Continuous Uncertain Interaction

John Williamson

DOCTOR OF PHILOSOPHY
DEPARTMENT OF COMPUTING SCIENCE
FACULTY OF INFORMATION AND MATHEMATICAL SCIENCES
UNIVERSITY OF GLASGOW

2006

ABSTRACT

This work describes a novel perspective on the theoretical foundation of human-computer interfaces, framing the problem as a continuous control process. In this view, the system continuously infers a distribution over potential user goals, and provides continuous feedback about its beliefs as it does so. The proper representation and manipulation of uncertainties in interaction – via probability theory – and the explicit inclusion of temporal characteristics – in the form of dynamic systems – are inherent to this framework.

The framework is used to derive a novel approach to interaction design, particularly in situations where rich or unusual sensing and display modalities are present. A number of key tools for describing and implementing systems which are consistent with this perspective are presented. The role of system dynamics as a mediating element between sensed state and decision making is described. The work sets out a paradigm for interaction which brings probabilistic models – and thus many of the techniques of modern machine learning – into the interface in a clean and principled manner.

The three major techniques for supporting the paradigm outlined in the thesis are: the display of changing probabilistic beliefs; dynamically adjusting system handling qualities according to an inference model; and a general probabilistic selection technique based on the detection of control.

Methods are presented for displaying the state of a system with appropriate representation of uncertainty, via Monte Carlo sampling techniques. Specifically, the use of granular synthesis for auditory display of the distributions involved in a period of interaction is described, and it is shown how predictive elements can be introduced into goal directed displays, mitigating delays present in the interaction loop. The use of these techniques in displaying particle filtering processes is illustrated.

The process by which the results of goal inference can be fed back into the dynamics that the user directly controlled is presented in general form, and applied specifically to the problem of text entry. It is shown that this inference feedback mechanism unifies a range of conventional techniques, including semantic pointing and bubble pointing. A novel text entry system for mobile devices augmented with inertial sensors is developed to illuminate the inference feedback technique.

By viewing human behaviour as a control process, a general, dynamic selection for many possible sensors is developed. This is fully probabilistic and widely applicable in many domains. It provides a sound method for designing the dynamics of an interactive selection system. This is extended to general exploration in high-dimensional spaces via controlled Markov Chain Monte Carlo processes.

ACKNOWLEDGEMENTS

Many people contributed, directly and indirectly, to the production of this work. The following list of acknowledgements is, as is always the case, incomplete. I apologize to those who I have inevitably forgotten to name.

The various collaborators I have worked with have enlarged the scope of my work, and provided many interesting applications to work upon. I'd especially like to thank Thomas Hermann and his colleagues at Bielefeld University who graciously hosted me for a short term scientific mission; my collaborators at the Intelligent Sound Group at DTU; and the members of the BCI group at Fraunhofer First. I also acknowledge the contributions of the various visting collaborators with whom I had an opportunity to work with: Jostien Hansen, Thomas Hermann, Vuokko Lantz, James Kelly, Kristjan Azman and Morten Proschowsky, all of whom brought many new thoughts with them, and were great fun, to boot.

EPSRC project GR/R98105 "Audioclouds" funded me during the course of this PhD. Their support is appreciated. Note should be made of the long-suffering residents of room F101 who put up with me patiently for several years: Ali, Agathe, Greg, Jian-Qing, Malcolm, Julie, Louise, Jose, Paul and the assorted transient visitors – thanks for your support. Andrew Ramsay wrote some very useful code for communications over Bluetooth, as well as the drivers for the MESH, which made my life very much easier. The other researchers in the Dynamics and Interaction group – Parisa Eslambolchilar, Steve Strachan and Andy Crossan – were the source of many fruitful discussions, and with all of whom succesful collaborations were had. Stephen Brewster provided useful input on the work presented here. My parents provided much appreciated support during this PhD (and also proof read several parts of the thesis).

Alistair Morrison, Andy Crossan, Steve Strachan, Parisa Eslambolchilar, Rod Murray-Smith and Jim Williamson proof read drafts of this work; their efforts removed many mistakes and greatly improved the clarity of the work.

Finally, I'd like to thank Rod Murray-Smith, who supervised me over the course of this PhD, and provided an unending source of inspirational ideas in countless discussions, as well as keeping things on track throughout the process.

DECLARATION

I declare that this thesis was composed by myself, and that the work contained herein is my own except where explicitly stated otherwise in the text. This work has not been submitted for any other degree or professional qualification except as specified.

(*John Williamson*)

CONTRIBUTING PUBLICATIONS

J. Williamson, S. Strachan, R. Murray-Smith, It's a Long Way to Monte-Carlo: Probabilistic Displays in GPS Navigation, *Proceedings of Mobile HCI 2006*, Helsinki, 2006

B. Blankertz, G. Dornhege, M. Krauledat, M. Schroder, J. Williamson, R. Murray-Smith, K-R. Muller, The Berlin Brain-Computer Interface Presents the Novel Mental Typewriter Hex-o-Spell, *3rd International BCI Workshop and Training Course 2006*, Graz, 2006

R. Kamnik, T. Bajd, J. Williamson, R. Murray-Smith, Rehabilitation Robot Cell for Multimodal Standing-Up Motion Augmentation, *International Conference on Robotics and Automation* (ICRA 2005), Barcelona, 2005.

J. Williamson, R. Murray-Smith, Sonification of Probabilistic Feedback through Granular Synthesis, *IEEE Multimedia*, Vol. 12, No. 2, p45-52, April/June, 2005.

J. Williamson, R. Murray-Smith, Dynamics and probabilistic text entry, *Proceedings of the Hamilton Summer School on Switching and Learning in Feedback systems*, Eds. R. Murray-Smith, R. Shorten, Springer-Verlag, Lecture Notes in Computing Science, Vol. 3355, p333-342, 2005.

P. Eslambolchilar, J. Williamson, R. Murray-Smith, Multimodal Feedback for tilt controlled Speed Dependent Automatic Zooming, *UIST 2004*, Santa Fe, 2004.

A. Crossan, J. Williamson, R. Murray-Smith, Haptic Granular Synthesis: Targeting, Visualisation and Texturing, *International Symposium on Non-visual and Multimodal Visualization*, London, IEEE Computer Society, p527-532, 2004.

J. Williamson, R. Murray-Smith, Granular synthesis for display of time-varying probability densities, *International Workshop on Interactive Sonification (Human Interaction with Auditory Displays)*, Bielefeld, Germany, January 2004.

J. Williamson, R. Murray-Smith, Pointing without a pointer, *CHI*, Vienna, p1407-1410, 2004.

J. Williamson, R. Murray-Smith, Audio feedback for gesture recognition, *DCS Technical Report TR-2002-127*, Department of Computing Science, Glasgow University, 2002.

TABLE OF CONTENTS

List of Figures

CHAPTER I

INTRODUCTION

Motivation, themes and structure.

I.1 SUMMARY

THIS introductory chapter briefly discusses some of the problems with conventional interaction design, paying particular attention to the gesture recognition problem; argues the need for more robust theoretical models of interaction; summarises the major themes that will run through this thesis, highlighting the overarching issues of uncertainty and temporal structure; and finally presents an outline of the thesis.

I.2 THE FLAWS OF EXISTING INTERFACE DESIGNS

This thesis sets out to develop new theoretical frameworks for human-computer interaction.

The next sections describe the limitations of interfaces in current use, and motivates the principles which will form the core arguments of this work.

I.2.1 CONVENTIONAL GRAPHICAL USER INTERFACES

Human-computer interaction has passed through a number of major stages in its evolution. From the earliest punch card machines, through keyboard-based command lines, to the now-dominant graphical user interface (Figure I.1), each step in this progression more tightly-coupled the interaction to the user, reducing the delays between action and response. Modern graphical user interfaces have been tuned and refined over a long period of time, and are consequently well adapted for use. However, they are *ad hoc* solutions, often piling fixes upon basic design elements to improve interaction quality. These adaptions have led to systems well suited to the desktop interaction scenario (where interaction has traditionally taken place), with standardised sensing and display technologies, but given changes in the context of use and the availability of new interaction modalities, it is hard to adapt such interfaces to account for the consequently changed communication channel properties.

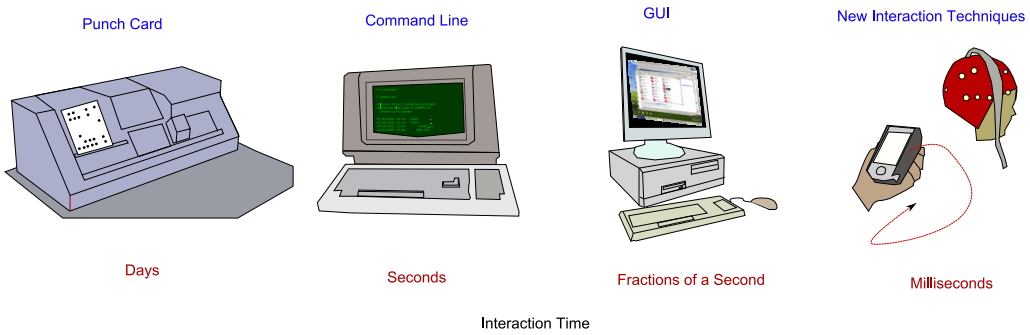


FIGURE I.1: The evolution of the human-computer interface. From the rigid, time delayed interactions of the punch card interface, to the conventional GUI, and beyond to tightly-coupled extensions of the mind.

The modern GUI is primarily spatial in nature, using targeting as the means of communication.[1] This simplifies the problem of providing feedback (simple position displays will do), but loses much of the potential richness of communication. Gesture recognition attempts to retain this richness, but pattern-based recognition approaches have largely been frustrating and unsuccessful, as the next section discusses.

Conventional interfaces generally do not mesh well with views of the world which involve degrees of belief. Normal practice is to treat all signals as known and all actions as definite. For the situations that they are designed for – an able user, sitting at a desktop with robust keyboard and mouse inputs and high-bandwidth displays – this is not unreasonable. Interface design, however, *implicitly* models the uncertainty of signals; scrollbar sizes, for example, indicate the designer's belief about the uncertainty of a targeting motion. Explicitly treating sensed signals as evidence for intention, and appropriately accounting for the accumulation of such evidence, would make adaptation to new situations a more rigorous process than simply tweaking interface elements.

Many conventional interfaces are also excessively sequentialised (partly as a consequence of ignoring uncertainty). Interaction is forced into a series of discrete steps which must occur in a precise and potentially inconvenient order. This may not be the most natural order from the perspective of a user. More importantly, such sequentialisation prohibits the flow of evidence from future observations to past ones. Events in the future may change the distribution of estimates for events observed previously; many interfaces cannot support such changes of belief, and so sub-optimally use the information the user generates.

