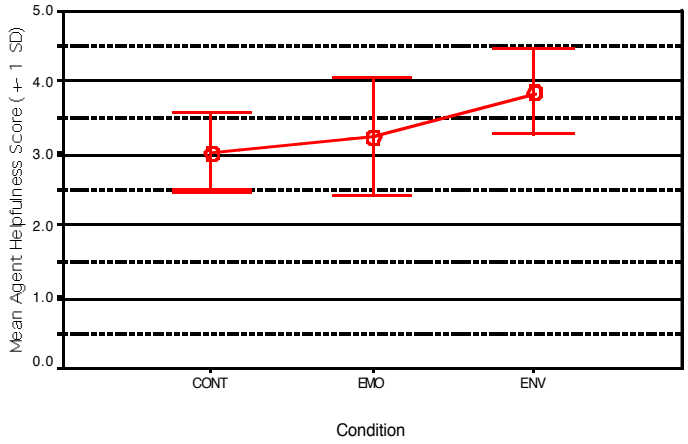


FIGURE 10-5. Mean score for "Agent Helpfulness" (question 3, Evaluation Questionnaire, page 216).

tions were: language understanding (means on a scale from 0 to 10: ENV=7.25, CE=6.67), language use (ENV=7.91, CE=6.83), smoothness of interaction (ENV=6.25, CE=5.41), smoothness of interaction compared to interacting with a dog (means on a scale from 1 to 5: ENV=4.08, CE=3.75), life-likeness compared to any computer character (ENV=3.83, CE=3.16). Comparison of the characters' lifelikeness to a fish in a fishbowl showed a ceiling effect (ENV=4.91, CE=4.83)

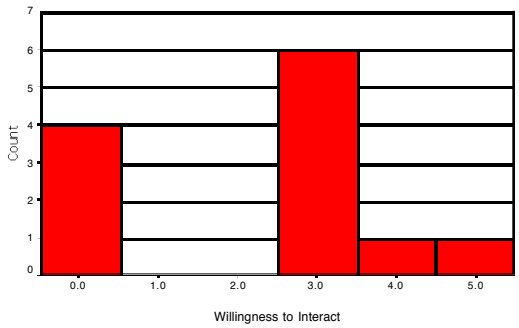


FIGURE 10-6. After interacting with the three characters, the majority of subjects reported that they were more willing to interact with a computer controlled character than before (0 = "No prior opinion", 1 ="Much less willing", 3 = "Equally willing", 5 = "Much more willing", Count = number of subjects).

and was not significant between the conditions, as were none of the other sub-questions. These are summarized in Table 10-3.

Descriptive Statistics

Subjects' reports on three additional variables are worth mentioning: Their answers to the question about increased or decreased willingness to interact with computer controlled characters based on this experience (Prior Beliefs Questionnaire, question 1-b) showed that none of the subjects were less willing, and about half of those with prior ideas about the issue were *more* willing than before (Figure 10-6). The subjects' changed perception of whether machines will ever become intelligent

TABLE 10-3. Items in Question 1, Evaluation Questionnaire for the ENV and CONT+EMO conditions that were *significantly different*. *Scale from 0 to 10; #scale from 1 to 5.

	ENV	CONT+EMO/2
Language Understanding*	7.25	6.67
Language Use*	7.91	6.83
Smoothness of Interaction#	6.25	5.41
Smoothness of Interaction Compared to Interacting with a Dog#	4.08	3.16
Life-likeness Compared to any Computer Character#	3.83	3.16

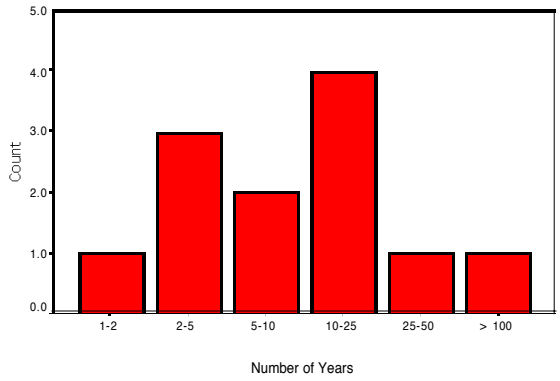


FIGURE 10-8. Distribution of answers for the question "How many years do you think it will take to create a computer controlled character that works perfectly?"

(Prior Beliefs Questionnaire, question 1-d) showed similar trend, with about one-third claiming *increased* confidence in intelligent machines (Figure 10-7).

Their estimation on how long it would take a research team to create a character that "works perfectly" (Prior Beliefs Questionnaire, question 3) showed very positive numbers, the mode being 10-25 years (Figure 10-8), meaning that most subjects, based on this experiment, expect to see characters that "work perfectly" well within their own life-times.

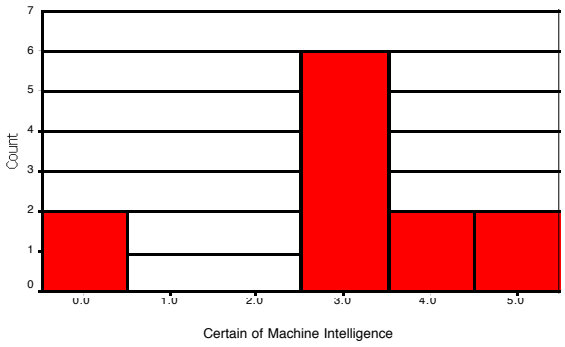


FIGURE 10-7. After interacting with the three characters, no subjects claimed the interaction to have changed their minds about whether machines will ever become intelligent (0 = "No prior opinion", 1 ="Much less certain", 3 = "Equally certain", 5 = "Much more certain", Count = number of subjects).

Answers to Open Questions

Subject responses to the open questions on the questionnaires were illuminating, and are summarized in Figure 10-9.

10.2.5 Discussion

All but two hypotheses where confirmed. This supports the general premise set forth in this experiment—that envelope feedback is important for language based, co-temporal, co-spatial interaction. The two hypotheses that were not confirmed showed a reverse pattern of what was expected, subjects tend to be more hesitant and frustrated in the ENV condition than the other two. A first attempt to answer why this could be might sound like this: Because the ENV condition provides more feedback about the state of the agent's processing, subjects tend to hesitate before speaking, simply because the agent displays behavior that allows them to hesitate in order to minimize overlapping speech. Unfortunately, if this were true, the overlaps in speech should have been the reverse of what they were, i.e. there should be fewer overlaps in the ENV condition than the other two. This was not the case. A more believable explanation to both these reversals is that since the agent's behavior in the ENV condition is more similar to human face-to-face interaction, subjects fell more easily back on a natural interaction style, a more complex one than they exhibited in the other two conditions. And since the characters' perception of the users' actions are limited, it couldn't respond to subtle features in the users' behavior, resulting in more overlaps and hesitations than in the other two conditions. If this is

FIGURE 10-9. Examples of responses to open questions in the Prior Beliefs Questionnaire (Appendix A3.4, page 219).

"Confirmed the idea that computer agents in the future will be able to aid, assist, educate and entertain in everyday life."

"I'm not sure I expect human faces."

"I was hoping that interaction would be faster, fewer pauses between inquiry and response."

"I'm not sure I expect human faces."

"Confirmed the idea that computer agents in the future will be able to aid, assist, educate and entertain in everyday life."

"I thought it would be much colder."

"Roland seemed totally stoned." (Subject in CONT condition)

the case, it points to a need for more sophisticated perceptual mechanisms to support natural, unhindered turn taking and information exchange. Needless to say, Ymir is designed to support such extensions.

Another source of observational evidence supports the above hypothesis. Although the biggest factor by far in determining how much nonverbal behavior the subjects exhibited was personal differences, subjects in the ENV condition tended to look more back and forth between the big screen and the character, tended to gesture more and seemed to be more drawn into the interaction in general. In general, participants tended to mimic the agents' behaviors: If the agent was rigid, they tended to stand still; if the agent was more animated, they tended to be animated. While useful information for future improvement, in the current prototype this could be expected to lead to a less predictable response pattern from the agents, resulting in more errors in judgement of the dialogue state, both of the subject and of the agent.

10.3 Ymir as a Foundation for Humanoid Agent research: Some Observations

One of the main goals in developing the Ymir architecture was to make it suitable as a platform for continued research in humanoid agents. The question then arises, how easy/hard is it to develop new modules, add functionality and modify existing structures? We will look at these questions in turn.

10.3.1 Developing New Modules with the Multimodal Recorder

To deal with the vexing complexity of developing new modules, a multimodal recorder facility was designed (Figure 10-10, Figure 10-13). The Multimodal Recorder allows an agent designer to graphically sketch out any of the internal module's states over time, for any particular period, and to compare events across layers and blackboards. An example of the entire repertoire of Functional Sketchboard[8] messages is shown in Figure 10-13. A real-time display of module states can be viewd in the Module Viewer window (Figure 10-14, page 180). This is very useful for initial testing of modules, to see if they respond correctly to events.

8. The Functional Sketchboard is a blackboard used in the Ymir architecture. It is discussed in Section 7.2.2, page 96.

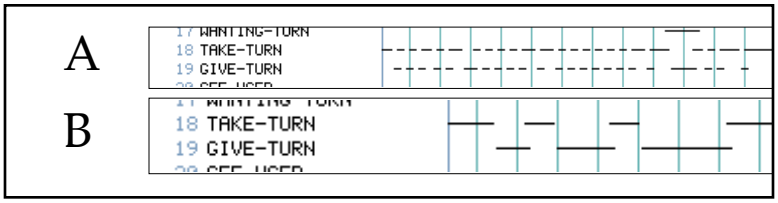
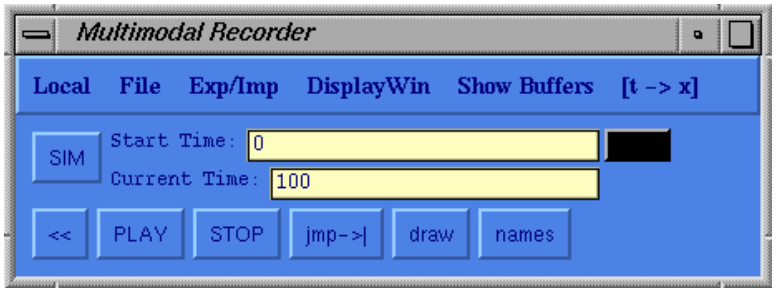


FIGURE 10-11. Using the visualization option of the multimodal recorder, an oscillation problem in the agent's decision pattern for giving and taking the turn (A) was fixed in a matter of hours (B).

Examples of Use

By using this recorder, a feedback problem in the turn taking rules was quickly resolved. The turn taking decision modules of Gandalf were showing a feedback problem causing oscillation in the State Decision Modules between the agent taking turn and giving turn (A, Figure 10-11). The same data from a human subject was fed back using the "simulation" option of the Recorder ("SIM" button in Figure 10-10). This option feeds back data recorded from the sensor and descriptor modules to the decision modules in real-time in the same manner they were originally generated by the human user's actions. The trigger rules for the turn taking modules were modified until a better[9] pattern was achieved (B, Figure 10-11). This modification took less than two hours using the Recorder; the problem had been causing inappropriate behaviors for days before.

FIGURE 10-10. The control panel of the Multimodal Recorder. Menus and buttons allow an agent designer to display the events of an interaction in a graphical format.



9. Notice that there is no "correct" pattern to be achieved here; simply a pattern that allows the agent to respond appropriately.

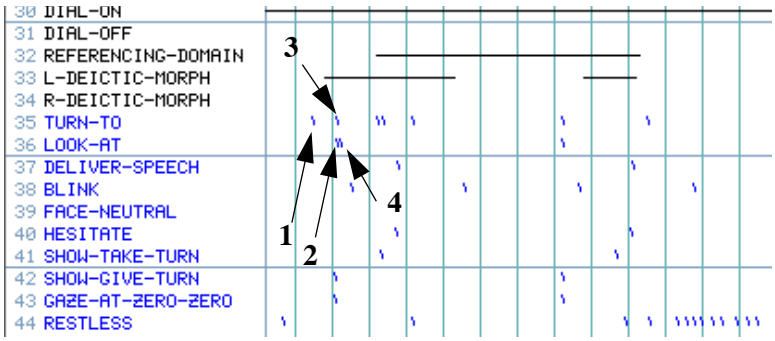


FIGURE 10-12. Example of the morphology (spatial) sensors for deictic gestures turning on. The multimodal descriptor REFERENCING-DOMAIN has more than just the two morphology sensors as input, so it stays on even though the sensor turns off. Gandalf first turns to the user [1], but upon the deictic detection it first looks in the direction of the gesture [2] and then turns in the same direction [3], and immediately readjusts the gaze to fall on the object [4].

Somewhat more difficult was the task of adding a PCL decision module that produces a "problem report" when speech is not recognized. The module (Table 9-4 on page 138, module 4) had to be highly constrained in its trigger and re-trigger conditions to work properly. It took about 5 days to get the right combination of conditions. Without the recorder, or some similar visualization tool, the construction of this module might have been all but impossible.

10.3.2 Adding Functionality: Deictic Gesture at the Input

We will now take an example of the task of adding the perception of, and ability to respond to, deictic manual gestures.

We begin by adding virtual sensors for the simple morphology of outstretched arm and hand above waistline. We come up with two sensors (Table 10-4, 1 & 2), one for each arm (this duplication of arms is a result of the sensing hardware used—other gadgets such as cameras might need a different breakup at this level). Because of the simulated parallel implementation of Ymir, little consideration needs to be paid to special scheduling of the various modules and processes when designing a humanoid, which leaves the implementer free to focus on other issues. Two procedures are called to compute the necessary values, one relating hand position to the trunk, the other comparing the elbow angle to a threshold. If we wanted to get detailed we could add an extended

index finger as yet another hint of deictic morphology. This data is used by a single Multimodal Descriptor (Table 10-4, 3) to determine if the user is addressing the work space.

Adding two new Decision Modules (Table 10-4, 4 & 5), the modules' Expected Lifetime can now be used to decide whether to respond to a deictic gesture, once detected, or to ignore it because it was detected to late.

Figure 10-12 shows part of the pattern created during a long deictic gesture and a shorter one. The speed of the whole loop, from perception to action, is the determinant of whether the character responds at all to the gesture.

10.3.3 Summary

Designing the first character in Ymir took somewhere between one and two months. This number may be expected to go down for second and third character, since much of the modules of the first one stay the same. The prototyping was made possible by using a multimodal recorder and visualization device which allowed the designer to graph internal events over time, at multiple resolutions. To achieve consistency in the internal workings of a character, a designer needs to be careful about stick-

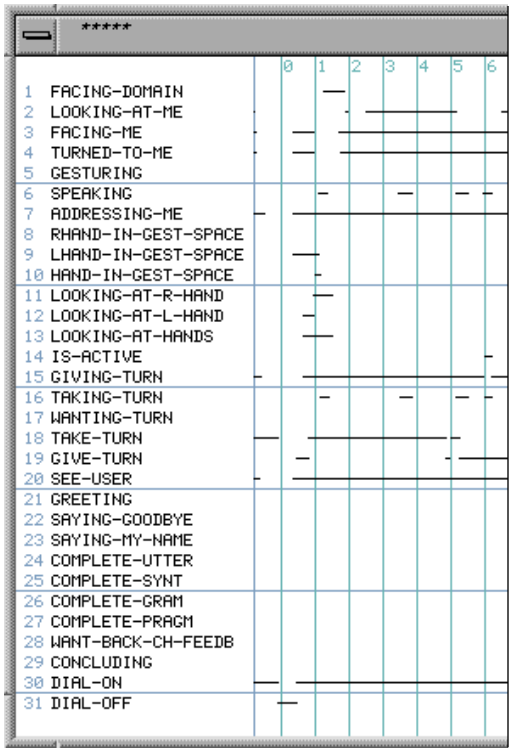


FIGURE 10-13. Example of the Multimodal Recorder display. It shows states of modules in the Reactive Layer over a period of 6 seconds. Lines mean that modules are true; where nothing is drawn they are false. Menu selections allow the user to switch between various sets of modules. In this example, the rules for Gandalf were being modified to produce the correct states. (See Tables 9-1, 9-3 & 9-3 in Chapter 9.)