I.2.2 FAILURE OF GESTURE RECOGNITION SYSTEMS

Much of the original motivation of this work is derived from observing how poorly "gesture recognition" systems fared as interaction techniques, despite their apparent promise from a theoretical point of view.[2] The information content of a gesture extending through time would seem to be far richer than that of a purely spatial interaction technique, such as the pointing metaphor which dominates desktop computing. Gestural approaches also hold promise in contexts where the well-evolved conventional techniques cannot be applied, such as with mobile computing devices lacking displays

[1] This involves both virtual pointing with a mouse, and physical pointing to keys on a keyboard.
[2] Wexelblat [1998], for example, presents some criticisms of the state of gesture recognition research.

practical for targeting, or with users whose disabilities preclude the use of such techniques.

The pattern recognition approach to gesture modelling has not, however, produced particularly usable interfaces. Notable exceptions include the gesture functionality on some web browsers, but these implement only the very simplest two-dimensional gestures (flicking and circling), and do not exploit the full potential of gestural interaction. One of the key problems with gesture recognition is lack of reference. Systems based around recognition of classes of patterns often fail to communicate the distinguishing features of those classes in a form that users can clearly recognise. Letter shaped gestures are common, for example, but many aspects of the formation of a letter are flexible in the mind of a user. Parameters ranging from the relative scale of elements, to the speed of progression, to the curvature of the gesture, all affect the recogniser in some manner, but users are often aware neither of their own performance attributes nor of those that would communicate the desired intention. Trying to remember which attributes lead to successful performance is taxing, and often results in frustrating interfaces which are apparently erratic.

A common approach is to build ever more sophisticated recognisers, and hope that with enough training examples, the classes assigned to particular goals will eventually converge to those of the user interacting with the system. This approach is costly – it is difficult to develop and test such algorithms – and unlikely to help users model how their own behaviour affects the system. In fact, because the recognition process increases in complexity, users may struggle further to understand exactly how the system interprets their behaviour.

The pattern based approach is often also cursed with segmentation problems: determining where one gesture ends and another begins. One popular approach is to "bootstrap", using a simple hardware device like a button to delineate the ends of the gesture, or using long quiescent periods between gestures. This is hardly optimal: the flow of interaction is interrupted. The extra hardware required is also costly and bulky and quiescent times are inefficient. This is a failure of feedback as much as a failure of recognition algorithms – a button's only advantage over another motion sensor is in its robust haptic feedback.

Displaying relevant, timely feedback, and putting the user in *direct control* of the system is an alternative to the pattern-based approach. Simply "tacking on" feedback to existing pattern recognition systems is rarely a fruitful approach. Feedback is only useful if it provides opportunities for control – if the recognition process is not oriented towards feedback-control from the outset, it is hard to benefit from additional feedback. At best, the feedback in such cases will indicate how motions should be adapted in *future* interactions, rather than those at hand.

Rather than relying on block patterns for interaction, the gesture forms can instead be designed as outcomes of some controllable process. An interactor[3] can rely on the feedback about this process to guide them through the generation process (though subelements of the process may become learned behaviours). This is, in effect, what physical devices like buttons do; they are directly controllable dynamic systems which transform a particular motion into a signal communicating intention, and provide reliable feedback as they do so. Physical hardware, however, lacks the flexibility of software gesture recognition. This work describes how the power of spatio-temporal gesture

[3]The term "interactor" is used throughout this thesis to indicate a human interacting with a system; it is effectively synonymous with user, but emphasises the interaction with the system rather than the simple *use* of the system.

recognition can be brought into a software system, while leaving the user in control of the interaction.

I.3 THE NEED FOR THEORETICAL MODELS

Although many successful computer interfaces have been constructed over the past half-century, there is a noticeable lack of well-founded theoretical principles in their design. Sixteen years ago, Thimbleby (Thimbleby [1990], p.170) noted:

> We can do as many 'experiments' as we like on complex systems, evaluating systems with vast numbers of people, doing sophisticated statistical tests, and so on, all to no avail unless we know what we are doing, and how the results of the experiment bear on future work. [...] I search the literature for **theories** that I can apply in my case [...]; instead I find reports of experiments – sometimes related to my particular problem – but without some underlying theories, how can I know how safely I can generalise those results to apply in my design, with my users, in my language? [emphasis in original]

There are still few solid theoretical frameworks for interaction. Psychology provides many insights into the behaviour of humans in different conditions, and psychological theories are of value in designing interactive systems. Similarly, there are extensive physiological models of the motion and responses of the human body. Both of these, however, detail only the human side of the interaction.

Probability theory gives theoretical models for the combination and classification of evidence. Machine learning techniques provide sophisticated algorithms for inference. Formal methods give rigorous theoretical foundations to the internal behaviour of a system. But these only model the internal behaviour of the computer, not the *interaction*.

The interface itself lacks a solid foundation, especially in the realm of short-term, continuous interaction. Much early work in this field was developed as part of manual control theory, developing and testing models of human interaction with dynamic systems. Little of this theory made it into the subsequent development of computer interfaces. Early systems, with long delays and limited power, were largely constrained to discrete, sparse interactions, where, rather than the user being in control of the system, the user and the system passed "messages" back and forth. The remnants of this philosophy are still widespread in existing user interfaces.

With the prevalence of high-power computing systems with rich sensing and display technologies, the design of the low-level interactions – mediating between the intentions of the user and the actions of the system – is of key importance. This thesis sets out a firm theoretical framework for the design and analysis of such interactions. It presents explicit techniques for designing interactions based upon these ideas, and example implementations to illustrate their utility. The implementations are intended to illuminate the principles; the usability of the specific systems developed will be considered only incidentally.

I.4 CENTRAL THEMES

The aim of this work is to describe a perspective on interface design which provides novel methods for the design and analysis of interfaces, and concrete techniques for working with these ideas. There is a number of themes which underlie the development of this view: these are enumerated below.

I.4.1 SPLITTING THE LOAD

The interaction problem is considered as a negotiated control process where the user works with the system to communicate intention. This involves *feedback* as a fundamental concept in the development of an interface. Correctly timed informative feedback splits the problem of communicating intention between the user and the system. This feedback extends from the long-term information necessary to learn the behaviour of the system to tightly-coupled feedback for direct control.

Many conventional interfaces rely on offline recognition of patterns to communicate. When a suitable pattern is observed, a particular event is generated. This is obvious, for example, in existing gesture recognition systems, but most interfaces operate in this manner to some extent, whether explicitly or implicitly (buttons, for example, recognise trajectories which terminate in the appropriate location with the appropriate velocity and force). In these systems, a model of the signals expected for each intention must be created, either by hand tuning or by training a system on many example cases. While this can be effective, it puts all of the task of inferring goals onto the system (and indirectly onto the designer), despite the fact that the user is quite capable of aiding in the inference process *as it is occurring*.

Feedback can dramatically cut down the space over which inference must be performed. A communicating system which provides feedback knows precisely what it has communicated to the user previously (although not how it was interpreted). Any inference performed can be conditioned on the feedback provided to the user, reducing the recognition problem significantly. The faster the rate of communication the more effective this reduction can become. Feedback also eases the task of modelling the system from the user's point of view. Providing a window onto the decision processes which lead to actions gives users the opportunity to shape their behaviours to best fit the system under control.

The negotiation approach aims to produce intelligent interfaces which leave the interactor in control of the system. By maintaining a tight control loop, and bringing in inference algorithms into this loop, the sense of control over the interface can be retained.

I.4.2 LOOP SUBSUMPTION

This level of control affords opportunities for including models of behaviour to improve the communication quality in the interaction loop. The basic idea is that bringing parts of the control loop into software is both flexible and cheap. Carefully tuned complex mechanical devices can offer high quality interaction (Figure I.2), but they lack the flexibility to incorporate significant intelligence.

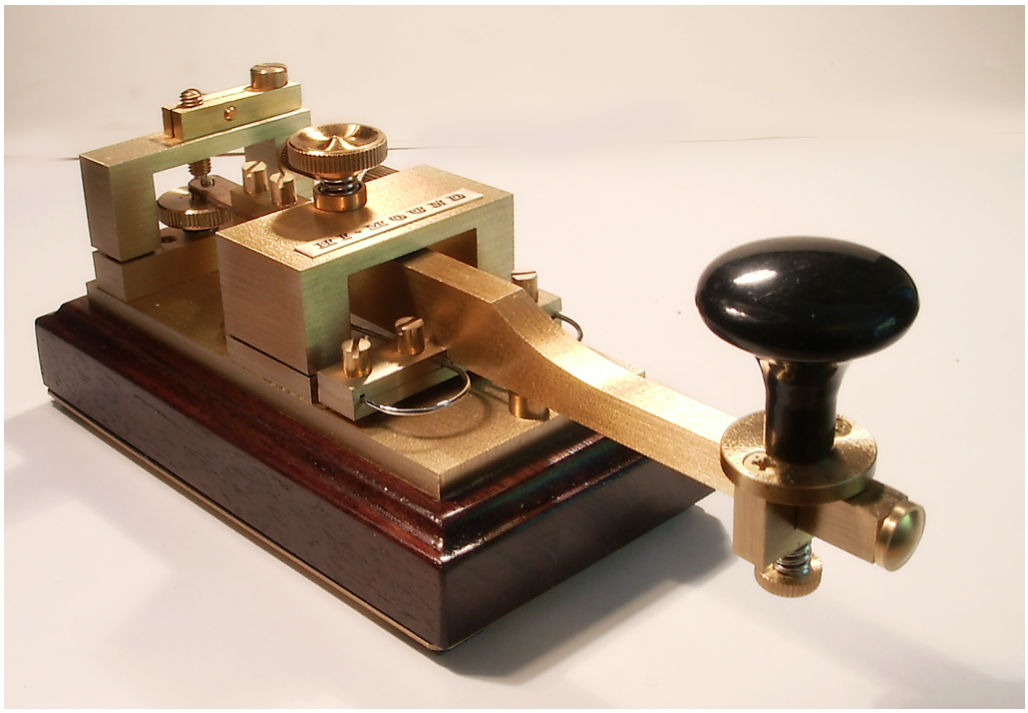


FIGURE I.2: A Morse key; an example of a highly refined physical interaction device. This device has evolved over a period of a century, and is extremely effective at transforming intentions into sequences of Morse code; the weighting, bearings, spring tensions and shape of the device have all been carefully adapted to provide the optimal dynamics to maximise the transfer of information. Building interfaces which have this quality of interaction, but also the flexibility of software control, is a major challenge.

Using general sensors and moving the dynamics of interaction from real world physical interactions into software, the joint human-computer system can be dynamically manipulated so as to maximise the potential flow of information. Some control behaviour remains in the internal proprioception loops within the user which are inaccessible (the targeting motions of the hand cannot itself, for example, be directly manipulated, although they can be modified through learning behaviour). However, much of the rest of the interaction can be brought into the simulated world inside the system.

As an extreme case, in direct brain-computer interfacing there are *no external dynamics* – everything must be simulated from ground up to create a usable system. Interactors are experienced in controlling motions in the world, not controlling the patterns of motor control signals. When the effect of limb dynamics is removed, a replacement must be provided to maintain control. Simulating motions in software allows great flexibility in designing and manipulating these dynamics.

Systems with software-controlled dynamics can bring the functionality of the system into the immediate interaction and make the system a direct extension of the interactor, rather than an entirely separate entity based on events without explicit temporal representation – a tool rather than a conversational partner. Interaction can be viewed as hierarchical control. As Kelley [1968] notes:

> Control refers in its most important sense to choice behaviour; it is through choice behaviour that man controls his environment. An important ingredient of such control is the operation, often automatic, of inner loops which regulate processes required in order that choices made in outer loops may be realised. Both outer-loop choice behaviour and the inner-loop automatic behaviour are aspects of control.

I.4.3 TIME AND PREDICTION

It is important, if interaction is to be efficient, that the delays inherent in the closed-loop are minimised and that the delays are stable; the temporal structure of the interaction is of vital importance. All human-machine interactions involve delays; they are inherent in every interface. Section II.8.4 enumerates common sources of delay: cognitive processing delays, response dynamics of limbs, and buffering times are some common examples. Because of these delays, control is not simply a matter of comparing observed states with desired states. Control depends on a *model* of the *future* responses of the system under control. If the system dynamics are smooth and the delays are short, a simple comparator controller will operate reasonably; an agent behaving like this is implicitly modelling the system as constant over the delay time. For longer delays or more complex behaviour, more sophisticated predictive models are required if control is to be maintained. Aircraft pilots, for example, must learn the effect of their control actions upon the vehicle they control, whose dynamics often introduce long delays between actuation and response.

When the information capacity of the delay in a system exceeds the modelling capability of the operator, control degrades from continuous control to bursts of control where short control signals are applied, followed by a wait period as the signal propagates through the loop. The "bang-bang" mode of control (see Section VI.4.1.1) is one common strategy when dealing with unfamiliar systems with long delays.

Interactive systems can improve interaction by displaying predictive information which helps the user form both immediate and long-term (learned) models of the system behaviour. Predictive feedback can help reduce the modelling load on the user (as Chapter IV discusses). Figure I.3 illustrates the effect of delays.

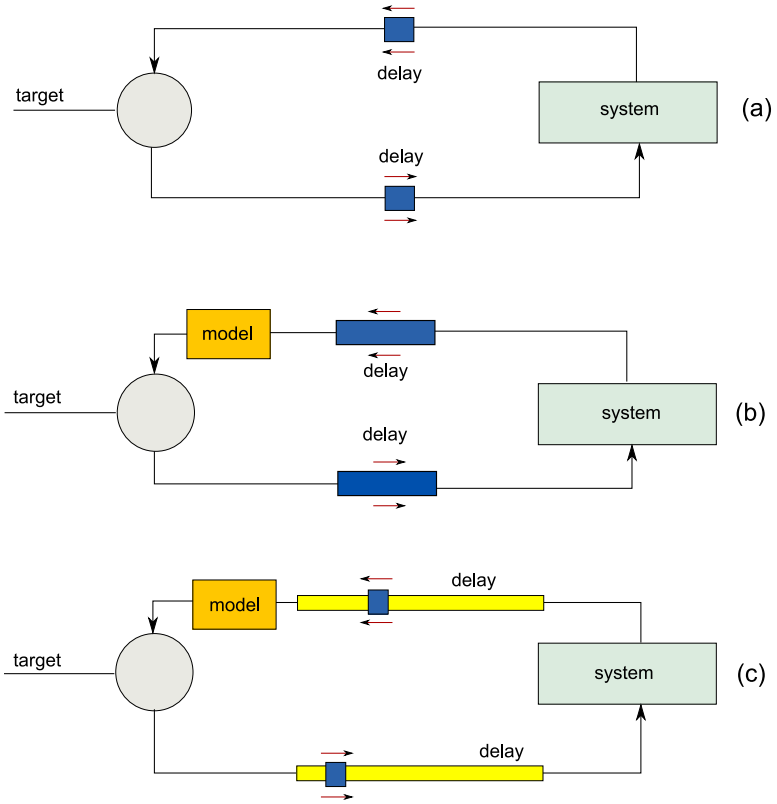


FIGURE I.3: The effect of delays on control. (a) Very short delay: simple comparison of current state and desired state is sufficient. (b) Medium delay: a model of the system dynamics must be introduced to compensate. (c) Long delay: capacity of the delay exceeds the complexity of the predictive model and control becomes bursty. All real world systems have some delay in both directions, but the lengths of the delays may be significantly different.

Systems and users must also deal with information arriving at different rates and delays (e.g. via different modalities). Explicitly accounting for the delay and rate properties of channels involved in an interaction is an important issue in interface design, and one that forms part of the motivation for the framework described here.

I.4.4 UNCERTAINTY AND PROBABILITY

Every interaction involves uncertainty in some form. A basic tenet of this work is that the interface should be *honest* – it must represent and work with the uncertainty present and not filter it out blindly. Such filtering of "noise" limits the information the system can use for inference, limits the information a user can work with to control the system, as well as forcing the inference to be sequentialised in a way that can be suboptimal. [4] Jaynes (Jaynes [1998] p.xxvii) notes:

> When a data set is mutilated (or, to use the common euphemism, 'filtered') by processing according to false assumptions, important information in it

[4]Some sequentialisation is always necessary, both for computational reasons and because systems interact with the external world.

> may be destroyed irreversibly. [...] However, old data sets, if preserved unmutilated by old assumptions, may have a new lease on life when our prior information advances.

Many such *ad hoc* filtering processes are present in conventional interfaces. These were often convenient when computational power was limited, but computational power is now quite sufficient that raw data can be captured and manipulated according to the rules of probabilistic inference. Control of a system depends on reliable information about the state of the system under control – the quality of control is directly affected by the quality of the feedback about the state of the system. Feedback should reflect the uncertainty of a system's beliefs.

I.4.4.1 Probabilistic Models

By representing the uncertainties in the system as probability densities, existing probabilistic tools for manipulating and dealing with the uncertainties can be brought to bear. Such techniques offer a principled approach to dealing with uncertainty. The densities involved in an interaction can be propagated right through an interaction from sensor readings, through the inferential models, through the feedback (i.e. a model of what the user will perceive) and back around the control loop. Techniques such as Monte Carlo sampling (which will play a large part in this work) make it possible to work with probabilistic models in a clean and easily implementable manner. The probabilistic modelling approach also makes it easy to use information theoretic techniques to quantify the flow of information around the control loop. The transfer of information is simply the reduction in entropy a signal provides.

I.4.4.2 Evidence Flow

In summary, a computer observes evidence for the intentions of the user, which is further based upon accessible evidence about the state of the world. There is a constant flow of evidence into the system reflecting changes in the the world. External states cannot be known with absolute accuracy; uncertainty always persists. Evidence for change of state is always delayed and can arrive at different times and at different rates on different channels. Equivalently, the user observes only evidence for the states of the system, which is subject to delay. The interaction problem is to effect changes in a way that reliably corresponds with the intentions of an interactor. Decisions must be made, based on the evidence acquired, which affect the world in irreversible ways.

I.4.4.3 Control in an Uncertain Interface

Almost all existing computer interfaces have feedback mechanisms which do not reflect the uncertainty of the models which describe their state. This simplifies the design of such systems, but restricts the information the user has to model and control the behaviour of the system. In such situations, users cannot observe whether their actions are effective in reducing system uncertainty or not, usually until some (irreversible) action occurs, and thus cannot adjust their behaviour to reduce the system entropy as effectively as possible. Proper representation of uncertainty can help improve control performance.

I.4.5 Multimodal Interfaces: The Potential of Novel Displays and Sensors

There are now a wide variety of sensing and display technologies that can be used to construct the physical aspects of a human-computer interface. Rich sensors, from accelerometers, to smart clothing, to GPS units, to pressure sensors and innumerable others, create the potential for whole new ways of interacting with computational devices

in a range of contexts. Each of these has different information capacities, noise properties, delays, frequency responses, and other modality-specific characteristics. Building interfaces that make use of these characteristics to create usable communication media is a challenge. This work sets out a number of techniques which are not tied to specific sensing or display devices, but generalise to wider classes of devices.

I.5 AGENDA AND OUTLINE

I.5.1 THE PRIMARY AIM OF THIS WORK

The primary aim of this thesis is to present a new perspective on interface design, which makes clear a number of relatively unexplored areas of interaction design while unifying a number of existing techniques. These techniques are intended to be suitable for use with a wide variety of sensing and display technologies, incorporating everything from slow "context"[5] sensing devices to fast, rich real-time interaction devices. The work here is concentrated on "low-level" interaction – the layer immediately on top of the physical hardware. The challenge of designing a usable word processor, for example, will not fall within the scope of this thesis; but the problem of entering text will be tackled.

I.5.2 A GLOBAL STRUCTURE FOR INTERACTION

Figure I.4 illustrates the global structure of the thesis. It shows the disarticulation of the interface loop into component parts: user, interface dynamics and inference. The fundamental thesis is that the interface can be described as continuous control of a point in an intention or goal space, and that the system dynamics form a series of shorter control loops which mediate the communication of information between the sensing and the (extremely high-dimensional) goal space by extending the inference over time. This potentiates a vast range of novel, intelligent interfaces by drawing the dynamics inside the software of the system; *performing inference in a software environment, with all of its associated freedoms, rather than in fixed hardware systems with necessarily limited complexity*.

[5]Context, as Dourish notes (Dourish [2004]), is a slippery notion. Here, context can be considered to be sensed information which affects the state of the interaction, but which the user is not directly controlling to communicate their intention.

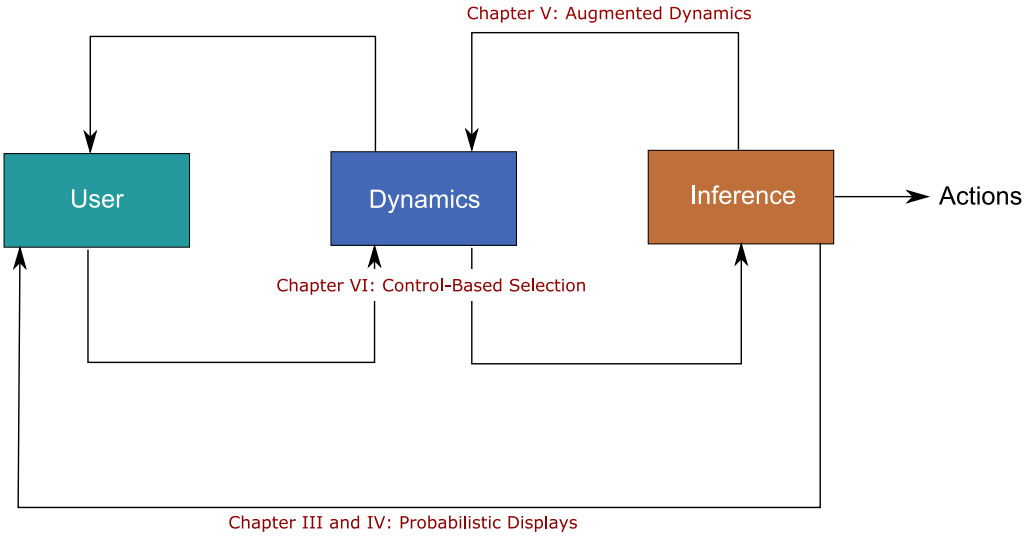


FIGURE I.4: An overview of the structure of the thesis. Chapters III and IV discuss feedback from the goals; Chapter V explores feedback from the inference process to the interface dynamics; and Chapter VI describes general techniques for building selection systems.