NAME: l-deictic-morph TYPE: body-sensor-var-ref DATA-1: nil DATA-2: nil INDEX-1: (**Get-Body-Part** Left-Arm-Index) INDEX-2: (**Get-Trunk-Dir**) FUNC: **Deictic-Sketch**	**1**
NAME: r-deictic-morph TYPE: body-sensor-var-ref DATA-1: nil DATA-2: nil INDEX-1: (**Get-Body-Part** Right-Arm-Index) INDEX-2: (**Get-Trunk-Dir**) FUNC: **Deictic-Sketch**	**2**
NAME: referencing-domain POS-CONDS: (l-deictic-morph 0.7)(r-deictic-morph 0.8)(facing-domain 0.5)(speaking 0.5)(facing-me 0.4) NEG-CONDS: nil THRESH: 1.0	**3**
NAME: look-where-pointing TYPE: RL-Ext-Dec-Mod EL: 200 MSGS: (look-at 'big-screen) POS-CONDS: (referencing-domain) NEG-CONDS: nil POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (referencing-domain)	**4**
NAME: turn-to-where-pointing TYPE: RL-Ext-Dec-Mod EL: 200 MSGS: (turn-to 'big-screen) POS-CONDS: (referencing-domain)(**FS-time-since** 'referencing-domain 100) NEG-CONDS: nil POS-RESTR-CONDS: nil NEG-RESTR-CONDS: (referencing-domain)	**5**

TABLE 10-4. The function **Deictic-Sketch** uses morphology to find deictic gestures. It checks the angle of the elbow and the height of the hand above the waistline to determine if the posture of an arm might be doing a deictic gesture. Additional information such as posture of the hand could increase the accuracy of this virtual sensor.

ing to the architecture of Ymir. When this is done, however, Ymir provides a powerful foundation for designing and expanding the design, of communicative characters.

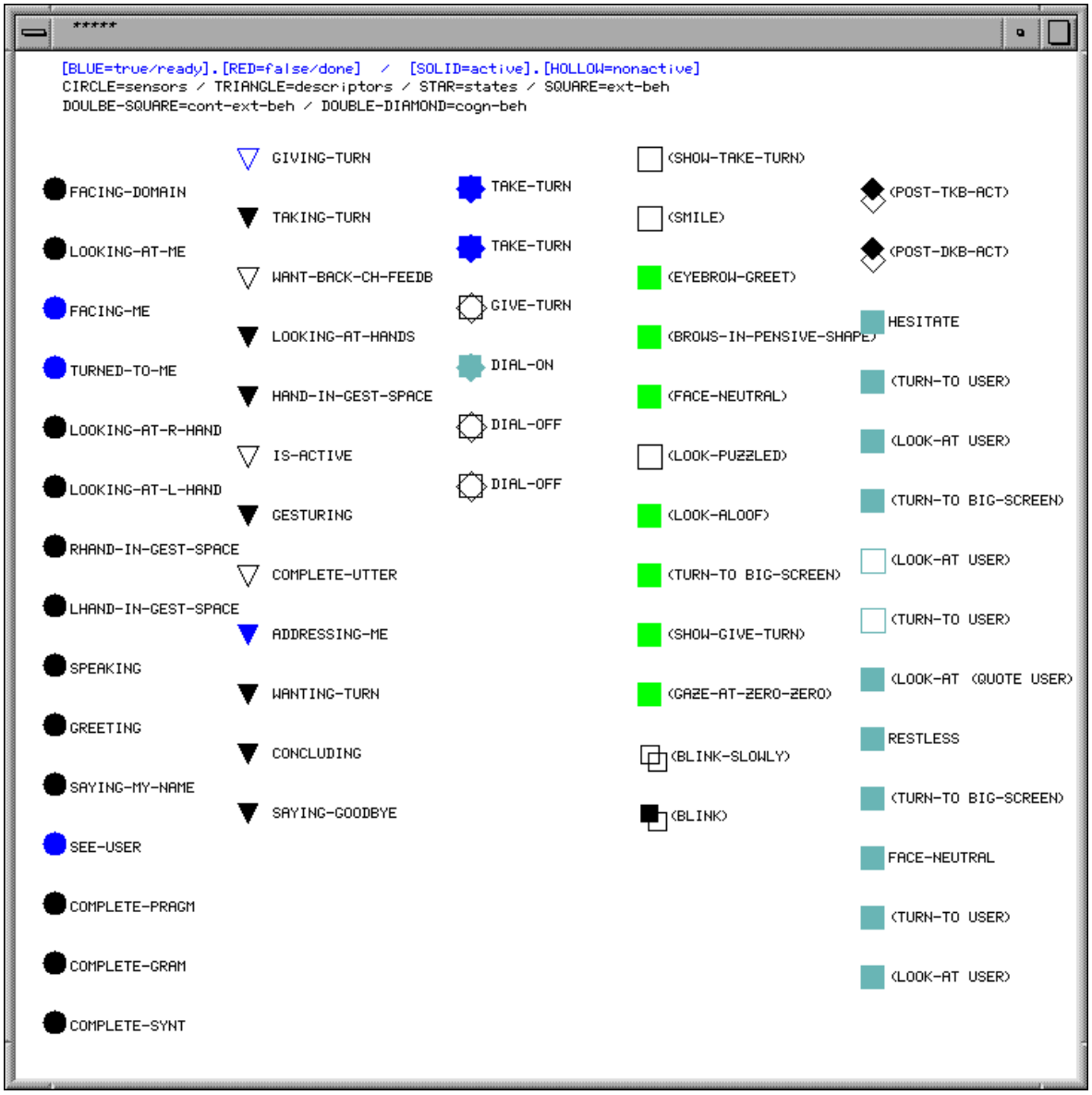


FIGURE 10-14. The Module Viewer allows a character designer to view the states of the modules designed, color-coded and updated in real-time.

Designing Humanoid Agents: Some High-Level Issues

11.

Having designed a dialogue-capable agent, and evaluated three versions of this humanoid, it is now time to take a step back and look at the issue of humanoid agent design in a larger perspective. In the real world, physics—and the workings of natural design and selection—dictate the way multimodal creatures look. In the digital world there is much more flexibility (or so we'd like to think). This flexibility (along with the fact that computer characters are neither animals nor human animals) leads us naturally to ask *"What* is my agent?", or more specifically, *"What kind of creature is this (collection of) software?"* The question is important for anyone who wishes to converse with an agent face-to-face.

In this chapter we will look at the question of system validity: What flexibility does the designer of a conversational computer controlled agent have in his designs? How much of that flexibility is limited by human communication capabilities on the one hand and technological limitations on the other? Is the *appearance* and *behavior* of the agent a *valid* representation of its capabilities? The discussion will have more questions than answers, but then again, this is uncharted territory and asking the right questions is more important at this point than providing what would inevitably be wrong answers. Some of the topics touched on here may be a precursor to the systematic evaluation of multimodal communicative systems.

11.1 Validity Types

The validity of a model is determined by reference to the real-world system or object it is intended to be a model of. In the case of communicative humanoids this object would be a human being, or more

specifically, a person engaged in face-to-face interaction. We can distinguish between at least three kinds of system design validity:

1. *Face* validity,
2. *Functional* validity (of control structures) and
3. *Structural* validity.

The first two validity types are especially relevant to the design of computer characters.

11.1.1 Face Validity

Achieving system face validity for a design involves making the system, on the surface, look and behave like the real thing. In the case of dialogue, an agent would have to seem to an observer like the real person engaging in a face-to-face interaction. No questions are asked of the underlying control structures to achieve the observed behavior of the agent.

11.1.2 Functional Validity

The functional validity of a system's control structures is the validity of what that system's control structures *do,* compared to those of the object modelled. For the agent's control structures to be *functionally* valid, the mechanisms controlling its behavior would have to have corollaries in the human mind, at least metaphorically. For example, in the functioning of the agent's system there would be processes and states that we could metaphorically refer to as "thinking", "listening", "attentive", "confused", etc. These processes and states also have a relationship to the system's other components that is functionally equivalent to the way components interact in the real system. A functionally valid system will, for all practical purposes, behave like the system it is trying to model: a response from it will be met with the appropriate counter-response, which in turn will be met with an appropriate counter-counter-reaction, etc. A moment's reflection quickly leads us to see that it is probably very hard to achieve face validity without system functional validity, although theoretically it may not be impossible.[1]

1. An example of face validity being achieved without functional validity is the use of fractal geometry to render very realistic-looking objects such as mountains, clouds, etc. The functional model of these phenomena would be achieved by actually modelling the individual atoms and light rays scattering off these.

11.1.3 Structural Validity

Finally, system *structural validity* is the amount of direct structural correspondence between the model and the object modelled. For a model to be structurally valid, it has to include all the necessary components of its real-world counterpart, including system architecture and its physical properties. For example, if we want to model a mind, it may be necessary to model the structure of the brain, although many believe this will not be required.

Eventually we might want our agents to share the same mental structures as people, but that may not be—and hopefully isn't—necessary for the purpose of building a useful agent. A useful agent, i.e. one that can participate with some skill in dialogue, needs high face validity, but as we already mentioned, this is hard to achieve without at least some functional validity to back it up.

11.2 Functional Validity in Humanoid Computer Characters

Functional validity may be applied to the design of humanoid agents along the following lines: A humanoid agent's outward behavior has to match the user's pre-conditioned (learned) expectation about the relationship between internal processing and morphology of a dialogue participant's behavior (Figure 11-1). For example, the user's mental model of a facial expression's meaning has to match the actual meaning of that facial expression. How to achieve this is an empirical question, and one that is likely to vary between cultures.

Let's put this in a more formal framework. For an agent to be a functionally valid conversant, two pairings that have to happen. The first being the match between the internal state of the machine and its expressions and behavior, denoted $\{\sigma, \sigma'\}$, the second being between the user's recognition of the expression and his or her interpretation of its meaning—i.e. relation to the expressee's mental state, denoted $\{\sigma', \psi\}$. As long as $\{\sigma, \sigma'\}$ and $\{\sigma', \psi\}$ approach a functional[2] correspondence, that is, we can make the assumption that a correct match exists for most or all $\{\sigma, \psi\}$ pairs, the agent's behavior will be a facilitator to the dialogue. In fact, as long as there is a better-than-chance correspon-

2. The term "functional" here is used in the conventional meaning of the term—i.e. what the system *does*. In this view a metaphor from human psychology can plausibly be mapped to the internal workings of the computer, as e.g. "thinking" could correspond to "processing utterance" and "confused" could correspond to "incomplete parse".

dence between the internal state and the expressions, the expressions will eventually always facilitate the dialogue because given time, the person would learn the correspondence. For practical purposes, however, one would want a much-better-than-chance correspondence to avoid frustrating the user.

11.3 What is my Agent?

A central question for agent design over the next decades is how to get around technological limitations that prevent us from achieving functional and face validity. Anyone who has read the preceding chapters in this thesis will by now have realized that physical makeup plays a part in making or breaking the fluidity and naturalness of face-to-face dialogue. And anyone who has ever walked into the bar in Star Wars I knows how hard it is to strike up a conversation with an alien that looks like that in Figure 11-2.

Nevertheless, computer characters should be represented outward in a way that conveys their functionality succinctly, without evoking false expectations in the user. For example, agents equipped with today's computer vision couldn't possibly recognize more than a handful of everyday objects, yet users might mistakenly assume that it can "see"

FIGURE 11-1. The state of the underlying mechanisms (σ) produces a facial expression σ′, which has to match, at least functionally, the user's intuition about the relationship between facial and mental (Ψ) states (σ′ ≡ Ψ). Of course, for a machine this would be a metaphor, and the only measure of its "correctness" is that it is beneficial for the communication.

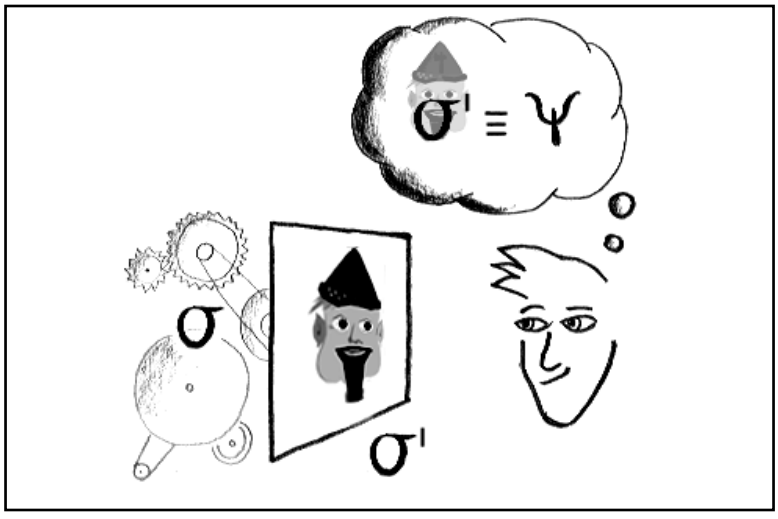


FIGURE 11-2. Anyone who wanders into the Star Wars [1977] bar is likely to wonder how to strike up a conversation with another guest.

objects in the surrounding just like they themselves can. This is a question of misjudged {1} perceptual capabilities. Speaker-independent speech recognition will undoubtedly be limited to a few hundred words for years to come (although pragmatic constraints will enable more top-down processing to improve it as we move toward situated characters). Users of speech-recognizing computers are invariably found to think that computers have a larger vocabulary than they actually do; this is a question of misjudged {2} language capabilities. Giving computer agents human bodies, users may easily think their agility equals their own. This is an issue of misjudged {3} motor skills. And finally, looking human makes people think you are as *smart* as humans. This is misjudged intelligence or {4} mental capabilities.

I think the solution to such inescapable problems lies in clever design; design that clearly shows the connection between appearance and abilities. Even more important is to implicitly and explicitly provide people with indicators of "mental" (computational) limitations in the way the agent *behaves*. An explicit way to achieve this is for example giving an agent the ability to guide a user in her interactions with it (e.g. "I know the names of all the planets but not their moons"). This will prove to be a very efficient guidance tool for helping users to adapt more quickly to the agent's limitations. Giving the agent a reduced ability to speak ("I know planets; not moons") will foster expectations on the user's side that the agent's understanding of speech is similarly limited. An implicit way to provide guideposts to the capabilities of characters is to link their inner functioning to subtle aspects of facial movement, intonational patterns and gaze control.

11.4 The Distributed Agent

As systems become more distributed and options for various kinds of implementations of humanoids become increasingly varied, the following question will eventually come up: Where is my Agent? The issue is more involved than one may think at first sight. This is not a matter of the Agent getting "lost" in cyberspace and cannot be solved by implementing a "search" program to locate your Agent. It is a matter of knowing "where" someone is when you're talking to them. Take a look at the following story.

11.4.1 Where is my Agent?

You're talking to a Mr. Lien at a restaurant. As he sits there in front of you at the small table in your dark little corner, his frontal brain lobes are contained in a jar in his living room at home; his visual processes in an automobile somewhere in Iceland, remotely connected to the two shiny cameras pointing at you form the other side of the table; his body has four arms, but only one of them is visible to you, the rest are plugged into material transporters—transporting his hands to god knows where in the world; and his "legs"—if you could call them that—don't look like they will ever be able to support the rest of him. And you wonder, *"Where* is this character?"[3] Is he at home, where his higher mental functioning resides, is he where is perception his located—somewhere in Iceland, is he where his hands are—where the "action" is, or is he here, the place where all of these seem to meet? You try to answer the question and then you give up. You tell him that you will consider talking to him again when he's pulled himself together. As you walk out you cannot but curse the designers of this agent for being so inconsiderate of its user's communicative needs.

And why did you get frustrated? For one, because Mr. A. Lien is an alien, you couldn't predict how it would respond to your actions, how its memory worked, or how it perceived you and the environment you met in (we covered this in the last section). Second, any references to events in the immediate environment such as actions of A. Lien during the conversation and the waitress that brought you the Brainblaster cocktail, were precluded because you had no way of knowing whether A. Lien had been paying attention (whatever that means for an alien) during those events. Since there is no centralized, localized place which action and perception are limited to (via a body), he could have been doing anything anywhere and not been present at all (a pair of cameras prove nothing). But most importantly, because there was a communica-

3. Readers of philosophy may recognize the theme of this muse—its precursor can be found in Dennett's [1981] excellent short story "Where Am I?"

tion time lag between the wrong pieces of Mr. Lien's "brain", its back channel never matched your pauses, the movement of the cameras was out of sync with its verbal output, and when it fell silent every now and then it was impossible to predict whether this was due to a "mental processing" delay, a "brain communication" delay or a general failure of a significant piece of its mental machinery.