This thesis will focus on three aspects of this diagram: displaying goal states and predictions thereof (feedback from the inference model to the user); automatically tuning the parameters of a dynamic system according to an inference model to increase interaction power (feedback from the inference model to system dynamics); and building general probabilistic selection methods (designing the dynamics unit).

I.5.3 THESIS OUTLINE

II THEORY AND DEFINITIONS	Explains the theoretical background to the work, expanding on the themes noted above. Introduces a decomposition of the interface into component parts, and the notion of goal space trajectories. Describes conventional interfaces under this model.
III PROBABILISTIC FEEDBACK	Introduces granular synthesis for display of probability densities, and gives general algorithms for asynchronous granular synthesis. Demonstrates how Monte Carlo sampling can be a powerful technique in interaction design.
IV PREDICTIVE UNCERTAIN FEEDBACK	Extends displays via prediction with appropriate uncertainty. These displays are applied in helicopter control and GPS navigation problems. The use of probabilistic predictive feedback with particle filters is discussed.
V AUGMENTED DYNAMICS	Describes linking the handling qualities of the dynamic system the user interacts with to longer term probabilistic inference. Existing methods, such as semantic pointing, are shown to be special cases of this model. The technique is demonstrated in a text entry system suitable for mobile platforms augmented with inertial sensors.
VI ACTIVE SELECTION	Describes a general framework for selection mechanisms. Methods are given for selection which are fully probabilistic and feature continuous changes in intention estimates. Illustrative implementations are described. Extensions to general exploration techniques in high-dimensional spaces via controlled Markov Chain Monte Carlo processes are outlined.
VII CONCLUSIONS	Conclusions drawn from the thesis, and discussion of the many future avenues of investigation.
APPENDIX A	Describes the granular synthesis library used for implementing examples in Chapters III and IV.
APPENDIX B	Summarises the thesis in terms of general guidelines for designers of interfaces.
APPENDIX C	Guide to the online materials which accompany this thesis, including video and audio footage.

CHAPTER II

THEORY AND DEFINITIONS

Defining closed-loop control, interface models, and the structure of a dynamic interaction.

Do I dare
Disturb the universe?
In a minute there is time
For decisions and revisions which a minute will reverse.

– T. S. Eliot

II.1 SUMMARY

THEORETICAL foundations for the remainder of this thesis are presented in this chapter. The meaning of "human-computer interaction" is analysed from first principles. The role of uncertainty and delays in interaction are discussed. The control theory perspective on interaction is introduced, along with the concept of the goal space. Interaction is defined as control of a "goal cursor" in this space. Existing interfaces are reviewed from this perspective.

II.2 FUNDAMENTAL ASSUMPTIONS

II.2.1 INTRODUCTION

This thesis describes novel interaction design strategies, all of which are based on a particular perspective on the interaction problem. By stating explicitly the assumptions that this perspective entails, the derivation and analysis of the new techniques can be made more rigorous.

II.2.1.1 WHY COMPUTERS?

Computers are providers of utility. The only utility a computer, in the abstract sense, can provide is information;[1] computers are useful because they provide useful data that otherwise would take greater effort to acquire. This can be a straightforward transformation of information, as in a calculator, or the computer can mediate between external mechanical

[1]The utility of a computer is simply the utility of the information it can provide. It is not the utility of the processing the system does – from the point of view of an interactor, computers would be just as useful if they were oracles rather than information processors.

devices or other humans, or to allow a user to explore their own intentions. In all of these cases, the computer is a provider of information. Olsen et al. [1997] puts it thus:

> The ultimate purpose of computation is to leverage the human intellect across the barriers of time, space and force. Computing achieves this leverage by providing control, communication and information that is far beyond what a given individual can do in a given amount of time. Computation provides the connectivity among human intellect, information acquisition and effecting actions in the physical world.

II.2.1.2 Why Interaction?

An entirely passive system[2] has limited value – the output of the system at any time is unlikely to be the most salient information from the perspective of the user. Interaction is a way of increasing the value of a computational system. It is also a cost to the user, who must exert effort to communicate. An efficient system has an interaction cost much less than the value of the gain in useful information as a result of interaction. An interactive system functions as an *information amplifier*; a continuously interactive system might be better described as an *information resonator*. The quality of interaction determines the "ratio of amplification".

II.2.1.3 The Meaning of Interaction

An interface, in the most general sense, communicates the state of hidden variables between the user and the system across the interaction modalities available. In the interactions that will be discussed in this work, the communication takes the form of a mutual negotiation of these variables. From this point of view, the system interface can be considered to be a continuous control system, with sensory inputs, a display and an inference mechanism which continuously interprets the sensor outputs to infer the intentions of the user and feeds the result back to the user. The user and system attempt to negotiate a satisfactory interpretation of the user's intention. The quality of the interface can be valued as the cost to the user in navigating the system towards the intended goals.

These fundamental assumptions are expanded in the following sections:

II.2.2 The Aim of an Interface

The ultimate function of an interface is to *optimally ascertain the intention of a user* with the minimum effort expended by the user. One interface can be considered better than another if the interactor can correctly specify their intention while expending less energy. More specifically, a cost can be associated with communicating one bit of information to a given interface. An interface is "good" if, among a set of potential variations, the cost per bit is minimised. In this thesis, the role of interfaces will largely be restricted to those in which specific actions of external utility are to be performed by a system.[3] The interaction process will be considered only as a means to an end.

[2] Passive from the perspective of any user, that is that no action a user takes has an effect on the subsequent evolution of the states of the system. Obviously, at some point in the past *someone* must have been in control of the system – it had to be designed and built. But from the perspective of the imaginary user, the system is not controllable.

[3] Some systems, such as games, exist simply to provide interaction without external utility. In these cases the *process* is more important than the results; communication is of less interest than the qualitative "feel" of the interaction. In such cases, an interactor uses the system to *control otherwise inaccessible mental states* – humans cannot, for example, stimulate their own pleasure centres, but they can utilise games to do so indirectly.

The bit-rate is a measurement of interaction quality which uses time taken to communicate as a proxy for bit-cost. This cost can be generalised to include factors such as discomfort, cognitive load, and frustration, but it is often reasonable to assume that such factors will have an indirect effect on rate over a longer period of time. Frustrated, uncomfortable or distracted users are usually less effective at communicating than happier interactors and consequently have lower communication rates.

From this assumption, bit rate can be seen as an effective measure of interaction quality. Important questions, however, remain as to when this measure of interaction quality is taken. Some interfaces may have extremely high terminal rates, but have very poor interaction before the user has learned the interface behaviour (e.g. chording keyboards), while other interfaces may permit effective communication from the very first interaction, but have no great increase in skill with further learning (Figure II.1). In these terms, an intuitive interface can be considered one that efficiently exploits the existing knowledge of the user and has a consequently rapid increase in interaction rate at the onset of use.[4]

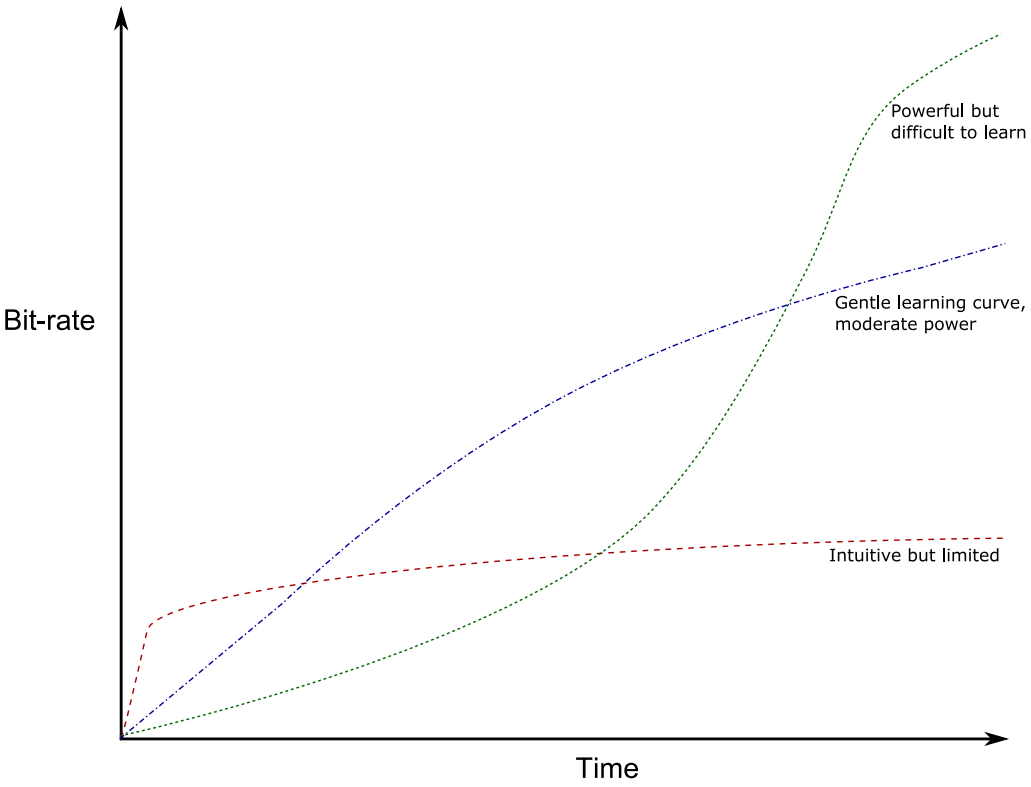


FIGURE II.1: Bit-rate curves for different interfaces. Many interface design decisions involve trading off ease-of-learning against ultimate interaction rates.

[4]This poses a practical experimental problem, as long-term studies of interaction are often infeasible – this strongly biases interaction experiments towards interfaces that are quickly mastered.

II.2.3 SIGNAL PROPERTIES

II.2.3.1 UNCERTAINTY

Any knowledge of the world is uncertain to some degree. Every indication of state from any perspective is inherently inexact. All sensors in a system will have a level of noise; the true state of inputs can never be precisely known. More general uncertainties arise from mismatches between models of the latent variables to be communicated. The time series from any set of sensors may give evidence as to the intention of a user, but while more accurate sensing or longer term measurements can produce more evidence, they can never eliminate all uncertainty. It is vital that an interface take account of the certainty of any inputs available to it, the certainty of any models it uses, and the certainty of interpretation of any outputs it may produce. The recursive reflection of uncertainty between multiple interacting parties – the uncertainty of your model of my model of your model of my model and so forth – forces the diffusion of uncertainty among them. A "good" system uses the control loop to reduce uncertainty; a "bad" system amplifies the uncertainty. It is important to consider the uncertainty of an entire interacting system and not just the uncertainty in each of its components.

II.2.3.2 CONTINUOUS TIME

All interaction occurs in continuous time. For optimal communication, time must be explicitly accounted for.[5] In this work, interaction is considered as a continuous control process at multiple timescales.

This does not require a continuous time computation (as in an true analogue computer), but does require that interactions are recognised to be happening as a continuous process – something often ignored in designing conventional interfaces, where simple static state machines are used to model interaction. Button presses, for example, are often modelled as discrete events which induce transitions between states in some model; but it must be borne in mind that these are not isolated, instantaneous events but have a particular temporal context and evolution (see Section II.11.3).

The interactions described throughout this work are assumed to be implemented on digital computers. Thus, the sensor inputs are assumed to be sampled (regularly or otherwise) and quantized, and display is assumed to be quantized similarly.

II.2.3.3 DELAYS

Every interactive loop has some level of delay present, from the fundamental physical limitations of the world (speed of light, speed of sound) through mechanical limitations (acceleration profiles of limbs, responses of surfaces) through cognitive delays (e.g. the processing delays in the visual cortex) to computational delays in processing sensory input and producing feedback. These properties both restrict the bandwidth of the interaction – in the case of lags,[6] where the frequency content is restricted – and in the case of simple delays which do not restrict the (one-way) physical bandwidth but influence the quality of interaction in a closed loop. The presence of such delays requires that interacting parties have models of the interaction loop which can predict the effect of actions upon it. The higher the quality of this model, the less reliant such a system is on feedback from the world and the longer delays can be tolerated. Improving both the predictive power of the system (predicting how the user will respond to feedback) and the predictive power of the user (providing

[5] Consider a single discrete binary switch. This would seem to communicate one bit – but if the timing is measured the total information communicated is bounded only by the accuracy in actuating and measuring the switch transition.

[6] Lag is used in manual control sense, i.e. as a filter response, not as a synonym for delay.

feedback to help model the system) should be key aims of interfaces. This is especially true in interaction contexts where very long physical delays exist (such as in remote robots in space applications), where very long processing delays exist (such as in EEG brain interfaces) or where attention must be divided between multiple tasks, introducing intermittent delays into the loop (such as occur when operating a device while on the move). The issues of the timing of events and prediction are further described in Section II.8.

II.2.3.4 CONTINUOUS INPUTS This thesis focuses on the problem of interaction with sensors producing continuously varying measurements. All sensing devices at some level measure analogue values, and it is only with significant post-processing that they can be treated as digital values – this processing is an implicit inference process, and often one which has been refined over many years, to the extent that it is often forgotten that the underlying physical process in continuous. By avoiding discrete state changes as long as possible, the need for after-the-fact correction systems such as undo can minimised.[7] Digital computers must quantize evidence; however this quantization should be as fine as possible, and the uncertainty created by the quantization should be modelled and retained. Maintaining the evidence in continuous form allows evidence to be accumulated in a way consistent with a probabilistic view of the world.

II.2.4 PROCESS VIEW

In summary, the inputs from the user are considered to be a continuous function

$$\mathbf{x} = f(t), \tag{II.1}$$

where the dimensionality of $\mathbf{x}$ is the input DoF. The actual observed inputs, from the perspective of the system, are considered to be noisy observations of true states, i.e.

$$f(t) = g(t) + \varepsilon(t), \tag{II.2}$$

where $g(t)$ is the true state of some physical variables, and $\varepsilon(t)$ is a (time-varying) random variable. These measurements are evidence for the state of latent variables $\mathbf{l} = l_0 \ldots l_n$; so therefore

$$f(t) = h(\mathbf{l}, t) + \varepsilon(t), \tag{II.3}$$

with some function h representing the time-dependent transformation of latent variables. All non-communicative information can be subsumed into the error term $\varepsilon(t)$.

II.2.4.1 A BAYESIAN VIEWPOINT This work adopts an explicitly Bayesian view (see Jaynes [1998] for a detailed examination of what this entails; Cox [1961] demonstrates the consistency of this view, and its consistency with classical logic). In particular, probability distributions will be freely assigned to beliefs in a system. These distributions are the representations of uncertainty in the system, and the axioms of probabilistic inference give the rules by which these uncertainties may be manipulated. Beliefs can be updated by combining priors with evidence to

[7] Consider a pressure sensor which operates like a switch, taking action after some threshold is reached. If the signal is quantized early in the process, a near-press will be recorded identically to a non-press. Even if subsequent evidence is observed which suggests an increase in the probability of there being a press at the original point, there will be no way to combine this new evidence with the original (partial) evidence for a press.

form posterior probabilities. The interaction process is considered as an inference problem: to infer the probability distribution over all potential actions a system provides. The interaction is a closed-loop control process and the ultimate control variable is the distribution over actionable goals. The purpose of the system is to perform recursive evidence updates to infer the new goal distribution, forming a trajectory through the space of distributions. The space in which this trajectory lies is the goal space; this is further described in II.7.

II.2.5 GOALS

From Assumption II.2.2, the function of the system is to estimate intention. This work shall specifically cover the case where there are assumed to be a finite number of goals g_i, that a user may be trying to achieve. This, for example, covers conventional interface components; buttons, menus, keyboard interactions and so on. Each goal corresponds to a potential change of world state, and at any time a goal g_i has a probability of intention $p(g_i|e_{0\rightarrow t})$, given the evidence e observed so far. To effect actual changes of state, goal probabilities must be thresholded in some manner; probabilities must be transformed to decisions via some consistent decision theory. The action-transfer problem is discussed further in II.8.1.

These assumptions encompass a large and rich area of human-computer interaction, and conventional interfaces can be examined under these assumptions (see Section II.11). By explicitly stating these premises it is possible to derive a number of novel interaction techniques – the subsequent chapters illustrate some of the important aspects of an interactive system under these assumptions, and show explicit, implemented systems which attack the interaction problem in a way consistent with the assumptions here described.

II.2.6 SUMMARY

The viewpoint adopted in this thesis can be summarised as follows: the interface is a mechanism for controlling the flow of information from a system, which can either flow directly back to the user, or out into the wider world. The quality of interaction is the ratio of effort expended to utility of the information obtained, both of which are unquantifiable, but can be approximated (e.g. with constant bit costs). The function of an interactive system is therefore to ascertain the intention of the user with the minimal effort on the part of the user. The interaction is formulated as a continuous control process, where the system is constantly engaged in recursively updating a distribution over the potential intentions of a user while feeding the results back at various timescales.

II.3 RESTRICTIONS OF THIS WORK

The class of possible interactions following from the above postulates is in fact so large that certain areas will be explicitly excluded from consideration in this work. These are practical, not fundamental, restrictions and the ideas presented here could potentially be extended to cover these topics. In particular, this work will be concerned with communication between one single person and one single computer. The cases where there exist multiple human parties or multiple computers will not be discussed.

II.4 CONTROL THEORY: BRIEF INTRODUCTION

One fruitful method of dealing with complex, time varying interactions is by treating the problem as a *control task*. An enormous literature exists on the topic of control theory for engineering systems, a broad field which has undergone development since the 1930's. Control theory is concerned with the analysis of *closed-loop systems*, where feedback is a critical component. A "controller" is a device which takes action to hold some value to a particular reference value based upon feedback it receives about the current measured state of the system. Controllers are normally negative feedback systems, where the error (some measure of distance between the reference and current value of a system) is applied to the current output so as to drive the error towards zero (see Figure II.2).

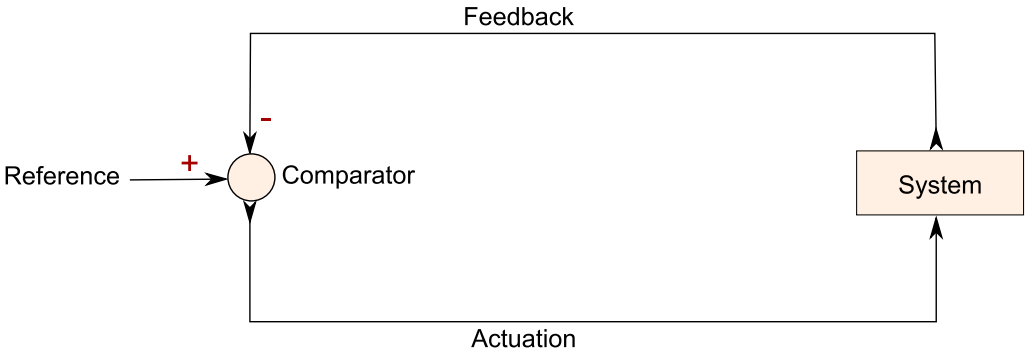


FIGURE II.2: A basic negative feedback control loop. The error signal is fed back to maintain control.

Control theory is concerned with the design and analysis of such control systems. Control systems need to optimise certain criteria; they must hold the error within tolerable limits and, crucially, they must maintain stability over their operational range. This applies both to controllers (where, for example, processing delays can introduce uncontrollable oscillations) and to systems which are to be controlled (e.g. the design of aircraft to avoid pilot-induced oscillation).

II.4.1 MANUAL CONTROL THEORY

Although much of control theory has traditionally been concerned with automatic regulation of external systems, the subfield of *manual control theory* is especially relevant for HCI. This deals with the human control of dynamic systems. Jagacinski and Flach [2003] give a modern and concise overview of manual control theory, while Kelley [1968], Sheridan and Ferrell [1974] and Poulton [1974] are excellent treatments of classical work in the field. This work largely grew out of military problems – especially flight control – where complex dynamic systems had to be manipulated quickly and accurately by a human interactor. Much of this work is concerned with modelling the behaviour of operators in control of different systems; creating generative models of human behaviour. By describing user interaction as a control task, many of the techniques and results of control theory can be applied. Doherty and Massink [1999] and Doherty et al. [2001] contain a basic discussion of the use of control theory for analysis and design of interfaces, and review elementary manual control theory from an HCI perspective.

II.4.1.1 INTENTION AND CONTROL

It has been stated previously in this work that the purpose of an interface is to ascertain the intention of a user. For this to make sense, the definition of what an intention is – and how it can be detected – must be clarified. The control perspective gives a rational way of ascribing intentionality to arbitrary systems (see Chapter 2 of Dennett [1984] for a thorough discussion of this concept). In this world view, the *intention* of a system (be it mechanical, electrical, biological or otherwise) is the reference value to which it attempts to hold some other system. This view implies that any device which engages in control can have an "intention". Thermostats, for example, can be imagined as intending to maintain a particular room temperature. Under this view, a usable system must bring the actions which the user desires under control of the user. This normally must be achieved by splitting control of the true intention into a number of smaller, more manageable control problems.