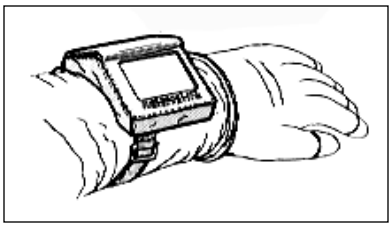


FIGURE 11-3. NASA scientists have designed a wrist-based computer with a touch-screen, intended as a portable assistant on space walks [NASA Tech Briefs 1995b].

11.4.2 Wristcomputer Humanoids

We will consider one version of the mobile agent: an agent that appears in the display of your watch. This idea isn't pure fantasy or science fiction; NASA has recently done some research with wrist-based, touch sensitive screens for space walks (Figure 11-3), AT&T are doing research on wristphones and various research is making it possible to miniaturize communications technologies, displays [Depp & Howard 1995], cameras and CPUs (Figure 11-3).

The question here is: what metaphors do we want for communication with machines that can have a distributed "brain" and "body"? If my machine agent talks through my watch, but has no visual sensory apparatus to sense me (or anything surrounding my watch) is the right metaphor that my agent is "in" my watch, or is it more accurate to talk about it being somewhere else, talking to me through a visual walkie-talkie? How about the situation where there is some sensory apparatus in my watch, but only very little? How about if all the sensory apparati were in my watch, but it's brain is somewhere else? These questions revolve around three things: {1} bandwidth limitations, {2} the breakup of an agent's mental functioning and {3} the metaphors we choose for the communication.

I will try to argue that the answers to these questions should be based on the mental limitations of the human user, and the limitations of the metaphors we use to simplify the interaction, *the communicative humanoid*.

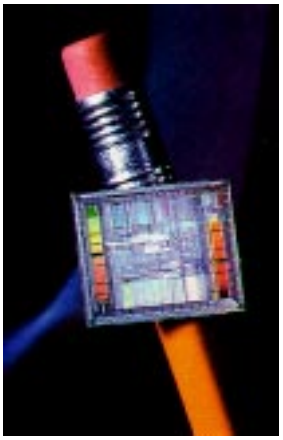


FIGURE 11-4. The Alpha 21064 chip from Digital Equipment Corporation measures 1.39 x 1.68 cm in size (this picture shows it roughly two times the actual size) and contains 1.68 million transistors. This is the chip that runs most of Ymir.

FIGURE 11-5. When humanoids start appearing on the LCD screen in our wrist computers, how will the primary communication problems manifest themselves?

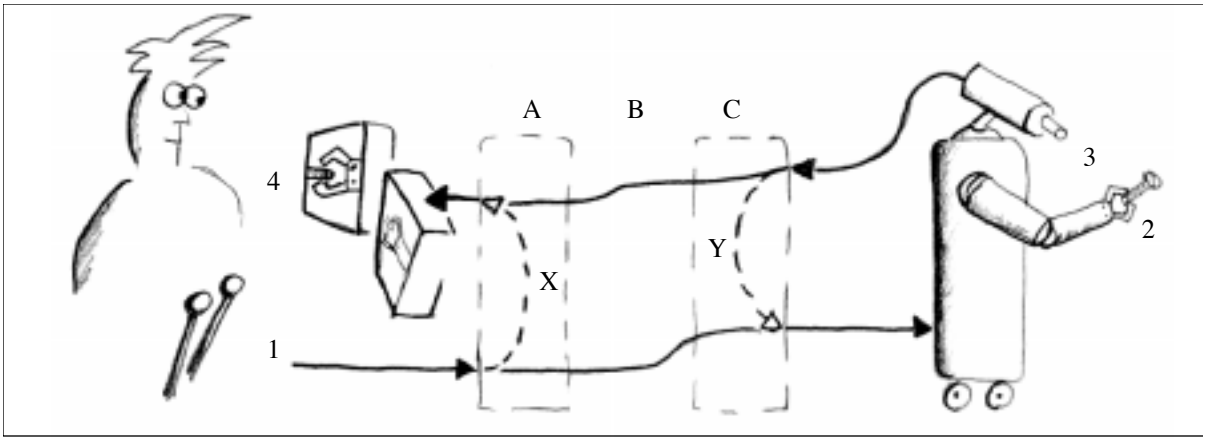


FIGURE 11-6. In a telerobot supervisory system, several paths define information flow. Here, the main loop of information flow is that from controls [1] to the robots actuators [2], through the robot's sensors [3] and back to the operator's displays [4]. System A provides a local feedback to the operator [X], system C provides a local feedback loop within the telerobot [Y]. Gap B represents some barrier, time, distance or inconvenience. (After Sheridan [1992].)

The question we need to answer is *Where can we accept limited bandwidth in multimodal communication?*

11.4.3 A Comparison to Teleoperation

There are several corollaries between teleoperation and agents with "distributed psychology". In Figure Figure 11-6 three loops characterize information flow: {1} The flow form controls to robot and back to displays, {2} the flow from controls directly to displays, and {3} the flow from robot sensors to robot manipulators. The reason these loops are important is the fact that a barrier exists in the transmission channel between the supervisor and the robot. This provides the reason for a local loop; a local loop displays immediate (time specific) data about the manipulation of controls without going through a sub-optimal channel, thus increasing the rate at which the operator can update his model of his own actions. Figure 11-7 shows the situation in face-to-face dialogue: Here the local loop goes through reactive paths in the robot itself [x]. This loop is responsible for responses under 1 second. A higher-level, slower loop takes care of administration of responses related to the process of dialogue. The third loop represents data flow through the rest of the agent's knowledge and reasoning systems. This is the slowest loop. The reader may recognize here the gross anatomy of Ymir. These loops all have a fixed relationship to one another (see Chapter 7.,

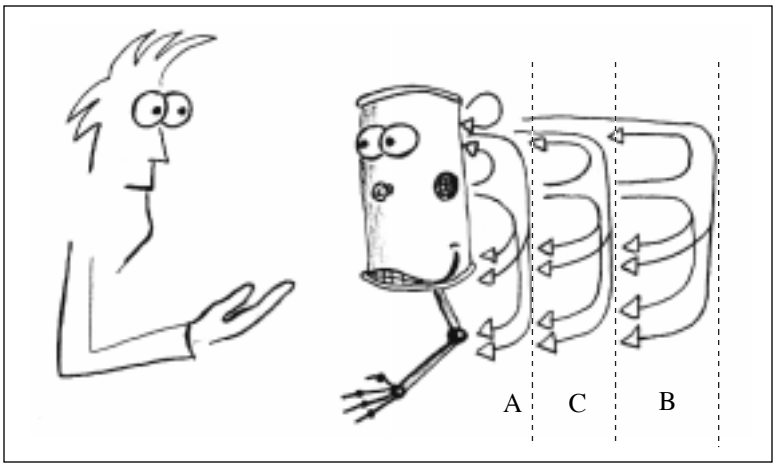


FIGURE 11-7. Many feedback loops exist in dialogue systems. A rough comparison between telerobotics and face-to-face conversation reveals some structural similarities. Here the human takes the role of the teleoperator, while the tincan humanoid corresponds to the role of a telerobot. Sections A, B and C in this figure correspond roughly to sections A, B and C in Figure 11-6. Loop A is highly reactive and the robot has very little control over it, yet it is to great benefit to the "operator" (human). (Of course such tight control loops also benefit the robot, e.g. the loop from the eye input to the eye's muscles). Loop C is the local loop of the robot—its "inner agenda". Loop B is the barrier that may exist with the input from the human to the robot's knowledge—this loop may be characterized in the same way as loop B in Figure 11-6, it constitutes time, distance and/or inconvenience.

page 92) which needs to be maintained in order for the dialogue to proceed at normal speed. Both the lowest and middle loop are highly time-specific; the top loop is semi-independent of time.

As we mentioned in "Back-Channel Feedback" on page 40, and as shown in the human subjects experiment ("Human Subjects Experiment" on page 161), breaking the lowest loop may result in discontinuities in the dialogue and overall lower satisfaction with the interaction. For an agent situated in a watch, with limited computational capacity in the device itself, it may be tempting to leave only the sensing devices in the watch, and move all computationally intensive processes to a remote location. However, as teleoperation has shown, this may lead to aliasing effects where the operator moves faster than the actual data display rate allows for. This invariably leads to error in operation, the most obvious in dialogue being overlaps in speech. All delay constants in the system (which are in fact unlikely to be constants) have to be measured to guarantee timely execution of events in each loop. To ensure the correct update time for gaze, its movements need to be driven by informa-

tion flowing in a tight loop from eye input to the gaze controlling mechanism. The update time for back channel feedback is (mainly) achieved by a close loop from the agent's hearing mechanism, to the vocal motor control and head motion.

The above analysis can be used to determine where we can "chop up" the wrist computer agent's psyche to distribute its functioning and make better use of computational resources. The reader shouldn't be surprised that what emerges is the basic structure of Ymir. For the loops in Figure 11-7 we have A = 100 [0 ~ 250] msec, C = 250 [150 ~ 1000] msec, and B > 1000 msec. Here we have approximate average total transmission times for each part of these loops. Notice that this is not just data transmission time, it is the *complete loop time*—data collection, processing, decision making, motor composition and motor execution. Thus, even with a very high bandwidth between the wrist computer and the remote location, going through a geosynchronous communications satellite you will always introduce a transmission delay of not less than 200 ms (uplink-downlink). That is clearly too high for loop A and maybe for loop B as well. A cellular connection will serve us better, but loop A would probably still need to be local to the agent's display. Several studies have indicated that this is precisely the reason why videophones and video conferencing hasn't caught on as was expected when the development of this technology started in the '50s [Whittaker 1994, Whittaker & O'Connaill 1993]: because the technology cannot support the high bandwidth necessary for correct synchronization between image and sound, as well as uneven refresh rates, the feedback in the reactive loop gets lost in the process. This leads people to choose telephones over videophones, where there is a higher data transmission rate and less synchronization problems.

11.5 Conclusion

Undoubtedly many problems will come up as we design more sophisticated agents and the systems get bigger and more complex. One problem with copying an activity such as face-to-face interaction in a machine, that integrates perception, planning and action, is scaling. This cannot be approached like a telephone network, where the mathematics of adding a certain number of new users is well understood and the problem scales well. The importance of using guidelines such as those presented in this chapter in pinpointing where possible problems could arise, and what those problems might look like, cannot be underestimated. However, this is just the beginning: We need to go far beyond the current understanding of communication and telerobotics to

be able to accurately estimate the efficiency, problems and satisfaction with such systems. But before we know how to design the systems, perhaps it is too soon to try to design evaluation methods.

Conclusions & Future Work

12.

In this last chapter we will give a summary of the contributions of this work and present proposals for extending Ymir. Specifically, we will look at the following issues: {1} contributions of this thesis and what remains to be done {2}, potential limitations of the architecture, and {3} how Ymir could support extensions such as dialogue history, simulated emotions, learning and multiple conversant situations.

12.1 High-Level Issues

Two high-level issues were to be addressed by this work. The first was the general issue human-computer interaction. The plan was to make an interface that takes advantage of people's knowledge about face-to-face interaction, turn-taking and perceptual abilities of interacting parties to provide a consistent metaphor for the relationship between human and computer. The prototype designed shows that these features can, in fact, be incorporated into a computer interface, and that the outcome gives its user a high-bandwidth communication channel with the computer. Expansion of the prototype presented here will result in robust agent-based systems that provide a powerful and intuitive new means for people to interact with computers. Potential applications are simply too numerous to list, covering such broad areas as education, office work (and work in general), entertainment and psychological and social research.

The second general issue addressed in this thesis was dialogue modeling. The Ymir architecture is a model of face-to-face communication, linking together a number of elements of multimodal dialogue for the purpose of providing a platform for creating communicative humanoids. Gandalf, the first prototype built in Ymir, has been provided with a set of skills that enabled him to carry on multimodal dialogue and exhibit

behavior comparable to that of a human engaged in dialogue. That it does exhibit behavior similar to a human conversational partner has been shown both by user testing and by a comparison to the Model Human Processor [Card et al. 1983]. Thus, Ymir has proven too be a sufficiently sophisticated architecture to provide the basis for designing interactive agents with language and multimodal capability.

Now lets look at the specific issues addressed in this thesis and how they were answered.

12.2 The Goals of Bridging Between Sensory Input and Action Generation

A main argument in this thesis has been that face-to-face dialogue happens on *multiple levels of granularity*, both in perception and action ("Challenges of Real-Time Multimodal Dialogue" on page 66). It is no surprise, therefore, that the model proposed here as a framework for multimodal agents, Ymir, is multilayered (Figure 7-10 on page 107). It uses the concept of layers (see "Layers" on page 92) to cluster processes that share similar time-constraints. Thus, highly reactive perceptual and action processes are contained in one layer ("Reactive Layer (RL)" on page 93), more reflective processes share another layer ("Process Control Layer (PCL)" on page 93) and the least time-dependent actions have their own layer ("Content Layer (CL)" on page 95). A fourth layer hosts the agent's ability to gracefully integrate multiple action goals into coherent outward behavior ("Action Scheduler (AS)" on page 94). Thus, as far as answering how to bridge from sensory input to action generation, Ymir has proven successful.

12.2.1 Continuous Input and Output Over Multiple Modes

The issue of continuity in multimodal interaction is a complex one. People manage to perceive multimodal acts to an extraordinary degree [Goodwin 1981], but entangling the rules how they do it has not been easy. The problem contains at least two parts: Segmenting out the significant parts of a unimodal act, and recognizing the function of multimodal acts in real time. In Chapter 5. ("Functional Analysis: A Precursor to Content Interpretation and (sometimes) Feedback Generation" on page 71) we argued that the latter was needed in order to do the former properly. While the former has gotten some attention in the literature, the latter has not been addressed, and was therefore made the focus in this work. The solution to the problem of continuous input advanced here lies in treating events at multiple levels of detail and integrating them in a way that supports directly the actions needed for the

dialogue interaction. Two kinds of processes were designed, *unimodal* and *multimodal*, the former providing data needed in the latter to determine the function of multimodal acts (see "Virtual Sensors" on page 98 and "Multimodal Descriptors" on page 99). Taking a simulated parallel approach, this model is likely to deal successfully with expansion—and thus the important issue of real-time constraints—should future implementation rely on massively parallel hardware.

The mirror problem to continuous input is continuous multimodal *output*. One part is the *coordination* of multimodal output, the other is *generation* of multimodal acts. The former, falling directly under the auspices of dialogue management, was the main focus here. Separating *coordination* from *generation* in this manner allows us to create direct links between the perceptual acts of a multimodal character and its actions, independent of the action's form. Decisions to act are made by a set of modules that monitor the conditions in the agent's perceptual and "cognitive" states ("State Decision Modules" on page 118). The actions of these modules are the *intentional phase* of a multimodal act; they in turn get evaluated in the Action Scheduler ("Action Scheduler (AS)" on page 94), which, depending on the current state of the agent's memory and motor state, turns them into appropriate motor actions.

12.2.2 Coordination of Actions on Multiple Levels

Having proposed a highly distributed system, we need a way for modules to speak to one another. For this purpose, Ymir uses three blackboards. For example, to enable separation between output generation and output *coordination*—or process control—the PCL and CL communicate with each other through a dedicated blackboard ("Blackboards" on page 96) using special communication primitives (Figure 8-8 on page 119). Such "limited access" blackboards allow us to accommodate multiple modules at multiple levels of granularity without sacrificing system comprehensibility.

Looking at Ymir at a more detailed level, the State Decision modules in the Process Control Layer (see "State Decision Modules" on page 118) can be made to turn on or off modules in that same layer, as well as in the layer below, the Reactive Layer. This is another promising method for exerting run-time control over multiple modules at multiple levels. However, this scheme might need to be expanded to deal with more complex perceptual modules.

12.2.3 Lack of Behaviors

One problem we encountered in our discussion of autonomous agents was the amount/number of behaviors one can put into the system ("The Lack of Behaviors Problem" on page 85). Ymir solves this problem by

allowing hierarchical definitions of behaviors and enabling other parts of the system to access these at any level (“Representation of Behaviors” on page 100). The main limitation of this scheme, as embodied in the implementation, is the lack of composition rules for piecing together various parts of various behaviors, to allow for the emergence of new behaviors (see “Inherent Limitations” below).

12.2.4 The Natural Language Problem

As demonstrated in the J. Jr. prototype (“The Reactive-Reflective Integration Problem” on page 85), adding natural language capabilities to agents based on most of the current reactive-only architectures is problematic. In Ymir the layer containing the agent’s knowledge base(s), Topic Knowledge Base (TKB) and Dialogue Knowledge Base (DKB), is separated from the layers exerting control over when actions happen. This allows for seamless integration of high-level behaviors such as language with lower-level ones.