II.4.1.2 FEEDBACK

It is feedback that transforms a simple one-directional communication into a control process. Feedback has two primary benefits: it simplifies the model users must create of the system, for they can *directly observe* the effect of the their actions; and it simplifies the model the system must make of the user, since the system can consider only those actions meaningful in the context of the feedback provided (this is the key point of Chapter VI).

II.4.2 PERCEPTUAL CONTROL THEORY

In the wake of the cybernetics movement (Wiener [1948], Ashby [1956]), which viewed processes in living things in terms of control and information theory, William Powers proposed the "perceptual control theory" view of organism behaviour (Powers [1973b], Powers [1973a], Powers [1989], Powers [1992]). This approach analyses behaviour – from the simplest physiological regulation to social interactions – in terms of control theory. The basic hypothesis of the work is that *behaviour is the control of perception*. In Powers' own words (Powers [1973a]):

> The purpose of any given behaviour is to prevent controlled perceptions from changing away from the reference condition.

It is important to note that the state of the world is *not* what is being controlled, but the perceptions which reveal the state of the world. Powers described a wide variety of behaviours and activities in terms of hierarchical perceptual control loops, from control of single muscles to control of postures and transitions between them. Numerous experiments were conducted by Powers, and later by Richard Marken (Marken [1995], Marken [2002]) which provide significant evidence that at least some human and other organism behaviour operates as a closed-loop control system; in particular, severing the feedback link destroys the behaviour. Although some behaviour is better described as an open-loop process – the saccades of the eye are one well-known example – and some activity clearly involves pre-programmed actions, the control view can often elegantly explain a number of otherwise apparently complex behaviours.

One consequence of this view is that many activities can be seen as *disturbance rejection*, where a *perceived variable* is being held at some reference value by a controller. Actions are performed to counteract any observed changes in this variable. These are not just low-level, physically sensed measurements such as "intensity of light falling on the retina". They can be quite complex mental constructs – for example, "the position of my hand relative to yours". This view provided Powers and his colleagues with a powerful experimental technique for testing the theories they laid out.

Thus, it is possible to test for intention by looking for behaviour compatible with the control of some variable. This idea of testing for control behaviour is critical in the development of the active selection interfaces described in Chapter VI.

II.4.3 Extension of the Control Loop

One of the aims of this work is to demonstrate how systems can be brought into a closed loop interaction, extending the control of the user much as a physical tool does, rather than the system as an external "Chinese-Room"-style entity. By bringing the tightly-coupled interaction into software, the interface can have the same power as a physical extension of the body but with the flexibility of software control.

Chapter V describes, for example, how by making the interface into a physically-inspired model and then modulating the parameters of that model, "intelligence" can be brought directly into the control loop.

II.5 Control Process

Given the assumptions in Section II.2, the interface can be viewed as a control process where the user is attempting to steer an "intention cursor" (the current probability distribution over the goals) towards a desired goal as smoothly and quickly as possible. The closed-loop structure is illustrated in Figure II.3. This control process is broken down into sub-loops so as to remain cognitively manageable and physically realisable. The user and the system are linked by some physical interface.

An effective control loop seeks to extend the function of a user in such a way by maintaining information over a period of time, but *only those parts related to the goals of the user*. Flexible, software controlled interaction loops recursively filter the state of the entire user/system loop to maintain a current state which usefully represents an interaction.

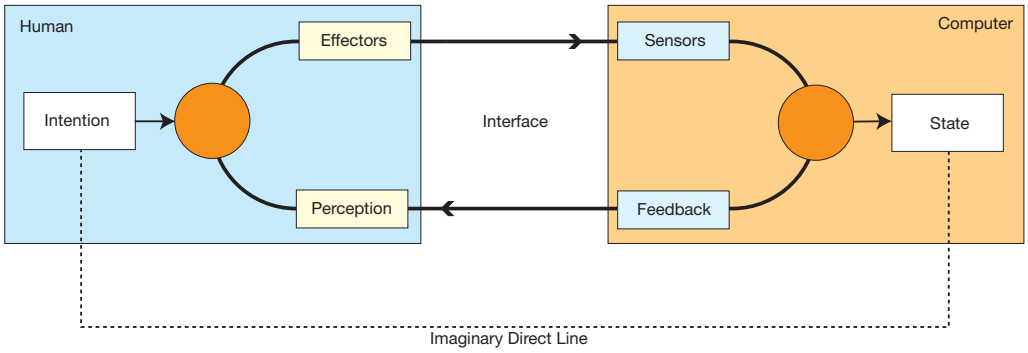


Figure II.3: The human-computer interaction as a closed-loop control process. The "imaginary path" between the intention of the user and the actions of the system is shown as a dashed line. Real communication must take place through the interface, via the control loop. The circles indicate comparator units.

II.5.0.1 BACKGROUND INTERACTION

The notion of *incidental* or *background* interaction is discussed in Dix [2002], Hinckley et al. [2005] and Buxton [1995]. Background interaction involves ascertaining the intention of an interactor without the interactor engaging in control activity to achieve that state. The change of state is instead inferred from evidence about a secondary control task (which may or may not directly involve the system which carries out actions); the dependencies between tasks in the world are used to minimise redundancy in the acquisition of evidence. For example, the act of picking up a device is a control process to bring the position and orientation of the device within some bounds; it also provides strong evidence that the interactor wishes to interact with the device in its new spatial configuration, and that the orientation of the screen should adapt to suit. Such concepts fit within the framework laid out here, but it should be borne in mind that in general actions which an interactor has no control over are undesirable. Background interaction must have strong dependencies between the known intentions and the inferred intentions which they act as proxy for, or must invoke secondary processes by which the background interactions can be directly manipulated.

II.5.1 NEGOTIATION

This control-oriented view leads to a notion of *negotiated interaction*, where the user continuously negotiates with the system to communicate beliefs. This is in contrast to many existing techniques, where the interaction takes the form of discrete action–accept/reject cycles. In such interfaces interactions of the form "select, confirm, perform" are common. The confirmation step indicates that the original decision was taken without sufficient evidence (for otherwise it would be redundant), but because the original selection was forced into a single discrete action, further evidence cannot be accumulated without additional interaction. This practice of sending a message "packet" to the system and waiting for a similarly packaged response is convenient from the point of view of design and implementation of interfaces (it can be easily modelled with state machines), but it does not necessarily make the best use of the capabilities of the user. Obviously, discrete, event-based interactions can be described under the framework set out above, but the space of possible interactions possible under this view is very much broader; it permits interfaces where the system can be considered a highly flexible tool, tightly coupled to the user, rather than a distant conversational partner.

The key idea of negotiated interaction is that information can be allowed to flow between the participants in an interaction without rigidly structuring the temporal aspects of the interaction. Decisions can be made whenever *appropriate evidence* is observed, and the accumulation of evidence can be spread over time in a flexible manner. The subsequent chapter of this thesis describe how elements of negotiated interaction interfaces can be designed and implemented.

II.6 DECOMPOSITION OF THE INTERFACE

A general architecture for the user interface is given in Figure II.4. This decomposes the interface into three components: the goal space, the state machine and the evidence space. The evidence space represents the current state of the user input as measured by sensors over some time window (in machine learning contexts this is referred to as the feature space – see e.g. Bishop [2006]). The goal space represents the space of potential distributions over goals, and the system's beliefs about user intention are represented as a point in this space. The state machine represents the current state of the machine, not

including those elements currently being interacted with; it undergoes discrete changes in state when a goal is considered sufficiently probable that an action should be taken.

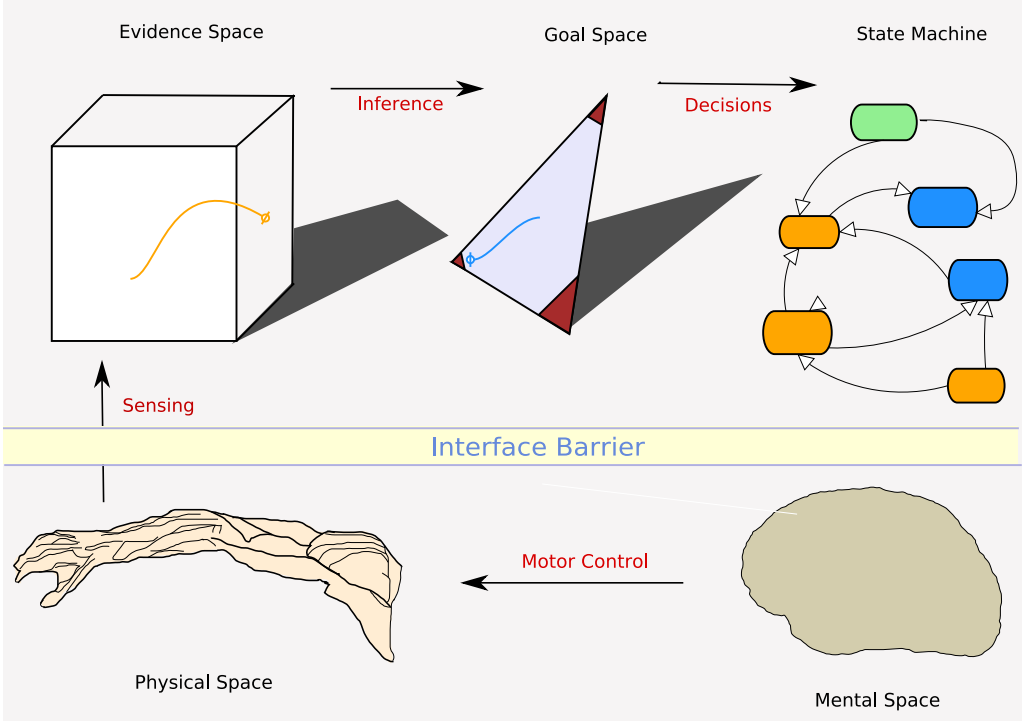


FIGURE II.4: The interface divided into the evidence space, the goal space and the state machine. Only the input side of the interface is illustrated, the feedback paths being omitted.

II.6.1 RELATION TO MODEL-VIEW-CONTROLLER ARCHITECTURE

This decomposition is somewhat reminiscent of the Model-View-Controller (MVC) approach in software design (Gamma et al. [1995] or Buschmann et al. [1996]). In the MVC paradigm, the system is separated into a display, a control or input unit, and an underlying model which communicates with the display and input units through a well defined interface. Here, the goal space is the underlying "true" model which describes the state of the interface with respect to useful actions. The state of this is displayed by some feedback mechanism (such as the ones described in Chapter III). The state of the model is updated via evidence from the sensors (the controller module).