While the separation of TKB and DKB from the rest of the system deals successfully with the issue of integrating natural language into a hybrid reactive-reflective system, we still need to define the mechanisms that interpret and generate the linguistic output. The multimodal interpreter in the Gandalf prototype is very simple, and works well in its limited arena, but it leaves several issues unanswered. At the top of this list is how to deal with *spatial data* in relation to language. Only considerable additions to what has been implemented in the prototype can clarify how extensible this scheme is.

12.2.5 The Expansion Problem

In the J. Jr. chapter (page 81) we saw how expanding the agent’s behaviors and perceptual capabilities became a problem of system complexity. This is a general problem for systems based on finite state machines, and other systems which use mechanisms at the same level of granularity. Because Ymir separates reactive processes from reflective ones, expanding either one is possible without interfering with the other. Using communication between the Process Control Layer and the Content Layer (see Figure 8-8 on page 119) allows us to continuously expand the system’s topic knowledge without having to change a single thing regarding the real-time execution of actions.

One question that remains unanswered is how well the idea of hierarchical perception works. More specifically, can we add more advanced perceptual modules into the Process Control Layer and the Content Layer—does this make sense? In the current implementation these are mainly contained in the Reactive Layer. More research is needed.

12.2.6 Goals: Conclusion

In conclusion, Ymir provides a foundation that fulfills the original goals it was meant to fulfill: It bridges between multimodal input analysis and multimodal output generation, allowing the construction of autonomous humanoid agents capable of continuous, interactive dialogue with people, comparable to human face-to-face conversation, proposing a number of mechanisms and ideas to fill in the missing knowledge that could make this possible.

12.3 Inherent Limitations

While Ymir provides quite a number of solutions to generating communicative humanoids, it also is obvious that as an architecture for accommodating *everything* needed in multimodal dialogue, Ymir has its limitations. What exactly those are is not clear at the present; the hope is that it will provide at least a solid stepping stone to the next level. However, it may be useful for other researchers to provide an insight into where the weak links may lie.

12.3.1 Reactive-Reflective Distinction

One can guess that the strict distinction between reactive and reflective behaviors proposed may need to be relaxed, perhaps to the point where the division into Reactive and Process Control layers no longer make sense. Such a course of events, however, is likely to make the construction of complex characters quite daunting, and may call for new principles for conceptually segmenting the systems responsible for those functions. Schemes such as those used by Blumberg & Gaylean [1995], which allow modules to be grouped, could be a potential solution in this case. Still, the level at which the modules are designed is likely to be adequate for constructing even relatively complex agents.

12.3.2 Communication Between Layers

It may also happen that communication between the Process Control Layer and the Content Layer eventually becomes so complex that they should be considered a single system, or a larger set of smaller systems. Again, mechanisms must be introduced to steer away from increasing numbers of identical modules, messages or rules, since designing these becomes difficult.

12.3.3 Behaviors and Action Generation

A limitation of the current implementation of Ymir relates to the way behaviors are generated. Since all behaviors are named (with the exception of spatial values being inserted where needed), a programmer needs to design hundreds of behaviors to create a fully capable character. A better solution would be to create a motor representation scheme where more flexibility exists in the generation of behaviors, such that new actions could be composed of old ones. This would also be a necessary change to accommodate motor learning.

Having looked closely at the main issues Ymir was meant to provide a solution to, we are now ready to delve into its limitations on a more global scale—how will more general behaviors be integrated into Ymir?

12.4 Extending Ymir/Gandalf

The Ymir architecture was designed over a period of about 3 years. In the process, many options were considered and many were rejected. The foundation for its design and design trade-offs was laid with the following goals:

1. Implementable by a single person,
2. incorporating—or allows for the extension to incorporate—as many features of face-to-face interaction as possible,
3. flexible, modifiable, and
4. real-time performance.

Rather than trying to get everything "right" the first time around, the system design was slanted toward *flexibility* in the architecture. There are therefore several levels of "entry" that a character designer could access the model presented, from the lowest level of Logic Net design, state modules and virtual sensors, up to the hybrid framework that the whole system relies on. Parallel to this is reuse of the LISP code written, i.e. the implementation itself: parts of this code could be used, for example the knowledge bases, motor scheduler or intonation analysis, or it could be used whole-sale, with other software modules entering the system at particular points.

At the low level, the most obvious place to start would be to experiment with various new modules. At this level one could add learning or self-regulation, for example by using "b-brain" modules [Minsky 1987] that modify the conditions in other modules. This could make the interaction more robust and the system more able to deal with varied conditions. One could also enter the system at a slightly higher level, adding new types of modules, keeping part or all of the current ones. At the highest level one could use the general architecture of the layers and

blackboards—even the modules themselves—but make entirely new module mechanisms. These are all ways to improve what Ymir currently has to offer. But we also want to add new features such as emotion, mobility, multiple participants, etc. For this we have to stretch the platform to incorporate new mechanisms.

12.4.1 Where Are We Now? Current Status

Several advanced features have been shown in the Gandalf prototype, among them deictic gesture recognition/reaction generation and single pair turns such as question->answer and request->action. Most of the current prototype's limitations come from the simplicity of the knowledge bases. Expansion of these would allow more features to be added. One such feature is multiple query-response turns: "Move the box / Do you mean the blue one? / No, sorry, move the sphere / This one? / Yes / Which way?", etc. Another feature that more extensive knowledge bases could allow are directives such as "Look at me" or "Look at me when I'm speaking". These are interesting language-based tools that help in the process of dialogue and may prove crucial for the success of future communicative agents.

Other features that could be added include emotional simulation, affecting behavior at many levels, linking gaze directly up with visual input and the spatial knowledge base, dialogue history, anaphora and other references to dialogue events, as well as recognition of more types of gestures and more extensive multimodal event representation. We would also want to extend Gandalf to be able to converse with more than one agent or person at a time. We will look at these in turn.

12.4.2 Multiple Turns for Single Utterances

To add multiple pair turns, one would need to create a topic knowledge base (TKB) with a parser that can mark the already-parsed utterances with the state of the dialogue. Such systems have already been built [Allen 1987] and we will not go into those here in any detail. The interface to this TKB would have to add a few primitives to the communication with the Process Control Layer (PCL). First, the PCL needs to recognize what class of utterance is being produced at any stage in the dialogue. This is needed to allow it to automatically add correct process control behaviors such as hesitations, paraverbals, even short verbal utterances such as "I know this...hang on a second" while gazing upward. Second, the PCL needs to send messages to the KB that don't exist in the current Gandalf prototype. These include telling it whether a certain utterance had been delivered, how it might have been modified, and what kinds of fillers, extraneous utterances and actions were taken. This kind of extension is relatively straightforward, and is currently being made.

12.4.3 Dialogue Process Directives

To enable a user to address an agent created in Ymir with directives such as “Look over there”, “Turn your head away”, “Stop staring!”, one needs special connections from the Dialogue Knowledge Base (DKB) to the Action Scheduler (AS) or the decision modules. A phrase like “Look at me” would produce an internal record, directed at a class of decision modules in the Process Control Layer, causing them to fire. A phrase like “Stop staring!” would produce a “constraints record” directed at a collection of behavior modules in the AS. These constraints would need to be fulfilled when the behaviors get executed, in the same manner that the motor composer trades off between behavior modules based on the time they take to execute. The action `Look-at-User`[1] (or the more advanced version `Look-at-Person[X]`) would have to fulfill the condition `(not Look-at-User)`. Whenever activated, the process computing spatial location for the user would have to modify the value to not coincide with the user’s location. How extensible this scheme is remains to be seen, but it seems certain to work for at least a substantial number of simple cases.

12.4.4 Emotional Simulation

Emotions affect behavior at many levels, although reactive behaviors may show less effect than medium- to long-term planning ones. In keeping with the modular approach to character building, an emotional module could reach in to both the decision modules, affecting variables such as effective lifetimes and responsivity to conditions. It could also change the behavior modules in certain, more powerful, ways, for example by limiting the number of options for executing certain behaviors, or changing the time scales of execution. This would require a substantially more powerful Action Scheduler, and additional features for the behavior modules, but would no require a change in the Ymir architecture.

12.4.5 Spatial Sensors & their Link to Spatial Knowledge

If we want the agent’s gaze to represent information collection on its part, the eyes have to be connected directly with the flow of visual input. This could be achieved simply by using cameras for eyes. Such a system would have to keep a spatial knowledge base updated every time the agent looked in a certain direction. Without going into the details of object recognition and maintaining the permanence of objects from moment to moment, such an extension would fit nicely into the model

1. See “Spatio-Motor Skills” on page 103.

proposed for spatial information storage accessible to both the Process Control Layer, Knowledge Layer and the Action Scheduler presented in the section "Spatio-Motor Skills" on page 103.

12.4.6 Dialogue History

In the current implementation of Gandalf the history of events simply piles up in the blackboards; there are no processes available for picking out events to make them available to the knowledge bases. This makes it impossible to refer to for example past dialogue events with expressions such as "...when you looked at me and said 'hello'". Obviously we need such mechanisms to provide a full system. The matter is mainly one of adding retrieval mechanisms to the knowledge bases; ones that would look in the various blackboards and "re-package" the accumulated events (such as `look-at-user`, `user-looking-at-me`, etc.) in a format that is directly accessible to the parsing mechanisms in the knowledge bases. This could also allow for references to past topic events such as "Go back to when the airplane was on the other side of the runway". Albeit not simple, this addition can build on knowledge base work already done in other systems [cf. Tanimoto 1987].

12.4.7 Advanced Gesture Recognition & Multimodal Event Representation

How do we determine what kind of gesture is being performed? Recognizing pointing gestures when that is the only kind possible is trivial in contrast to the situation where any gesture is possible—deictic, pantomimic, iconic, emblems, self-adjustors—and they all have to be discriminated in real-time. The solution proposed her is one where any kind of feature from any mode can support in deciding whether a certain behavior constitutes one type of gesture or another. For example, hearing the word "that" along with an extended arm might be enough of a trigger to classify the event as "deictic". Likewise, a verb combined with a complex hand motion and gaze that intersects hand position could be enough of a cue to signal an iconic gesture. Whatever may be a winning strategy for each kind of dialogue event, as argued in Chapter 5., the function of an event needs to be found before an action can be taken, and an action sometimes also needs to be taken before further processing can take place, as in the example of a deictic gesture where you *have* to look in the direction pointed to be able to follow the dialogue.

Along with the need to classify a number of manual gestures with different functions comes a need to package multimodal dialogue events so that they can be easily related to each other. Currently, speech is stored in frames (parse templates) where information about the words, type of utterance, type of dialogue event and intonation are stored. We need

meta-frames that store combinations of multimodal events, perhaps at more than one level of detail. This would enable a knowledge base to access dialogue events in a coherent manner, similar to what people do when reviewing these events (a person does not store an event such as rising final intonation separately from the words spoken, but rather combines the two into packages called "questions").

12.4.8 Multi-Participant Conversation

To allow Gandalf to participate in conversations with more than one user, or another agent, we need to first modify the perceptual modules particularly designed to detect events such as "looking-at-me" and "addressing-me" to use variables, where the variables can be assigned to the dialogue participants. This is not a radical addition to the architecture, but one that requires some additions in the dialogue knowledge base. These include coding of dialogue priority—currently the user has priority (and drives the interaction). If we want the agent to be able to take initiative, it needs to know when and how it is allowed to interrupt the user or other artificial agents. We might also have to look at the turn-taking mechanism to allow multi-participant information to influence turn behavior. Again, these are not major reworkings of the architecture—they can be built directly on the functionality that Ymir provides.

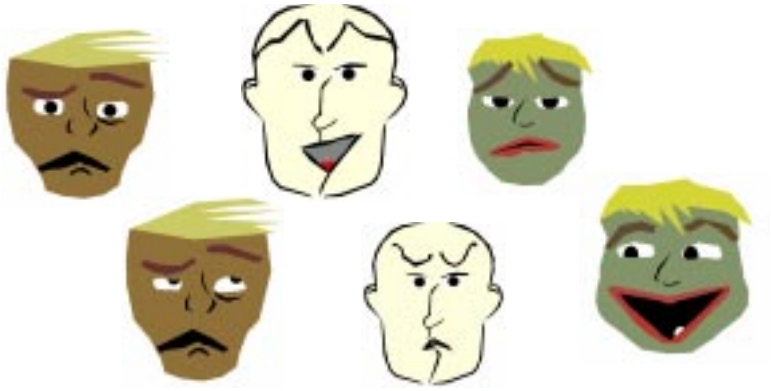
Whatever the future may hold for Ymir and Ymir-like architectures, its modular design makes it very likely to be useful for a number of years to come.

Character Animation

APPENDIX A1

This appendix describes ToonFace, the system used for animating Gandalf's face and hand. It employs a simple scheme for generating effective facial animation (Figure A1-1). It differs from prior efforts for character animation primarily in its simplicity and its way of representing facial features. The next section discusses the background for this work, as well as its motivation and goals. Section A1.2 describes the particulars of the drawing and animation routines. Section A1.3 gives a comparison between ToonFace and the Facial Action Coding Scheme (FACS, Ekman & Friesen 1978). Lastly, current applications and future enhancements are described in section A1.4. A quick user guide to the Editor and Animator are found in Thórisson [1996].

FIGURE A1-1. Examples of faces and expressions generated in ToonFace.



A1.1 Background, Motivation, Goals

While computer graphics work concerned with faces has to date focused extensively on visual appearance, interactivity and effectiveness for information transmission via the face has not been of primary concern. As the modes of speech, gesture and gaze become a routine part of the computer interface [Thórisson 1995, Koons et al. 1993, Bolt & Herranz 1992, Bolt 1980, Thórisson et al. 1992, Britton 1991, Neal & Shapiro 1991, Tyler et al. 1991] the demand increases for effective facial displays on the computer's side that can facilitate such multi-modal interaction.

Making facial computer animation look convincing has proven to be a difficult task. A common limitation of physically-modeled faces [Essa 1995, Essa et al. 1994, Waters 1990, Waite 1989] and computer-manipulated images of real faces [NASA Tech Briefs 1995, Takeuchi & Nagao 1993] is that their expressions is often strange looking or vague. An ideal solution to this would be to exaggerate facial expression, but within a physical modeling framework this may look unconvincing or awkward. An alternative is what might be called a "caricature" approach [Thórisson, 1994, 1994a, 1993a, Britton 1991, Laurel 1990] where details in the face are minimized and the important features therefore exaggerated (see Hamm [1967] for an excellent discussion on cartooning the head and face). In this fashion, Brennan [1985] created a system that could automatically generate caricature line-drawings of real people from examples that had been entered by hand. Librande [1992] describes a system called *Xspace* that can generate hundreds of artistically acceptable two-dimensional drawings from a small example base. Simplified faces seem like a very attractive alternative to physical modeling for animating interface agents, both in terms of computational cost and expressive power.

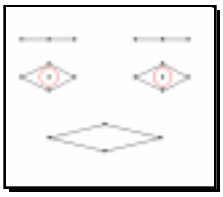


FIGURE A1-2. In ToonFace, seven objects comprise the animated parts of the face: Two eye brows, eyes and pupils, and one mouth. Control points (shown as dots) can be positioned anywhere within the face, by selecting and moving them with the mouse.

Most current systems for facial animation are very complex, include between 70 and 80 control parameters [Essa 1995, Essa et al. 1994, Terzopoulos & Waters 1993, Waters & Terzopoulos 1991, Waters 1987] require powerful computers and seldom run in real-time. There is a clear need for a simple, yet versatile method of animation that allows for interactive control. ToonFace is an attempt to create such an animation package. The primary goal of ToonFace is to create facial expressions in real time in response to a human interacting with it. ToonFace meets this requirement by being simple: mostly two-dimensional graphics with five kinds of polygons (three of which are user-definable) and four kinds of polygon manipulations. It employs very simple linear interpolation methods for achieving the animation—a clear win under time-constraints. By reducing the degrees of freedom in the movements of the face to a managable number (21 df), it is easier to control of the face than in most other approaches. A secondary goal of the system is that it

meet mininal criteria for graphical quality and look. The scheme employed allows people to use their own artistic abilities to create the look that they need for their system.

A1.2 ToonFace Architecture

ToonFace consist of two parts, an Editor and an animation engine or Animator. The Editor allows a user to construct a face within a point-and-click environment. The Editor runs on an Apple Macintosh™ computer in Macintosh Common Lisp (MCL) [Macintosh Common Lisp Reference 1990, Steele 1990]. The Animator is a C/C++ program running on an SGI using OpenGL [Neider et al. 1993] routines for real-time rendering. We will now look at how a face is represented in ToonFace and the drawing and animation routines.

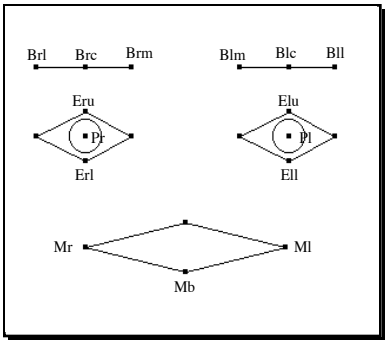


FIGURE A1-3. Codes used for the animated control points (seealso Figure 8-11 on page 123).

A1.2.1 Facial Coding Scheme

A face is divided into seven main features: Two eye brows, two eyes, two pupils and a mouth. The eye brows have three control points each, the eyes and mouth four and pupils one each (Figure 2).

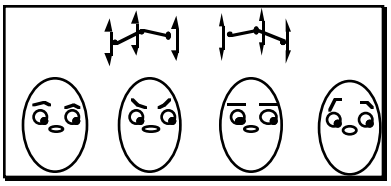


FIGURE A1-4. Eye brows have three control points, each with one degree of freedom in the vertical.

Control points that can be animated are given the codes shown in Figure 8-11 on page 123. These points were selected to maximize the expressive/complexity tradeoff. In the case of points that can move in two dimensions, each dimension is denoted as either "h" for horizontal or "v" for vertical. The following is a complete list of all one-dimensional motors that can be manipulated in a face [control point number in brackets]:

BRL = BROW/RIGHT/LATERAL [3];
BRC = BROW/RIGHT/CENTRAL [2];
BRM = BROW/RIGHT/MEDIAL [1]
BLL = BROW/LEFT/LATERAL [6];
BLC = BROW/LEFT/CENTRAL [5]; BLM = BROW/LEFT/MEDIAL [4]
ERU = EYE/RIGHT/UPPER [7]; ERL = EYE/RIGHT/LOWER [9]
ELU = EYE/LEFT/UPPER [8]; ELL = EYE/LEFT/LOWER [10]

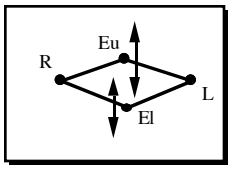
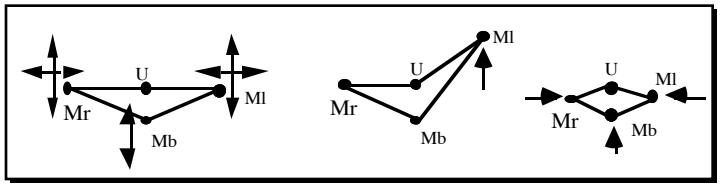


FIGURE A1-6. Each eye has four control points, but only two of those move. Upper (Eu) and lower (El) control points have one degree of freedom each in the vertical.

FIGURE A1-5. The mouth has four control points, three of which actually move. The ones on the sides (Ml & Mr) have two degrees of freedom, the bottom control point (Mb) has one.



PLH = PUPIL/RIGHT/HORIZONTAL [15];
PLV = PUPIL/LEFT/VERT [15]
PRH = PUPIL/RIGHT/HORIZ [16-H];
PRV = PUPIL/RIGHT/VERT [16-V]
MLH = MOUTH/LEFT/HORIZONTAL [14-H];
MLV = MOUTH/LEFT/VERTICAL [14-V]
MRH = MOUTH/RIGHT/HORIZONTAL [13-H];
MRV = MOUTH/RIGHT/VERTICAL [13-V]
MB = MOUTH/BOTTOM [12]
HH = HEAD/HORIZONTAL [17-H]; HV = HEAD/VERTICAL [17-V]

Horizontal motion is coded as 0, vertical as 1. Each of the motors can move a control point between a minimum and a maximum position (for a given dimension). Thus, max and min values mark the limits of movement for each motor. For the eyes and head, these are given in degrees, (0,0) being straight out of the screen; upper left quadrant being (pos, pos), lower left quadrant being (pos, neg). Figure A1-1 shows these limits as they appear graphically in the Editor. A line extends the full range of a control point's path. The limits can be changed by clicking on and dragging the ends of these lines.

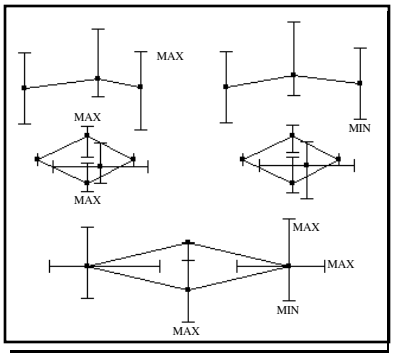


FIGURE A1-7. Limits of control point movement and direction of their dimensions.

A1.2.2 Drawing Scheme: Polygons

As mentioned before, drawing is done by filled, two-dimensional polygons. There are three kinds of user-manipulable polygons which all can have an arbitrary number of vertices. A new polygon is created by selecting the desired type from a menu, then selecting the feature or control point to attach it to (unless it is a free polygon). A polygon is moved by dragging it; its vertices are changed by dragging them to the desired locations. A new polygon always has eight vertices, which can be deleted or added to as desired.

Free Polygons

This is the simplest kind of polygon in the system. Free polygons are simply drawn in place and cannot be animated. They are used for constructing features that do not need to move relative to other features, including hair, ears, decorations, scars, etc. An example is given in Figure A1-8.

FIGURE A1-8. Free polygons are used for objects that don't have to move relative to others, like hats, hair, nose and ears.

Feature-attached Polygons

These polygons are associated with a whole feature. An example is a polygon representing an eye brow (Figure A1-9). These polygons are animated in relation to the whole feature: if one point in the feature moves, all the points on that polygon are recalculated and redrawn: as a result, the polygon changes shape.

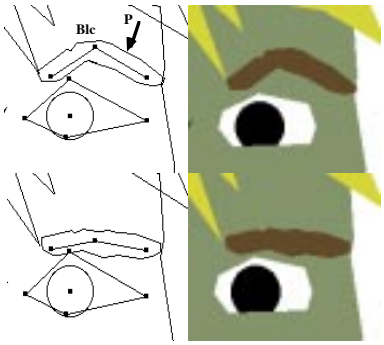
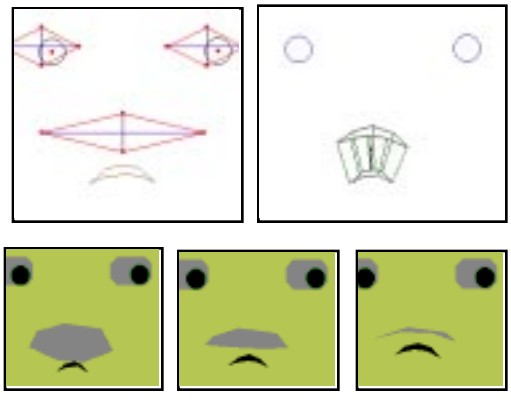


FIGURE A1-9. As the central control point on the left eye brow (Blc) is moved down, the vertices of its attached polygon (P) are recalculated according to how the angle of the lines between the control points changes. The left side shows the control points of the eye brow with connecting lines, the right side shows the polygons when filled.

Point-attached Polygons

A point-attached polygon only changes form/position when a single control point—the point to which it is attached—changes position. The user defines two states for the polygons, one showing how it should look when its control point is at its max position, the other corresponding to its min value (Figure A1-9). When the control point is moved during animation, a linear interpolation is performed between the polygon's two states.

FIGURE A1-10. Polygons attached to a single control point have two defined states (shown in the upper right with lines connecting common vertices). As the control point moves (in this case the bottom mouth point), the vertices of the polygon are interpolated between the two predefined states.



Drawing Order

For purposes of making features overlap correctly, three kinds of special-case polygons are used. *Hole polygons*, *pupils* and the *face polygon*. Hole polygons are the insides of the eyes and mouth. When the face is drawn, the hole polygons are drawn first, then the pupils, then the face polygon—except for the regions defined by the hole polygons—then the free polygons, then point-attached polygons, and lastly the feature-attached polygons:

STEP
1. DRAW (HOLE POLYGONS)
2. DRAW (PUPIL POLYGONS)
3. DRAW (FACE POLYGON) — (AREAS DEFINED BY HOLE POLYGONS)
4. DRAW (FREE POLYGONS)
5. DRAW (POINT-ATTACHED POLYGONS)
6. DRAW (FEATURE-ATTACHED POLYGONS)

A1.2.3 Interpolation Algorithms

Figure 12. Example of polygon point interpolation (see text).

The control points of a face's feature are connected by lines, as shown in figures 4, 5, 6 and 10. These lines are used to determine how the feature-attached polygon's vertices move when any single control point on the feature is moved. A feature like the left eyebrow has three control points (Bll, Blc, Blm) which all move in the vertical dimension. In Figure 12 h0 and h1 are the horizontal positions of Blm and Blc; the vertical would be {v0, v1}. From these the slope of L1 is determined:

$$SL = (V1 - V0) / (H1 - H0) \quad \textbf{(A1.1)}$$

The y-intercept of line L1 is given by:

$$IYL = V0 - (SL * H0) \quad \textbf{(A1.2)}$$

The following method is then used to calculate the position {x,y} of a vertice v on a feature-attacghed polygon P

P = {V1, V2, V3, ... }

V = {X , Y}

$$X = H0 + (VRL * (H1 - H0)) \quad \textbf{(A1.3)}$$

$$Y = (X * SL) + IYL + D \quad \textbf{(A1.4)}$$

where vrl is the relative horizontal position of point v betweeen h0 and h1 (along line L1) and d is the distance of point v from L1. This is exemplified in Figure 10: When the control point Blc is moved down, vertices on polygon P move to keep a constant distance to the lines between the control points, resulting in a new shape for the eyebrow.)

The feature lines are not used for point-attached polygons. These simply have two states, one for the control point's max position, and another for its min position (Figure 11). The following linear interpolation method is used to calculate a point-attached polygon's vertice (v) value {x,y}:

V = {MIN-X , MIN-Y, MAX-X, MAX-Y}

$$X = V_{MIN}\text{-}X + (P_{CTRL} * (V_{MAX}\text{-}X - V_{MIN}\text{-}X)) \quad \textbf{(A1.5)}$$

$$Y = V_{MIN}\text{-}Y + (P_{CTRL} * (V_{MAX}\text{-}Y - V_{MIN}\text{-}Y)) \quad \textbf{(A1.6)}$$

where P_{CTRL} is the position of the associated control point along its min-max dimension (a float between 0.0 and 1.0).

A1.2.4 Animation Scheduling Algorithms

The Animator part of ToonFace uses a multi-threaded scheduling algorithm to simulate parallel execution of motors. The main loop has a constant, loop-time, which determines the number of animation frames per second. The value for this constant should be equal to the maximum time the main loop could ever take to execute one loop. In the current implementation this constant is set to 100 ms, giving a fixed rate of 10 animation frames per second. When a command to move multiple motors is received, the total time this action is supposed to take is divided into loop-time slices. Since all motors are independent from each other, separate slices are made for each motor. So for a close-left-eye command (i.e. control point Elu) of a 500 ms duration, 5 slices would be made for the left eye, each slice to be executed on each main-loop. If the eye is fully open when the command is initially recieved, the eye will be 20% closer to being fully closed on each loop, and fully closed when the last slice has been executed. If a command for closing both eyes in 500 ms were to be given, a total of 10 slices would initially be produced and each time through main loop one slice for the left eyelid and one slice for the right eyelid would be executed, bringing both eyes to a close in 500 ms. If all pending slices have been executed before the 100 ms loop-time constant has been reached, the program waits the remaining time, thus guaranteeing a constant loop time.

Here is a rough outline of the main loop in pseudo-code:

```
LOOP FOREVER
START-TIME = READ-CLOCK
COMMANDS-RECEIVED = READ SOCKET INPUT
IF COMMANDS-RECEIVED
FOR EACH MOTOR IN COMMANDS-RECEIVED
MAKE-SLICES
FOR EACH MOTOR
EXECUTE-ONE-SLICE
PAUSE (LOOP-TIME — (READ-CLOCK — START-TIME))
```

The faster the rendering, the lower the loop-time constant can be set, resulting in smoother animation. The value for this constant is most easily chosen by experimentation, since execution time of depends on various factors, such as number of slices in each loop, amount of commands received per second, etc., whose interactions are difficult to predict.

It is expected that the program connecting to the ToonFace Animator contain libraries of standard motions, such as smiling, frowning, neutral appearance, etc. This is a non-trivial issue and will not be discussed here.

A1.3 The ToonFace Coding Scheme: A Comparison to FACS

The Facial Action Coding System (FACS) [Ekman & Friesen 1978] is a system designed for empirical coding of human facial expressions. The FACS model is based on a simplification of the muscle actions involved in producing human facial expression, where muscles are grouped together into what the authors call Action Units. Waite [1989] modeled a human face based on a control structure that incorporates several of the action units described in Ekman & Friesen [1978]. In her system, the action units are represented by collections of data points which are covered by a single rendered surface that mimics human skin. The approach taken does not automatically solve how to draw the eyes, control gaze, or add other decorative features (such as ears or hair) to the rendered face. Because the system relies on a model of muscles and bone structure, it is computationally intensive. More recently, Takeuchi & Nagao [1993] describe a system that tries to model a real face in three dimensions based on a similar approach, and Essa [1995, Essa et al. 1994] describes a computational extension to FACS.

The ToonFace coding scheme is not intended to be a competitor to FACS—it simply provides a new way to code facial expressions that requires less detail. Control points were selected to maximize the expressivity/complexity tradeoff. Compared to prior computer systems based on FACS, ToonFace allows for animation with more of a cartoon style look. The motivation for the ToonFace control scheme has already been discussed. However, a comparison to FACS may help the interested reader get a better understanding of the limits and possibilities of this scheme. It should be noted that since the FACS coding scheme is quite complex, the FACS Manual [Ekman & Friesen 1978] is recommended for those who wish to seek a thorough understanding of the issue.

ToonFace is a considerable simplification of FACS, but it is precisely for this reason that it is an attractive alternative. The head motions of humans have three degrees of freedom: head turn, medial (forward-backward) head tilt , lateral (side to side) head tilt. ToonFace simplifies this into two degrees of freedom, eliminating the lateral head tilt. For the upper face, the only features that are identical between the two are the eyes, which have 2 df each. Action unit (AU) 1 (inner brow raiser) and AU 4 (inner brow lowerer) are represented in ToonFace by Bm, with AU 4 approximated by motor Bm having an extended range downward (this depends on the particular face design). AU 2 (outer brow raiser) is approximated by motors Bc and Bl, which also help in capturing motions involving AU 1. Eu, or Eu and El together, approximates the following AUs: AU 5 (upper lid raiser), AU 7 (lid tightener), AU 41 (lid droop), AU 42 (eye slit), AU 43 (eyes closed), AU 44 (squint), AU 45 (blink), and AU 46 (wink). The only one left out from the upper face is AU 6, cheek raiser and lid compressor.

For the lower face, AUs 9 (nose wrinkler), 10 (upper lip raiser) and 17 (chin raiser) are not addressed in ToonFace. Ml represents the motions involving AUs 15 (vertical lip corner depressor), 25 (vertical lips part) and 26 (jaw drop). No differentiation is made between AU 26 and AU 27 (vertical mouth stretch), since the jaw is not modeled separately from the lower lip. Ml and Mr together can approximate the AUs 20 (horizontal lip stretcher) and 14 (dimpler), as well as what Ekman and Friesen [1978] call “oblique” actions—pulling out and up diagonally on the corners of the mouth.

Of course the ToonFace scheme provides nowhere near an exact match to the action of a human face (for which even FACS is a simplification), but that is a problem all computer graphics schemes to date have in common, to various degrees. Where the ToonFace scheme falls especially short is in facial expression involving the physics of skin contraction and excessive exertion of muscle force, and in the combinatorial explosion possible with combinations of the numerous action units included in FACS. With patience, a skilled ToonFace designer could possibly approximate FACS better than indicated here, but that would be going against its design philosophy, which is simply to get a handful of usable facial expressions relevant to multimodal dialogue, while allowing for a playful design that doesn’t get the user’s expactations up.