The primary advantage of the MVC architecture is that changes to any of the modules can be made without disrupting the others, and so it is here; although the model never changes – it is always a probability distribution – the display and the evidence update modules are free to be modified. Any new display mechanism can be slotted into the framework for new contexts and any new technique for the updating of belief given the evidence flowing in can be substituted. It should be noted that the evidence unit is not a simple feedforward unit; it incorporates feedback and is (or can be) an independent control loop. The user controls some variables in this unit from which the evidence for

intention is obtained.

II.7 THE GOAL SPACE

The "goal space" is a way of analysing the changing state of the systems beliefs. Given an input x_t (a point in the evidence space), and the current state of the inference process, the system infers a new probability distribution over goals $p_0 \ldots p_n$. This is a point on an n-dimensional simplex.[8] This can be considered the "goal space cursor". Each goal is located at a corner of this simplex. The changing result of the inference can be represented as a trajectory in this space. Figure II.6 shows a trajectory in a 3-goal space, where the probabilities are given by the position of a cursor in a two-dimensional space with several possible densities defined on that space. Figure II.5 illustrates how entropy varies across the simplex for a 3-goal case.

A system which is operating rationally performs actions when sufficient evidence for an intention has been accumulated. The thresholds for these actions can be drawn in the goal space (these are shown, for example as dark red regions in Figure II.4). Crossing into these regions indicates that an action will be executed and the goal estimate will return to some other part of the space. The return position is the new prior over goals, given by the previous action history. [9]

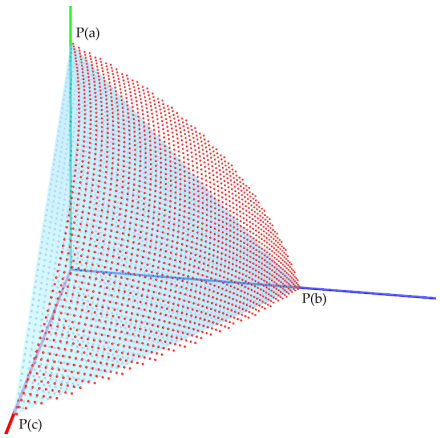


FIGURE II.5: The changing entropy H(x) across regions of the goal space, shown here for a three goal system. Entropy is shown as the dotted surface above the simplex – distance from the surface indicates entropy at that point. It reaches its maximum of 1.584 bits at $< \frac{1}{3}, \frac{1}{3}, \frac{1}{3} >$, and drops to zero at the vertices.

II.7.1 TRAJECTORIES IN GOAL SPACE

The interaction problem can now be formulated as a control problem in goal space – the user tries to steer the inferential process towards the goal intended. This is most effective when timely feedback of the goal state of the system is presented to the user while interacting (the subject of Chapter III). A particularly interesting issue is then how to determine the optimal trajectories in the goal space for an arbitrary system.

[8] Consider the space of possible probabilities. All lie in in the positive orthant and touch the axes at (1,0,0..), (0,1,0,..) etc. The probabilities are normalised such that $\sum_n p_i = 1$. This requires that the surface is normal to the vector (1,1,1,...), i.e a simplex touching each axis in the positive orthant.

[9] The issue of "returning" is a result of considering the goal space to be a space of probabilities of what might be termed single or atomic actions. A "pure" goal space would include every possible sequence of actions – in this space action performance has no effect on goal estimates. This is, however, entirely impractical as an analytical tool, as the goal space in that case is unbounded, and so the "reset to new prior" operation suffices as an approximation.

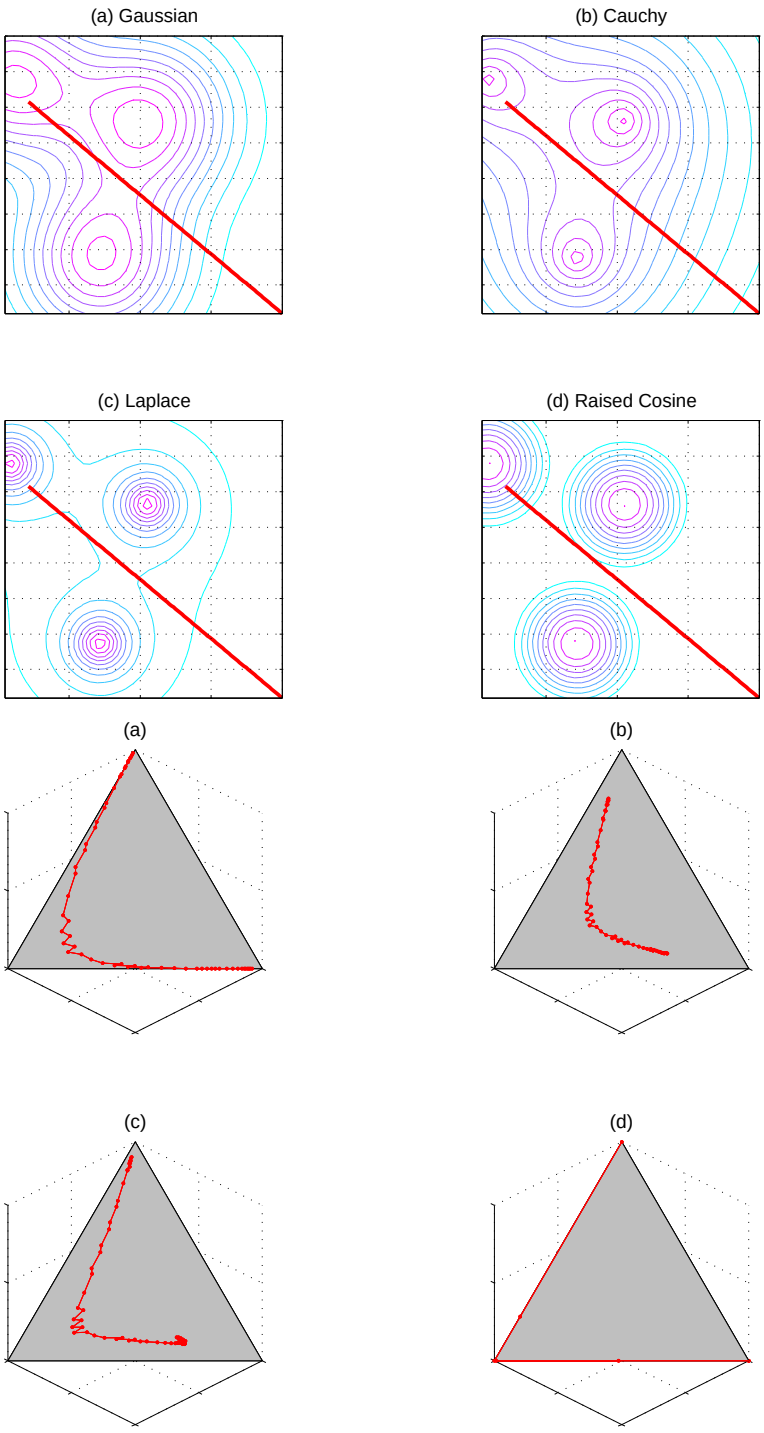


FIGURE II.6: The plots at the top show four example densities on a two dimensional space. The contours show the total probability at each point (i.e. sum of all three goal probabilities). The red line is the path of a test trajectory, beginning at the lower right.

(a) shows a Gaussian/squared exponential kernel $e^{\frac{-x^2}{\sigma^2}}$,
(b) a Cauchy kernel $\frac{1}{\pi\left(1+\frac{x}{\sigma}\right)}$,
(c) a Laplace/double exponential kernel $\frac{1}{e^{\sqrt{\frac{x}{\sigma}}}}$,
and (d) a raised cosine kernel $1+\cos\left(\pi\frac{\sqrt{x}}{\sigma}\right)$.
The lower plots show the goal space trajectories corresponding to the example path on the 2-simplex for each of the p.d.f's. These densities transform spatial (i.e. sensed) measurements into positions in the goal space (distributions over goals).

II.7.2 Information and Smoothness Constraints

If a point x in the goal space is considered, $H(x) = -\sum_n p_i \log_2 p_i$ is the entropy at that point. The communication rate of the system is given by $\frac{dH(x)}{dt}$. There is assumed to be a maximum potential communication bit-rate b_{max} – the information capacity of the interacting muscle group is one such upper bound, for example; the sampling rate of a sensor is another. If the process is to be controlled by the interactor, however, the bandwidth of the feedback must also lie within the user's ability, as otherwise the interaction will be unpredictably unstable.[10]

So $b_{max} = \min(b_{max_{in}}, b_{max_{out}})$. b_{max} enforces a smoothness constraint on the goal space trajectories; since $\frac{dH(x)}{dt} \leq b_{max}$.

II.7.2.1 Maximum Information Limit: Prohibiting Excessive Bandwidth

The above implies that every well-designed system should have smooth movements in the goal space – large jumps indicate either that evidence has been too slowly sampled (e.g. in a keyboard system, where only the terminal result is available as a discrete decision, although this will still obey the bit-rate law *on average*), and that correspondingly little feedback can have been provided, or that excessive weight is placed on evidence and so decisions are made without basis.

II.7.3 Decision Making and Goals

An effective interface performs actions which are of value to the user. These actions are a reflection of the interpretation of the user's intention. As the user's intention is not directly observable, the system must rely on accumulating evidence and performing actions when sufficient evidence has been acquired. This poses two problems: what is "sufficient" evidence, and when does it arrive?

Firstly, the question of sufficiency. The evidence required for a decision depends on the risk the user is willing to take in making an incorrect decision. There must be a compromise between the quantity of evidence required (measured in bits) and the robustness of the interface. It is easy to construct an interface with arbitrarily high robustness – just confirm every action n times, for suitably high n. This is of course inefficient – the information required should depend on the utility or value of the outcome to the user. Given that communicating bits takes effort on the part of the user, the cost of communication must be balanced against the value of the outcomes. Low-risk, high-value decisions should require less evidence (and thus user effort) than high-risk decisions. The expense of bit generation can be tested systematically (e.g. the measurements of information capacity of a limb, such as those in Langolf et al. [1976]), but the value of actions is generally subjective, and it is up to the designer to provide reasonable benefit analyses of the potential system outcomes.

The decision boundaries for executing a particular action lie on isoprobability contours in the goal space. As these boundaries shrink towards the vertices, more information is required to enter the decision region. The design of these boundaries should imply that high-risk actions occupy less area in this space than low-risk actions. Actually setting the boundaries is a task which must be tuned for each interface – but the boundaries should always respect the utility of the action they are related to.

[10] The estimation of the maximum input rate of the user is complicated in practice, since the predictability of the system depends on the user's (potentially very long term) memory of previous behaviour – it is not just the current output of the system which is used to predict the system response.