A1.4 Future Enhancements

The ToonFace system is primarily a research tool. As such, it is still missing a number of features that would be desirable and not too difficult to implement. For the Editor, a useful feature would for example be multiple-level UNDOs, as well as improved user interface layout. Also,

adding animation libraries to the Editor would help a designer envision what a face looks like when it moves. Currently the animator has no user interface for adjusting such things as background color, size of the face, or window. These would all make the system easier to use. Looking further along, control points allowing the nose and ears to move would extend the kinds of creatures that can be designed in the system. A feature that allowed a face to be texture-mapped onto three-dimensional shapes would of course improve the look of the system quite a bit. The control point scheme described here is easily applicable to more conventional three-dimensional computer graphics, keeping the simplicity without compromizing facial expression.

Lastly, an interesting—and useful—addition would be a mechanism to adjust the face's direction of gaze as it appears to the viewer; research has shown that factors such as face curvature, pupil placement and screen curvature interact in determining where a two-dimensional projection of a face seems to be looking, from the observer's point of view [Anstis et al. 1969]. The same would apply to head motion. This is especially important for systems that track a user's line of gaze and thus allow for reciprocal behavior from the machine.

System Specifications

APPENDIX A2

A2.1 Hardware

Following is a list of the computers used to run the Gandalf system, and which piece of software each one ran.

Perception & decisionmaking (Ymir Alpha):

- Model: Dec Alpha 3000 Model 300 AXP
- CPU: DECAlpha 21064, 64 bit RISC microprocessor
- Clock speed: 150 MHz
- Memory: 64 Mb RAM

Motor Scheduler (Ymir Alpha):

- Model: Dec 5000 Model 240
- CPU: R34000 CISC microprocessor
- Clock speed: 40 MHz
- Memory: 40 MB RAM

HARK speech recogntion:

- Model: SGI Iris Indigo
- CPU: IRIS Indigo R4000 CISC microprocessor
- Clock speed: 100 MHz
- Memory: 40 MB RAM

Intonation:

- Model: Macintosh Quadra 950
- CPU: M68040 microprocessor
- Clock speed: 33 MHz
- Memory: 81 Mb

Body model:

- Model: Packard Bell 486
- Memory: 6,5 Mb RAM

Eye tracking:

- Model: IBM 386
- Memory: 3.7 Mb RAM

Gandalf graphics:

- Model: SGI Indigo2
- CPU: MIPS R4000 Revision 3.0
- Clock speed: 100MHz
- Memory: 64 MB RAM

Solar system graphics:

- Model: Hewlet Packard Apollo 9000 series 700 model 750 workstation
- CPU: PA-RISC processor with 66-MHz floating point co-processor
- Clock speed: 66 MHz
- Memory: 128 MB RAM

A2.2 Software

A2.2.1 Main Software

- Intonation: 1,100
- Motor Scheduler: 1,600
- Knowledge Base: 2,600
- Perception & Action: 2,700
- Socket, data, miscellaneous: 4,000
- Multimodal Recorder: 1,700

Total lines of LISP code: 13,700

A2.2.2 Support software

- Bodymodel: 5000
- Speech-recogntion related: 400[1]

Total lines of C code: 5,400

1. HARK [BBN 1993] not included.

Questionnaires & Scoring

APPENDIX A3

A3.1 Scoring

A3.1.1 Contributions

Relative number of subject contributions was estimated in the following way: for each utterance or action of the agent, the number of times a subject repeats the same request or makes a new request is counted.

A3.1.2 Hesitations / Frustration

The following behaviors will be counted as constituting interaction-related hesitations and frustration, on behalf of the subjects:

1. Restarts (related to agent's behavior—no to recalling a command).
2. Subject looks at experimenter while waiting for agent to respond.
3. Subject clearly indicates frustration with agent's response or lack of response, by gesture, facial expression or verbally, for example by asking experimenter what to do.

A3.2 Instructions for Subjects

You are about to test a system that employs the latest advances in artificial intelligence and human-computer interaction. The experiment requires you wearing a suit and gloves to track your body movements, a head-mounted eye tracker that allows the computer to estimate where you are looking, and a microphone that allows the computer to hear your speech. After putting on this tracking system, you will be asked to calibrate it. Follow the spoken instructions of your administrator.

Next, a face will appear on the monitor to your left. This is your solar system expert. You will interact with three different characters. They are all experts on the solar system, but their abilities, intelligence and/or behaviors may vary. After your interaction with each of them, you will evaluate their performance. The estimated total interaction time for all 3 characters is 45 minutes.

On the big display in front of you is a computer model of the solar system. The character understands commands to go certain places in the solar system, and can tell you about the planets.

The characters are programmed to respond to natural multimodal behavior: They can "see" your body, hear your speech, and they can tell where you're looking. We ask that you try to interact with them as normally as you can. If they don't understand you, repeat your sentence or say something else. You can ask questions in any sequence you choose.

A3.3 Evaluation Questionnaire

(Notice to the reader: The name appearing in the questions, Gandalf, Bilbo or Roland, depended on which character the subject just interacted with.)

You have just interacted with Gandalf, one of three computer-enacted characters we are developing. Following are questions related to your experience with Gandalf. Please answer them to the best of your ability.

Estimated time: 5 minutes.

Your cooperation is appreciated.

1. Please rate Gandalf on the following issues:

1-a. On a scale from 0 to 10, assuming that a human gets a score of 10, Gandalf's understanding of your spoken language gets a score of _____.

1-b. On a scale from 0 to 10, assuming that a human gets a score of 10, Gandalf's use of spoken language gets a score of _____.

When two people interact face-to-face, their interaction is most of the time very smooth, with minimal hesitations and misunderstandings.

1-c. On a scale from 0 to 10, assuming that a human gets a score of 10, the smoothness of the interaction with Gandalf gets a score of _____.

1-d. Compared to interacting with a dog, Gandalf's understanding of spoken language is ...

o ...much better. o ...somewhat better. o ...about equal.
o ...slightly worse. o ...much worse

1-e. Compared to interacting with a dog, Gandalf's use of language is ...

o ...much better. o ...somewhat better. o ...about equal.
o ...slightly worse. o ...much worse.

1-f. Compared to interaction with a dog, the smoothness of the interaction with Gandalf is ...

o ...much better. o ...somewhat better. o ...about equal.
o ...slightly worse. o ...much worse.

1-g. Compared to interacting with a fish in a fishbowl, interacting with Gandalf is ...

o ...much more interesting.
o ...somewhat more interesting.
o ...about equal
o ...somewhat less interesting.
o ...much less interesting.

1-h. Compared to any real animal (excluding humans), Gandalf seems...

o ...increadibly life-like.
o ...very life-like.
o ..somewhat life-like.
o ...not very life-like.
o ...not life-like at all.

1-i. Compared to the most life-like character in any computer game or program you have seen, Gandalf seems...

o ...increadibly life-like.
o ...very life-like.
o ..somewhat life-like.
o ...not very life-like.
o ...not life-like at all.

2. If the video monitor with Gandalf's face had been turned off for the whole time, do you think your interaction with the computer would have been different?

o Yes
o Yes, perhaps, but not significantly.
o No, probably not.

2-b. If Yes, how would it be different? (Mark A or B or leave blank.)

	A		B
1.	o More fun.	OR	o Less fun.
2.	o More difficult.	OR	o Less difficult.
3.	o More efficient.	OR	o Less efficient.
4.	o Smoother.	OR	o Less smooth.

o Other:

3. How helpful to the interaction did you find ...

3-a. ...the content of Gandalf's speech?
o Very helpful.
o Somewhat helpful.
o Neither helpful nor unhelpful.
o Unhelpful.
o Counterproductive.

3-b. ...Gandalf's head motions?
o Very helpful.
o Somewhat helpful.
o Neither helpful nor unhelpful.
o Unhelpful.
o Counterproductive.

3-c. ...Gandalf's expressions?
o Very helpful.
o Somewhat helpful.
o Neither helpful nor unhelpful.
o Unhelpful.
o Counterproductive.

3-d. ...Gandalf's gaze?
o Very helpful.
o Somewhat helpful.
o Neither helpful nor unhelpful.
o Unhelpful.
o Counterproductive.

3-e. ...Gandalf's hand gestures?
- o Very helpful.
- o Somewhat helpful.
- o Neither helpful nor unhelpful.
- o Unhelpful.
- o Counterproductive.

A3.4 Prior Beliefs Questoinnaire

1. The following questions relate to your preconceived notions of interacting with computers.

1-a. Did interacting with these computer controlled characters confirm or disconfirm any of your pre-conceived notions about the difficulty / ease of interacting multimodally with a computer?

- o I had no preconceived notions of that.
- o Yes, I thought it would be....
  - o ...much easier than it was.
  - o ...somewhat easier than it was.
  - o ...somewhat smoother than it was.
  - o ...much smoother than it was.
  - o ...very much like this.
  - o ...somwhat more difficult than it was.
  - o ...much more difficult than it was.
- o No.

1-b. Did interacting with these computer controlled characters confirm or disconfirm any of your pre-conceived notions about your willingness to interact with a computer system that acts as a human?

- o I had no preconceived notions of that.
- o Yes, I am now ....
  - o ...much more willing to interact with such systems.
  - o ...somewhat more willing to interact wtih such systems.
  - o ...somwhat less willing to interact wtih such systems.
  - o ...much less willing to interact wtih such systems.
- o No, I am equally willing as I was before.

1-c. Did interacting with these computer controlled characters confirm or disconfirm any of your pre-conceived notions about what future intelligent machines might look like?

o No.
o Yes. Please explain:

1-d. Did interacting with these computer controlled characters confirm or disconfirm any of your pre-conceived notions about when/if machines will ever become intelligent?

o I had no preconceived notions of that.
o Yes, I am now ....
  o ...much more certain they will become intelligent.
  o ...somewhat more certain they will become intelligent.
  o ...somwhat less certain they will become intelligent.
  o ...much less certain they will become intelligent.
o No, I hold the exact same belief as before.

1-e. Did interacting with these computer controlled characters confirm or disconfirm any other pre-conceived notions you had before about computers, intelligent machines or humanoid computer assistants?

o No.
o Yes. Please explain:

2. Would you use an assistant like Gandalf, Bilbo or Roland in your home to help you with various tasks?

2-a. o Yes, even in its current form.
2-b. o No, not in its current form; ONLY if .... (mark all that apply:)
  o I didn't have to "dress up" to communicate with it.
  o it were a little bit smarter.
  o it were much smarter.
  o the interaction were smoother.
  o it had more knowledge (about various topics).
  o it was easier to interact with.
  o it were mobile and had a physical body like a robot.
  o it had knowledge about my favorite topic.
2-c.o No.

3. It took approximately 3 person-years to make these prototype characters (a person-year is the amount of work a person can do in one year). How many years do you think it would take a dedicated research team (10-20 researchers) to create a character such as these that works perfectly?

o more than 100 years
o between 5 & 10 years
o between 50 & 100 years
o between 2 & 5 years
o between 25 & 50 years
o between 1 & 2 years
o between 10 & 25 years
o less than 1 year

References

Adler, R. (1989). Blackboard Systems. In S. C. Shapiro (ed.), *The Encyclopedia of Artificial Intelligence*, 2nd ed., 110-116. New York, NY: Wiley Interscience.

Agre, P. E. (1985). Robotics and Common Sense. In M. Minsky (ed.), *Robotics*, 71-97. Garden City, New York: Anchor Press/Doubleday.

Agre, P. E. (1994). Review of L. A. Suchman's 'Plans and Situated Actions: The Problem of Human-Machine Communication'. In W. J Clancey, S. W. Smolinar & M. J. Stefik (eds.) *Comtemplating Minds*, 223-238. Cambridge, MA: M.I.T. Press.

Agre, P. E. & Chapman, D. (1990). What Are Plans for? In P. Maes (ed.), *Designing Autonomous Agents*, 17-34. Cambridge, MA: MIT Press.

Agre, P. & Chapman, D. (1987). Pengi: An Implementation of a Theory of Activity. *Proceedings 6th AAAI*, 268-72.

Albus, J, McCain, H & Lumia, R. (1987). NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM), Technical Report Technical Note 1235, National Bureau of Standards, Gaithersburg, Maryland.

Allen, J. (1987). *Natural Language Understanding*. Reading, MA: Benjamin/Cummings Publishing Co. Inc.

Aloimonos, Y. (ed.) (1993). *Active Perception*. Hillsdale, NJ: Lawrence Earlbaum Associates, Inc.

Anstis, S. M., Mayhew, J. W. & Morley, T. (1969). The Perception of Where a Face or Television 'Portrait' is Looking. *American Journal of Psychology*, 82, 474-89.

Arbib, M. A. (1992). Schema Theory. In S.C. Shapiro (ed.), *The Encyclopedia of Artificial Intelligence*, 2nd ed., 1427-1443. N.Y.: Wiley Interscience.

Arbib, M. A. (1994). Allen Newell, Unified Theories of Cognition (review). In W. J. Clancey, S. W. Smolinar & M. J. Stefik (eds.), *Contemplating Minds*, 21-39. Cambridge, MA: MIT Press.

Argyle, M. & Cook, M. (1976). *Gaze and Mutual Gaze*. England: Cambridge University Press.

Argyle, M. & Ingham, R. (1972). Gaze, Mutual Gaze, and Proximity. *Semiotica,* IV(1), 32-49.

Argyle, M., Lefebvre, L., & Cook, M. (1974). The meaning of five patterns of Gaze. *European Journal of Social Psychology*, 4(2), 125-136.

Austin, J. L. (1962). *How to do Things with Words*. Oxford, England: Clarendon Press.

Badler, N. I., Phillips, C. B. & Webber, B. L. (1993). *Simulating Humans: Computer Grahphics Animation and Control*. New York, New York: Oxford University Press.

Bares, J., Hebert, M., Kanade, T., Krotkov, E., Mitchell, T., Simmons, R. & Whittaker, W. (1989). Ambler: An Autonomous Rover for Planetary Exploration. *IEEE Computer*, 2 (6), June, 18-28.

Barr, A. & Feigenbaum, E. A. (eds.) (1982). *The Handbook of Artificial Intelligence*, *Ch. VIII, B-7*. Reading, MA: Addison-Wesley Publishing Co.

BBN (1993). HARK REcognizer Release 1.1 Beta. Document No. 100-1.1. Bolt, Beranek & Newman, Inc., Speech and Natural Language Processing Department.

Bers, J. (1996). A Body Model Server for Human Motion Capture and Representation. To be published in *Presence: Teleoperators and Virtual Environments*, 5(3).

Bers, J. (1995a). A Geometric Model of a User's Upper Body. M.I.T. Media Laboratory, Advanced Human Interface Group Technical Report 1-95.

Bers, J. (1995b). Direction Animated Creatures through Gesture and Speech. Master's Thesis, Media Arts and Sciences, Massachusetts Institute of Technology, Media Laboratory.

Bates, J., Loyall, A. B. & Reilly, W. S. (1992). Integrating Reactivity, Goals and Emotion in a Broad Agent. *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*, Bloomington, Indiana, July.

Blumberg, B. M. & Galyean, T. A. (1995). Multi-Level Direction of Autonomous Creatures for Real-Time Virtual Environments. *Proceedings of SIGGRAPH '95*, August, 47-54.

Bolt, R. A. & Herranz, E. (1992). Two-Handed Gesture in Multi-Modal Dialog. *Proceedings of UIST '92, Fifth Annual Symposium on User Interface Software and Technology*, Monterey, CA, November 15-18.

Bolt, R. A. (1980). "Put-That-There": Voice and Gesture at the Graphics Interface. *Computer Graphics*, 14(3), 262-70.

Bolt, R. A. (1985). Conversing With Computers. *Technology Review*, Feb./March. Cambridge, MA: MIT Press.

Bolt, R. A. (1987). The Integrated Multi-Modal Interface. *The Transactions of the Institute of Electronics, Information and Communication Engineers* (Japan), November, J79-D(11), 2017-2025.

Bond, A. H. & Gasser, L. (eds.) (1988). *Readings in Distributed Artificial Intelligence*. San Mateo, CA: Morgan Kaufman Publishers.

Brems, D. J., Rabin, M. D. & Waggett, J. L. (1995). Using Natural Language Conventions in the User Interface Design of Automatic Speech Recognition Systems. *Human Factors*, 37(2), 265-282.

Brennan, S. (1985). The Caricature Generator. *Leonardo*, 18, 170-178.

Brennan, S. (1990). Conversation as Direct Manipulation: An Iconoclastic View. In B. Laurel (ed.), *The Art of Human-Computer Interface Design*, 393-404. Reading, MA: Addison-Wesley Publishing Company.

Britton, B. C. J. (1991). Enhancing Computer-Human Interaction With Animated Facial Expressions. Master's Thesis, Massachusetts Institute of Technology. Cambridge, MA.

Brooks, R. & Stein, L. A. (1993). Building Brains for Bodies. M.I.T. Artifical Intelligence Laboratory memo No. 1439, August.

Brooks, R. (1990). Elephants Don't Play Chess. In P. Maes (ed.), *Designing Autonomous Agents*, 3-15. Cambridge, MA: MIT Press.

Brooks, R. (1991). Intelligence Without Reason. Massachusetts Institute of Technology, Artificial Intelligence Laboratory AI Memo No. 1293.

Brooks, R. A. (1986). A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robotics and Automation*, 2(1), March.

Brooks, R. A. (1990). Elephants Don't Play Chess. In P. Maes (ed.) *Designing Autonomous Agents*, 3-15. Cambridge, MA: MIT Press.

Cahn, J. (1992). An Investigation into the Correlation of Cue Phrases, Unfilled Pauses and the Structuring of Spoken Discourse. *Proceedings of the IRCS Workshop on Prosody in Natural Speech,* Technical Report IRCS-92-73, University of Pennsylvania, Institute for Research in Cognitive Science, Philadelphia, Pennsylvania, August, 19-30.

Cannon, D. (1992). Point-and-Direct Telerobotics: Interactive Supervisory Control at the Object Level in Unstructured Human-Machine System Environments. Ph.D. Thesis, Stanford University.

Capêk, K. (1920). R.U.R. English translation reprinted in M. H. Greenberg (ed.), *Great Tales of Science Fiction*, 96-143. New York, New York: Galahad Books Inc, 1988.

Card, S. K.; Moran, T. P.; & Newell, A. (1986). The Model Human Processor: An Engineering Model of Human Performance. In K. R. Boff, L. Kaufman, & J. P. Thomas (eds.), *Handbook of Human Perception Vol. II*. New York, New York: John Wiley and Sons.

Card, S. K., Moran, T. P. & Newell, A. (1983). *The Psychology of Human-Computer Interaction*. Hillsdale, New Jersey: Lawrence Earlbaum Associates.

Cassell, J., Pelachaud, C., Badler, N., Steedman, M., Achorn, B., Becket, T., Deouville, B., Prevost, S. & Stone, M. (1994a). Animated Conversation: Rule-based Generation of Faceial Expression, Gesture & Spoken Intonation for Multiple Conversational Agents. *Proceedings of SIGGRAPH '94*.

Cassell, J., Stone, M., Douville, B., Prevost, S., Achorn, B., Steedman, M., Badler, N. & Pelachaud, C. (1994b). Modeling the Interaction between Speech and Gesture. *Sixteenth Annual Conference of the Cognitive Science Society*, Atlanta, Georgia, August 13-16, 153-158.

Cassell, J. & McNeill, D. (1991). Gesture and the Poetics of Prose. *Poetics Today*, 12(3), Fall, 375-404.

Chernoff, H. (1973). The Use of Faces to Represent Points in k-Dimensional Space Graphically. *Journal of the American Statistical Association*, 68, June, 361-368.

Chin, D. N. (1991). Intelligent Interfaces as Agents. In J. W. Sullivan & S. W. Tyler (eds.), *Intelligent User Interfaces*, 177-206. New York, NY: Addison-Wesley Publishing Company.

Clark, H. H. (1992). *Arenas of Language Use*. Chicago, Illinoi: University of Chicago Press.

Clark, H. H. & Brennan, S. E. (1990). Grounding in Communication. In L. B. Resnick, J. Levine & S. D. Bahrend (eds.), *Perspectives on Socially Shared Cognition*, 127-149. American Psychological Association.

Common Lisp: The Reference (1988). Reading, Massachusetts: Addison-Wesley.

Coren, S. & Ward, L. M. (1989). *Sensation and Perception*, Third Edition. San Diego, California: Harcourt Brace Jovanovich.

Cooper, R. M. (1974). The Control of Eye Fixation by the Meaning of Spoken Language: A New Methodology for the Real-Time Investigation of Speech Perception, Memory, and Language Processing. *Cognitive Psychology*, 6, 84-107.

Crowston, K. & Malone, T. W. (1988). Intelligent Software Agents. *Byte*, December, 13(13), 267-272.

Crowston, K., Malone, T. W. & Lin, F. (1988). Cognitive Science and Organizational Design: A Case Study of Computer Conferencing. *Human-Computer Interaction*, 3, 59-85.

Cutkosky, M. R. (1992). Robot Hands and End-Effectors. In S.C. Shapiro (ed.), *The Encyclopedia of Artificial Intelligence*, 2nd ed., 1357-1375. N.Y.: Wiley Interscience.

Davis, R. & Smith, R. G. (1988). Negotiation as a Metaphor for Distributed Problem Solving. In A. H. Bond & L. Gasser (eds.), *Readings in Distributed Artificial Intelligence*, 333-356. San Mateo, CA: Morgan Kaufman Publishers.

Dean, T. & Boddy, M. (1987). An Analysis of Time-Dependent Planing. *Proceedings fo the 10th International Joint Conference on Artificial Intelligence*, 49-54. San Mateo, California: Morgan-Kaufman.

Dectalk (1985). Dectalk and DTC03 text-to-speech system owner's manual. Bedford, MA: Digital Equipment Corporation.

Dennett, D. C. (1978). *Brainstorms: Philosophical Essays on Mind and Psychology.* Cambridge, Massachusetts: MIT Press.

Depp, S. W. & Howard, W. E. (1995). Flat-Panel Displays: Recent Advances in Microelectronics and liquid crystals make possible video screens that can be hung on a wall or warn on a wrist. *Scientific American: The Computer in the 21st Century*, Special Issue, Vol 6(1), 1995, 70-75. Originally published in *Scientific American*, March, 1993.

Dickerson, J. A. & Kosko, B. (1994). Virtual Worlds as Fuzzy Cognitive Maps. *Presence: Teleoperators and Virtual Environments*, 3(2), Spring, 173-189.

Dodhiawala, R. T. (1989). Blackboard Systems in Real-Time Problem Solving. In Jagannathan, V., Dodhiawala, R. & Baum, L. S. (eds.), *Blackboard Architectures and Applications*, 181-191. Boston: Academic Press, Inc.

Duncan, S. Jr. (1972). Some Signals and Rules for Taking Speaking Turns in Conversations. *Journal of Personality and Social Psychology*, 23(2), 283-292.

Effron, D. (1941/1972). *Gesture, Race and Culture*. The Hague: Mouton.

Ekman, P. (1979). About Brows: Emotional and Conversational Signals. In M. vonCranach, K. Foppa, W. Lepenies, & D. Ploog (eds.), *Human Ethology*, 169-249.

Ekman, P. & Friesen, W. (1975). *Unmasking the Face*. New Jersey: Prentice-Hall.

Ekman, P. & Friesen, W. (1978). *Facial Action Coding System*. Palo Alto, CA: Consulting Psychologists Press.

Ekman, P. & Friesen, W. (1974). Nonverbal Leakage and Clues to Deception. In S. Weitz (ed.), *Nonverbal Communication*, 269-290. New York, NY: Oxford University Press.

Ekman, P. & Friesen, W. (1969). The Repertoire of Non-Verbal Behavior: Categories, Origins, Usage, and Coding. *Semiotica*, 1, 49-98.

Engelmore, R. & Morgan, T. (eds.)(1988). *Blackboard Systems*. Wokingham, England: Addisson-Wesley.

Essa, I. A. (1995). Analysis, Interpretation and Synthesis of Facial Expressions. Ph.D. Thesis, Media Arts and Sciences, Massachusetts Institute of Technology. M.I.T. Media Laboratory Perceptual Computing Technical Report #303.

Essa, I. A., Darrell, T. & Pentland, A. (1994). Modeling and Interactive Animation of Facial Expression using Vision. M.I.T. Media Laboratory Perceptual Computing Section Technical Report No. 256.

Fehling, M. R., Altman, A. M. & Wilber, B. M. (1989). The Heuristic Control Virtual Machine: An Implementation of the Schemer Computational Model of Reflective, Real-Time Problem-Solving. In Jagannathan, R. Dodhiawala & L. S. Buam, *Blackboard Architectures and Applications*, 191-218. Boston: Academic Press, Inc.

Forbidden Planet (1956). Motion picture. F. M. Wilcox (dir.), MGM.

Gibson, J. J. (1979). *The Ecological Approach to Visual Perception*. Boston, Massachusetts: Houghton-Mifflin.

Gibson, J. J. & Pick, A. D. (1963). Perception of Another Person's Looking Behaivor. *American Journal of Psychology*, 46, 386-94.

Goodwin, C. (1981). *Conversational Organization: Interaction Between Speakers and Hearers*. New York, NY: Academic Press.

Goodwin, C. (1986). Gestures as a Resource for the Organization of Mutual Orientation. *Semiotica*, 62(1/2), 29-49.

Goodwin, M. H. & Goodwin, C. (1986). Gesture and Coparticipation in the Activity of Searching for a Word. *Semiotica*, 62(1/2), 51-75.

Grice, H. P. (1989). *Studies in the Way of Words*. Cambridge, Massachusetts: Harvard University Press.

Grosz, B. J. & Sidner, C. L. (1986). Attention, Intentions, and the Strucutre of Discourse. *Computational Linguistics*, 12(3), 175-204.

Hamm, J. (1967). *Cartooning the Head and Figure*. New York, NY: The Putnam Publishing Group.

Hasegawa, O., Itou, K., Kurita, T., Hayamizu, S., Tanaka, K., Yamamoto, K. and Otsu, N. (1995). *Active Agent Oriented Multimodal Interface System*. Proceedings of IJCAI, 82-87.

Held, R. M. & Durlach, N. I. (1992). Telepresence. *Presence: Teleoperators and Virtual Environments*, 1(1), 109-12.

Hauptman, A. G. (1989). Speech and Gestures for Graphic Image Manipulation. *Proceedings of SIGCHI*, May, 241-5.

Hayes-Roth, B., Hayes-Roth, F., Rosenschein, S. & Cammarata, S. (1988). Modeling Planning as an Incremental, Opportunistic Process. In R. Engelmore & T. Morgan, *Blackboard Systems*, 231-245. Reading, MA: Addison-Wesley Publishing Co.

Huberman, B. A. (ed.)(1988). *The Ecology of Computation*. Amsterdam: North-Holland.

Huhns, M. (ed.) (1987). *Distributed Artificial Intelligence*. San Mateo, CA: Morgan-Kaufmann.

Jagannathan, V., Dodhiawala, R. & Baum, L. S. (eds.)(1989). *Blackboard Architectures and Applications*. Boston: Academic Press, Inc.

Kacprzyk, J. (1992). Fuzzy Sets and Fuzzy Logic. In S.C. Shapiro (ed.), *The Encyclopedia of Artificial Intelligence*, 2nd ed., 537-542. N.Y.: Wiley Interscience.

Kahneman, D. (1973). *Attention and Effort*. New Jersey: Prentice-Hall, Inc.

Kay, A. (1984). Computer Software. *Scientific American*, 251(3), Sept., 53-59.

Keene, S. E. (1989). *Object-Oriented Programming in Common Lisp: A Programmer's Guide to CLOS*. Reading, Massachusetts: Addison-Wesley Publishing Co.

Kendon, A. (1980). Gesticulation and Speech: Two Aspects fo the Process of Utterance. In M. R. Key (ed.), *The Relation Between Verbal and Non-Verbal Communication*. Mouton: The Hague.

Kernigan, B. W. & Ritchie, D. M. (1988). *The C Programming Lanugage*, second ed. Englewood Cliffs, New Jersey: Prentice Hall.

King, W. J. & Ohya, J. (1996). The Representation of Agents: Anthropomorphism, Agency and Intelligece. *Proceedings of SIGCHI '96*, 289-90.

Kleinke, C. (1986). Gaze and Eye Contact: A Research Review. *Psychological Bulletin*, 100(1), 78-100.

Koons, D. B., Sparrell, C. J. & Thórisson, K. R. (1993). Integrating Simultaneous Input from Speech, Gaze and Hand Gestures. Chapter 11 in M. T. Maybury (ed.), *Intelligent Multi-Media Interfaces*, 252-276. Cambridge, MA: AAAI Press/M.I.T. Press.

Koons, D. B. & Thórisson, K. R. (1993). Estimating Direction of Gaze in Multi-Modal Context. Presented at *3CYBERCONF—The Third International Conference on Cyberspace*, Austin TX, May 13-14, 14 pp.

Koren, J. U. (1984). *A Geometric Investigation of Reach*. Cambridge, Massachusetts: The MIT Press.

Kosslyn, S. M. & Koenig, O. (1992). *Wet Mind: The New Cognitive Neuroscience*. New York, New York: The Free Press.

Lanier, J. (1995). Agents of Alienation. *ACM Interactions*, july, 66-72.

Lasseter, J. (1987). Principles of Traditional Animation Applied to 3D Computer Animation. *Computer Graphics*, 21(4), 35-44.

Laurel, B. (1990). Interface agents: Metaphors with character. In B. Laurel (ed.) *The Art of Human-Computer Interface Design*, 355-365. Reading, MA: Addison-Wesley Publishing Co.

Laurel, B. (1992). Anthropomorphism: From Eliza to Terminator 2. Panelist discussion, A. Don (moderator). *Proceedings of SIGCHI '92*, 67-70.

Laurel, B., Oren, T. & Don, A. (1990). Issues in Multimedia Interface Design: Media Integration and Interface Agents. *Proceedings of SIGCHI '90*, 133-9, April 1-5, Seattle, Washington.

Lawless, J. A. & Miller, M. M. (1991). *Understanding CLOS, the Common Lisp Object System*. Cambridge, Massachusetts: Digital Press.

Librande, S. (1992). Example-Based Character Drawing. Master's Thesis, Massachusetts Institute of Technology. Cambridge, MA.

Lord, C. & Haith, M. M. (1974). The Perception of Eye Contact. *Perception & Psychophysics*, 16(3), 413-16.

Lyons, D. M. & Hendriks, A. J. (1992). Planning, Reactive. In S.C. Shapiro (ed.), *The Encyclopedia of Artificial Intelligence*, 2nd ed., 1171-1181. New York, New York: Wiley Interscience.

Macintosh Common Lisp Programming Guide. Apple Computer Inc. 1990.

Maes, P. (1994). Agents that Reduce Work and Information Overload. *Communications of the ACM*, July 1994, vol. 37(7), 31-40, 146.

Maes, P. (ed.) (1990a). *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*. Cambridge, MA: MIT Press/Elsevier.

Maes, P. (1990b). Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back. In P. Maes (ed.), *Designing Autonomous Agents*, 1-2. Cambridge, MA: MIT Press.

Maes, P. (1990c). Situated Agents can have Goals. In P. Maes (ed.), *Designing Autonomous Agents*, 49-70. Cambridge, MA: MIT Press.

Maes, P. (1989). How to Do the Right Thing. A.I. Memo No. 1180, December, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Maes, P., Darrell, T., Blumberg, B. & Pentland, A. (1995). The ALIVE System: Full-body Interaction with Autonomous Agents. *IEEE Computer,* Special Issue on Virtual Environments, 11-18.

Maes, P., Darrell, T., Blumberg, B. & Pentland, A. (1994). Interacting with Animated Autonomous Agents. *AAAI Spring Symposium on Believable Agents Working Notes*, Stanford University, California, March 19-20, 50-54.

Maes, P. & Kozierok, R. (1993). Learning Interface Agents. *Proceedings of AAAI '93*, 459-465.

Malone, T. W. & Crowston, K. (1991). Toward an Interdisciplinary Theory of Coordination. Center for Coordination Science Working Paper #120, Massachusetts Institute of Technology, Cambridge, Massachusetts.

Malone, T. W., Yates, J. & Benjamin, R. I. (1988). Electronic Markets and Electronic Hierarchies. In I. Greif (ed.), *Computer-Supported Cooperative Work: A Book of Readings*, 581-607. San Mateo, CA: Morgan Kaufman Publishers.

Maulsby, D., Greenberg, D. & Mander, R. (1993). Prototyping an Intelligent Agent through Wizard of Oz. *Proceedings of InterCHI '93*, 277-84, April 24-29, Amsterdam.

Mayer, J. A. & Wilson, S. (eds.) (1991). *From Animals to Animats*. Cambridge, MA: MIT Press.

McNeill, D. & Levy, E. (1982). Conceptual Representations in Language Activity and Gesture. In R. J. Jarvella & W. Klein (eds.), *Speech, Place, and Action*, 271-295. New York, NY: John Wiley & Sons Ltd.

McNeill, D. (1992). *Hand and Mind: What Gestures Reveal about Thought*. Chicago, IL: University of Chicago Press.

Metropolis (1925). Motion picture. F. Lange (dir.). Paramount.

MIDI: Musical Instrument Digital Interface Specification 1.0. International MIDI Association, North Hollywood, California, 1983.

Minsky, M. (1985). *The Society of Mind*. New York: Simon & Schuster.

Minsky, M. & Riecken, D. (1994). An Interview with Marvin Minsky about Agents. *Communications of the ACM*, July, Vol. 37(7), 23-29.

Mitchell, T., Caruana, R., Freitag, D., McDermott, J. & Zabowski, D. Experience with a Learning Personal Assistant. *Communications of the ACM*, July 1994, vol. 37(7), 81-91.

Nagao, K. & Takeuchi, A. (1994). Social Interaction: Multimodal Conversation with Social Agents. *Proceedings of the 12th National Conference on Artificial Intelligence* (AAAI-94), vol. 1, 22-28.

NASA Tech Briefs (1995). Cuff-Mounted Electronic Checklis Display Unit. *NASA Tech Briefs: The Digest of New Technology*, July, 19(7), p. 44.

NASA Tech Briefs (1995b). Synthesis of Realistic Animations of a Person Speaking. *NASA Tech Briefs: The Digest of New Technolgy*, August, 19(8), p. 72.

Nass, C., Steuer, J. & Tauber, E. R. (1994). Computers are Social Actors. *Proceedings of SIGCHI '94*, Boston, MA, April 24-28, 72-78.

Nass, C. Steuer, J. Tauber, E. & Reeder, H. (1993). Antrhopomorphism, Agency & Ethopoeia: Computers as Social Actors. *InterCHI '93 Adjunct Proceedings*, 111-112.

Neal, J. G. & Shapiro, S. C. (1991). Intelligent Multi-Media Interface Technology. In J. W. Sullivan & S. W. Tyler (eds.), *Intelligent User Interfaces*, 11-45. New York, NY: Addison-Wesley Publishing Company.

Negroponte, N. (1990). Hospital Corners. In B. Laurel (ed.) The Art of Human-Computer Interface Design, 347-353. Reading, MA: Addison-Wesley Publishing Co.

Neider, J., Davis, T. & Woo, M. (1993). *Open GL Programming Guide*. Reading, Massachusetts: Addison-Wesley.

Nespolous, J-L & Lecours, A. R. (1986). Gestures: Nature and Function. In J-L Nespolous, P. Perron & A. R. Lecours (eds.), *The Biological Foundations of Gestures: Motor and Semiotic Aspects*, 49-62. Hillsdale, NJ: Lawrence Earlbaum Associates.

Newell, A. (1990). *Unified Theories of Cognition.* Cambridge, MA: Harvard University Press.

Nii, P. (1989). Blackboard Systems. In A. Barr, P. R. Cohen & E. A. Feigenbaum (eds.), *The Handbook of Artificial Intelligence*, Vol. IV, 1-74. Reading, MA: Addison-Wesley Publishing Co.

Ochsman, R. B. & Chapanis, A. (1974). The Effects of 10 Communication Modes on the Behavior of Teams During Co-operative Problem-solving. *Int. J. Man-Machine Studies*, 6, 579-619.

O´Conaill, B, Whittaker, S. & Wilbur, S. (1993). Conversations Over Video Conferences: An Evaluation of the Spoken Aspects of Video-Mediated Communication. *Human-Computer Interaction*, 8, 389-428.

Oren, T., Salomon, G., Kreitman, K. & Don, A. (1990). Guides: Characterizing the Interface. In B. Laurel (ed.) *The Art of Human-Computer Interface Design*, 367-81. Reading, MA: Addison-Wesley Publishing Co.

Pelachaud, C., Badler, N. I. & Steedman, M. (1996). Generating Facial Expressions for Speech. *Cognitive Science*, 20 (1), 1-46.

Pelachaud, C., Badler, N. I. & Steedman, M. (1991). Linguistic Issues in Facial Animation. N. M. Thalmann & D. Thalmann (Eds.), *Computer Animation '91*, 15-30. Tokyo: Springer Verlag.

Pentland, A., Starner, T. Etchoff, N., Masoiu, A., Oliyide, O. & Turk, M. (1993). Experiments with Eigenfaces. *Looking at People Workshop, IJCAI '93*, August. Chamberry, France.

Pierrehumbert, J. & Hirschberg, J. (1990). The Meaning of Intonational Contours in the Interpretation of Discourse. In P. R. Cohen, J. Morgan & M. E. Pollack (eds.), *Intentions in Communication.* Cambridge: MIT Press.

Poyatos, F. (1980). Interactive Functions and Limitations of Verbal and Nonverbal Behaviors in Natural Conversation. *Semiotica*, 30-3/4, 211-244.

Prevost, S. (1996). A Semantics of Contrast and Information Structure for Specifying Intonation in Spoken Language Generation. Ph.D. Thesis, Faculty of Coputer and Information Science, University of Pennsylvania.

Prevost, S. & Steedman, M. (1994). Specifying Intonation from Context for Speech Synthesis. *Speech Communication*, 15, 139-153.

Rayner, K. (1984). Visual Selection in Reading, Picture Perception, and Visual Search: A Tutorial Review. In H. Bouma & D. G. Bouwhuis (eds.), *Attention and Performance X: Control of Language Processes*, 67-96. London: Lawrence Earlbaum Associates.

Red Dwarf (1988). Television series. B. Grant & D. Naylor (auth.). British Broadcasting Corporation.

Reddy, Y. V., Erman, L. D. & Neely, R. B. (1973). A Model and a System for Machine Recognition of Speech. *IEEE Transactions on Audio and Electro-Acoustics*, AU-21, 229-238.

Rich, C., Waters, R. C., Strohecker, C., Schabes, Y., Freeman, W. T., Torrance, M. C., Golding, A. R. and Roth, M. (1994). Demonstration of an Interactive Multimedia Environment. *IEEE Computer*, 27(12), Dec., 15-22.

Riesberg, D., Scheiber, R. & Potemken, L. (1981). Eye Position and the Control of Auditory Attention. *Journal of Experimental Psychology: Human Perception and Performance*, 7(2), 318-323.

Rimé, B. & Schiaratura, L. (1991). Gesture and Speech. In R. S. Feldman & B. Rimé, *Fundamentals of Nonverbal Behavior*, 239-281. New York: Press Syndicate of the University of Cambridge.?

Rosenbaum, D. A. & Kirst, H. (1992). Antecedents of Action. In H. Heuer & S. W. Keele (eds.), *Handbook of Motor Skills*. New York, NY: Academic Press.

Rosenbaum, D. A., Slotta, J. D., Vaughan, J. & Plamondon, M. J. (1991). Optimal Movement Selection. *Psychological Science*, 2, 86-91.

Sabiston, W. (1991). Extracting 3-D motion from Hand-Drawn Animated Figures. Master's Thesis, Media Arts and Sciences, Massachusetts Institute of Technology.

Sacks, H. (1992a). *Lectures on Conversation, vol I.* Cambridge, MA: Blackwell.

Sacks, H. (1992b). *Lectures on Conversation, vol II.* Cambridge, MA: Blackwell.

Sacks, H., Schegloff, E. A.. & Jefferson, G. A. (1974). A Simplest Systematics for the Organization of Turn-Taking in Conversation. *Language*, 50, 696-735.

Seth, B. & Maes, P. (1993). Evolving Agents for Personalized Information Retrieval. *Proceedings of the Ninth IEEE Conference on Artificial Intelligence for Applications*.

Schank, R. (1990). *Tell Me a Story*. New York, NY: Charles Scribner's Sons.

Schank, R. & Abelson, R. (1977). *Scripts, Plans, Goals and Understanding*. Hillsdale, NJ: Lawrence Earlbaum Associates.

Schegloff, E. A. & Sacks, H. (1973). Opening up Closings. *Semiotica*, 8, 289-327.

Schmandt, C., Arons, B. & Simmons, C. (1985). Voice Interaction in an Integrated Office and Telecommunications Environment. *Proceedings of 1985 Conference, American Voice Input/Output Society, AVIOS*, Palo Alto, CA, pp. 51-56.

Searle, J. R. (1975). A Taxonomy of Illocutionary Acts. In K. Gunderson (ed.), *Language, Mind and Knowledge*, 344-69. Minneapolis: University of Minesota Press.

Searle, J. R. (1971). What is a Speech Act? In J. R. Searle (ed.) *The Philosophy of Language*. London: Oxford University Press.

Searle, J. R. (1969). *Speech Acts: An Essay in the Philosophy of Language*. Cambridge: Cambridge University Press.

Selfridge, O. (1959). Pandemonium: A Paradigm for Learning. *Proceedings of Symposium on the Mechanization of Thought Processes*, 511-29.

Seu, J., Chang, R., Li, J., Evens, M., Michael, J. & Rovik, A. (1991). Language Differences in Face-to-Face and Keyboard-to-Keyboard Tutoring Sessions. *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*, August, 576-80.

Sheridan, T. B. (1992). *Telerobotics, Automation and Human Supervisory Control*. Cambridge, MA: MIT Press.

Silverstein, M. (1987). The Three Faces of Function. In M. Hickman (ed.), *Social and Functional Approaches to Language and Thought*, 17-38. Orlando, FL: Academic Press.

Sommer, R. (1959). Studies in Personal Space. *Sociometry*, 23, 247-260.

Sparrell, C. J. (1993). Coverbal Iconic Gesture in Human-Computer Interaction. Master's Thesis, Massachusetts Institute of Technology. Cambridge, Massachusetts.

Sparrell, C. J. & Koons, D. B. (1994). Capturing and Interpreting Coverbal Depictive Gestures. *AAAI 1994 Spring Symposium Series*, Stanford, USA, March 21-23, 8-12.

Star Wars (1977). Motion picture. G. Lucas (dir.). 20th Century Fox.

Starker, I. & Bolt, R. A. (1990). A Gaze-Responsive Self-Disclosing Display. *Proceedings of SIGCHI '90*, Seattle, Washington, April 1-5, 3-9.

Steele, G. L. (1990). *Common Lisp the Language*, second ed. Cambridge, Massachusetts: Digital Press.

Steels, L. (1990). Cooperation Between Distributed Agents Through Self-Organization. In Y. Demazeau & J. P. Müller (eds.), *Decentralized A. I.* Amsterdam: Elsevier Science Publishers B. V. (North-Holland).

Stroustrup, B. (1991). *The C++ Programming Language*. Reading, Massachusetts: Addison Wesley.

Sturluson, S. (1300~1325). *Edda*. Á. Björnsson prepared for publication. Iðunn, Reykjavík, 1975.

Takeuchi, A. & Nagao, K. (1993). Communicative Facial Displays as a New Conversational Modality. *Proceedings of InterCHI*, Amsterdam, April, 187-193.

Tanimoto, S. L. (1987). *The Elements of Artificial Intelligence*. Rockville, MD: Computer Science Press, Inc.

Thomas, F. & Johnston, O. (1981). *Disney Animation—The Illusion of Life*. New York, N.Y: Abbeville Press.

Thorpe, C. E. (1992). Robots, Mobile. In S.C. Shapiro (ed.), *The Encyclopedia of Artificial Intelligence*, 2nd ed., 1409-1416. New York, New York: Wiley Interscience.

Thórisson, K. R. (1996). ToonFace: A System for Creating and Animating Interactive Cartoon Faces. M.I.T. Media Laboratory Learning and Common Sense Section technical report 96-01, 13 pp.

Thórisson, K. R. (1995a). Multimodal Interaction with Humanoid Computer Characters. *Conference on Lifelike Computer Characters,* Snowbird, Utah, September 26-29, p. 45 (Abstract).

Thórisson, K. R. (1995b). Computational Characteristics of Multimodal Dialouge. AAAI Fall Symposium Series on Embodied Lanugage and Action, November 10-12, Massachusetts Institute of Technology, Cambridge, 102-108.

Thórisson, K. R. (1994). Face-to-Face Communication with Computer Agents. *AAAI Spring Symposium on Believable Agents Working Notes*, Stanford University, California, March 19-20, 86-90.

Thórisson, K. R. (1993). Simulated Gestalt Perception: An Applications to Human-Computer Interaction. *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society*, Atlanta, Georgia, August 13-16, 876-881.

Thórisson, K. R. (1993a). Dialogue Control in Social Interface Agents. *InterCHI Adjunct Proceedings '93*, Amsterdam, April, 139-140.

Thórisson, K. R. (1993b). Social Interface Agents. General Examination Written Requirement, Media Arts and Sciences, Massachusetts Institute of Technology, June 30.

Thórisson, K. R., Koons, D. B. & Bolt, R. A. (1992). Multi-Modal Natural Dialogue. *SIGCHI Proceedings '92*, April, 653-4. Also in *SIGGRAPH Video Review, CHI '92 Technical Video Program*, 76(1).

Tolkien, J. R. R. (1937). *The Hobbit*. Ballantine, 1986.

Tufte, E. R. (1990). *Envisioning Information*. Cheshire, Conecticut: Graphics Press.

Tufte, E. R. (1983). *The Visual Display of Quantitative Information*. Cheshire, Connecticut: Graphics Press.

Turk, M. & Pentland, A. (1991). Eigenfaces for Recognition. *Journal of Cognitive Neuroscience*, 3(1), 71-86.

2001: A Space Oddyssey (1968). Motion picture. S. Kubrik (dir.). Turner Entertainment Company.

Tyler, S. W., Schlossberg, J. L., & Cook, L. K. (1991). CHORIS: An Intelligent Interface Architecture for Multimodal Interaction. *AAAI '91 Workshop Notes*, 99-106.

Wahlster, W. (1991). User and Discourse Models for Multimodal Communication. In J. W. Sullivan & S. W. Tyler (eds.), *Intelligent User Interfaces*, 45-67. New York, New York: ACM Press, Addison-Wesley Publishing Company.?

Waite, C. T. (1989). The facial action control editor, FACE: A parametric facial expression editor for computer generated animation. Master's Thesis, Media Arts and Sciences, Massachusetts Institute of Technology.

Walker, M. A. (1989). Natural Language in a Desktop Environment. *Proceedings of SIGCHI '98.*

Walker, M. & Whittaker, S. (1990). Mixed Initiative in Dialogue: An Investigation into Discourse Segmentation. *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*.

Wang, M. Q. & Hirschberg, J. (1992). Automatic Classification of Intonational Phrase Boundaries. *Computer Speech and Language*, 6(2), April, 175-196.

Waters, K. (1990). Modeling 3D Facial Expressions. *ACM SIGGRAPH '90 Course Notes*, State in the Art in Facial Animation, 108-129.

Wexelblatt, A. (1994). A Feature-Based Approach to Continuous-Gestures Analysis. Master's Thesis, Media Arts and Sciences, Massachusetts Institute of Technology.

Whittaker, S. (1994). A Communication Framework for Mediated Interaction. Lotus Development Corporation Technical Report, Feb., 26 pp.

Whittaker, S., Brennan, S. E. & Clark, H. H. (1991). Co-ordinated Activity: An Analysis of Interaction in Computer-Supported Co-operative Work. *Proceedings of Conference on Computer Human Interaction*, 361-367.

Whittaker, S. & O'Conaill (1993). An Evaluation of Video Mediated Communication. *InterCHI Adjunct Proceedings*, Amsterdam, April, 73-4.

Whittaker, S. & Stenton, P. (1988). Cues and Control in Expert-Client Dialogues. *Proc. 26th Annual Meeting of the Association of Computational Linguistics*, 123-130.

Whittaker, S., & Walker, M. A. (1991). Toward a Theory of Multi-Modal Interaction. *AAAI '91 Workshop Notes*, 78-85.

Wilson, S. W. (1991). The Animat Path to AI. In J-A. Meyer & S. W. Wilson (eds.), *From Animals to Animats*. Cambridge, MA: MIT Press.

Yarbus, A. L. (1967). *Eye Movements and Vision*. New York, NY: Plenum Press.

Yngve, V. H. (1970). On Getting a Word in Edgewise. *Papers from the Sixth Regional Meeting*., Chicago Linguistics Society, 567-78.

Zimmer, C. (1995). Early Signifiers. *Discover*, May.