II.8 TIME AND PREDICTION

The temporal structure of evidence is also an issue. Evidence accumulates over some period of time; some of this is far in the past, as when a designer elicits requirements and codes them as prior probabilities in an interface (either explicitly or as implicit decisions such as the sizing of targets). Other evidence is acquired during the interaction, but may be widely distributed within it. For example, a text entry system may use the previous letters, words, and sentences, in combination with a language model, to set new priors for the next letter to be entered. To be efficient, a system must be capable of integrating evidence arriving at varying times into a single estimate of intention; one that relies on only near-term inputs is ignoring a great deal of potentially salient information. Unfortunately it is also possible for evidence for an action to arrive after a decision has been made. The *reversibility* of actions is important in the cost of making decisions – an irreversible decision is much more costly to a user than a reversible one. Some existing systems (a well-known example is Tegic's T9 text entry system) postpone decisions and use "future" actions to infer previous states – the actual ordering of events is obviously always causal, but the *apparent* order is not so.

II.8.1 THE BARRIER OF ACTION

At some point however, irreversible decisions must be made ("the barrier of action"), as otherwise the space of potential states of the system explodes exponentially, and also because the external world must be affected at some point. The number of decisions that can be kept reversible has a significant effect on the usability of the interface. The most obvious reversibility in existing systems is explicit undo functionality where the user provides evidence for a new decision – reverse the last one. This has been shown to improve usability of complex systems, although undo typically does not exactly reverse the state of the system; instead some particular elements are restored to a previous state. Undo, and its various implementations are covered in detail in Thimbleby [1990].

Undo is necessary for three reasons: an interactor was unable to predict the response of the system and so performed the wrong action; an interactor attempted to, but was unable to perform the appropriate action (for example because of physical slippage); or the interactor changed intentions (e.g. the user was exploring the capabilities of the system, and decided that the action performed was not the appropriate one in retrospect). The first problem can be improved with predictive displays; the second is question of appropriate evidence acquisition; the third is not generally a system problem, but predictive interfaces which show the potential consequences and alternatives to a particular action can mitigate it.

It is also possible to split decisions into multiple reversible sub-decisions, and allow display of the current decision state while still allowing evidence to flow in to affect previous sub-decisions. The potential outcomes and their likelihoods can be maintained for some time, continually updating the previous n states as well as the current one. This is a more efficient use of the information provided by the user, but requires significant additional computational resources. In a text entry example, strong evidence for following letters may affect the probability of a previous letter: observing strong evidence for "at" makes it much more likely that the previous character was "c" or "h" than "z". The final word can then be estimated via a suitable decoding technique, for example Viterbi decoding. This is similar in concept to the distinction between "eager" and "lazy" interaction (Thimbleby [1990]) where the strict temporal ordering of interaction is relaxed; users can fill in information according to their own needs, rather than those of the designer.

The designer of an interface must carefully trade-off the reversibility of decisions with the computational effort required, and also with user expectations of the system. Certain decisions may have to be forced to be irreversible to avoid confusion on the part of the user; the notion of *closure* (Dix et al. [1993]) is considered to be of importance in HCI. Leaving options hanging in flux for too long can prove stressful for the interactor. A good display can, however, mitigate these issues by making the state of the system clear during an interaction.

II.8.2 Utility And Time

When considering a utility model for an interaction, the timing of evidence is often crucial. In interacting with external systems via a computer interface, strict constraints are often placed upon the periods in which actions are useful – it is not much use to command an aircraft to ascend if it has already hit the mountain. The relation between the arrival of evidence and the utility of the actions for which evidence is obtained requires a more complex decision process than a simple static system. Horvitz and Rutledge [1991] discuss some of the issues in making decisions in uncertain conditions with time-varying utility. Figure II.7 shows possible curves for evidence for, and utility of, particular events. Balancing the evidence and the utility for different systems will depend on both these curves and on the delays inherent in a system – even if the average *rate* of evidence acquisition is high, delays can disrupt the interaction. Accepting quick but potentially inaccurate decisions may be optimal in such cases.

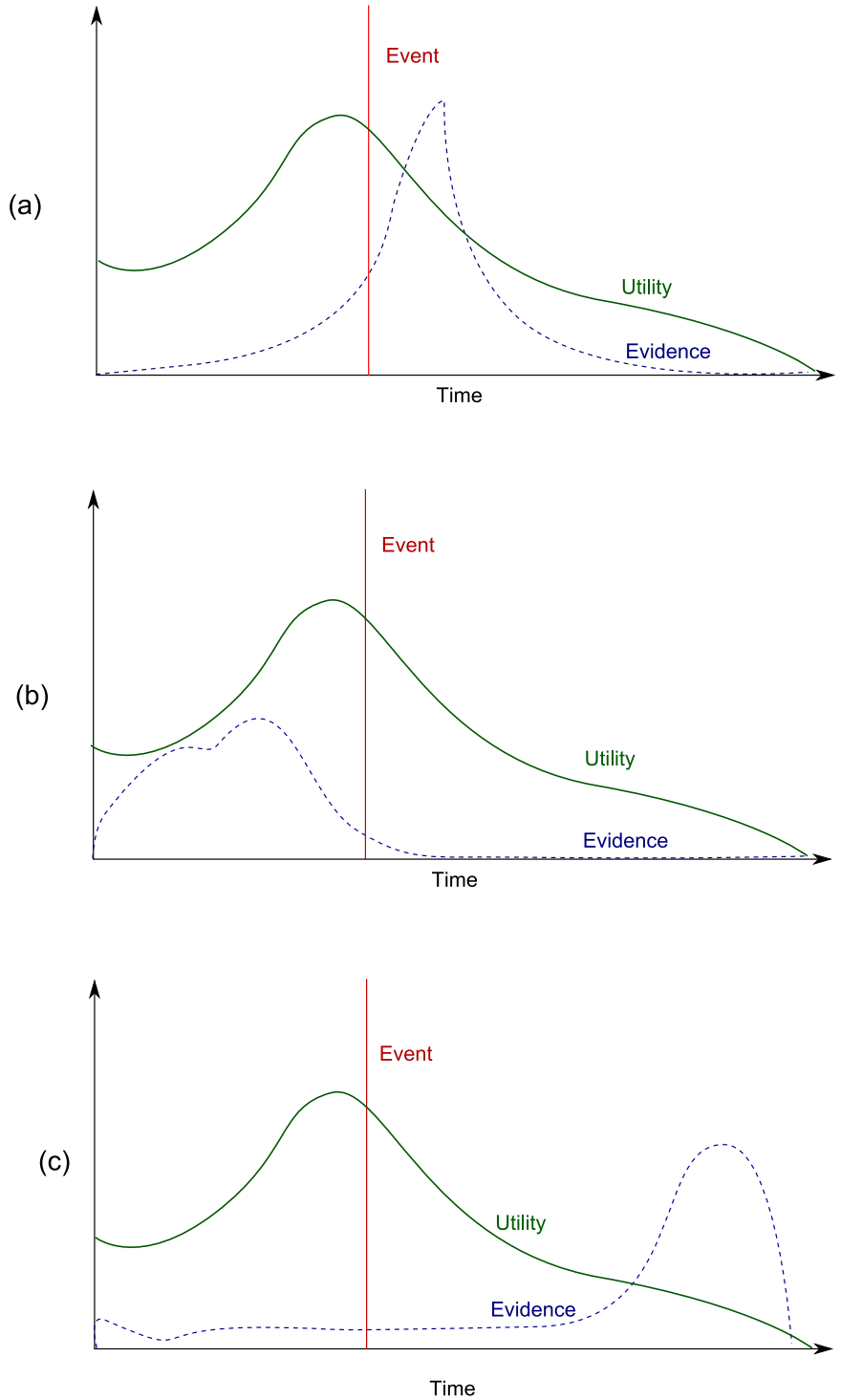


FIGURE II.7: Time-dependent utility of information and the arrival of evidence. Each curve shows the utility of information as a function of time (solid green, fixed for all three cases) and the arrival of evidence for that piece of information (dotted blue). In (a) evidence arrives at nearly the same time as the maximal utility of the information; effective control is possible in this case. (b) shows a situation where the system under control is very predictable; evidence for the state of the system is accumulated well in advance. In (c), information is not reliable until long after the event has occurred, and is no longer useful at that point. Such a system cannot reliably be controlled.

II.8.3 Delays and Lags

Delays in the control loop can reduce the quality of interaction, even where the apparent physical bandwidth remains high. From the perspective of a control process, delays can introduce instability (e.g. pilot induced oscillation, see Robbins [1999]) because of the misalignment of feedback and responses. Humans are quite sensitive to delays in interaction loops; Mackenize in MacKenzie and Ware [1993][11] found that delays above 75ms induced significant reduction in performance in visual targeting tasks, with error rates increasing by 200% by the time lag reached 225ms. Ellis et al. [1999] tested user response to changing latencies between motor movements and visual response with a virtual reality display. Delays of less than 33ms were noticeable; the authors speculate that delays down to around 16ms may be perceptible. Cunningham et al. [2001] and Miall and Jackson [2006] examine the adaptation to delays between motor control and visual display; delays of around 300ms initially have a strong negative effect on tracking performance but can be compensated to some extent over time. Maki-Patola and Hamalainen [2004] examined the effect of latency in musical expression tasks (i.e. the effect of delay in audio feedback systems); latencies as small as 20ms were found to be noticeable. Bryson [1993] suggests that there is an approximately linear relationship between visual delays and RMS error in tracking tasks with "virtual reality"-style 3 dimensional displays. An extensive bibliography on the effect of delays in manual control is given in Ricard [1994].

One key function of an interface is to create a *shared* sense of time between the interactor and the system, so that actions and responses can be synchronised with each other. A system with slow, irregular sampling, for example, is very difficult to control because the system and the user "perceive" the flow of time differently. Implementing the interface as a dynamic system, whose evolution implicitly represents the flow of time, is a way of achieving this shared temporal context.[12]

II.8.4 Common Sources of Delays and Lags

Some sources of delay in interactions (and some rough estimates of the actual induced delay) are shown in Figure II.8. It is apparent that there are a large number of potential sources of delay. Where possible, the best solution is to remove or reduce the delay (e.g. with faster processors), but in many cases this is impossible – the cognitive processing time for visual stimuli is permanently fixed,[13] for example. In these cases the only effective solution is to build better models of the interaction so that control can be performed on more accurate predictions of future states. This is the major theme of Chapter IV.

[11] The term lag is used as a synonym for delay in Mackenzie's paper, as opposed to its sense in manual control.

[12] Interactions with oscillatory components are an interesting field of research (see Lantz and Murray-Smith [2004], or the resonant displays of Chapter VI). Such rhythmic interaction, where phase synchronisation locks the interacting parties in step, can mitigate the effect of interaction delays, and fits naturally with typical motion patterns generated by humans.

[13] Although careful display design can still minimise this – certain visual patterns will be responded to more quickly than others, for example. The effect of brightness is, for instance, described in Rains [1963], and the effect of flicker is discussed in Roufs [1974].

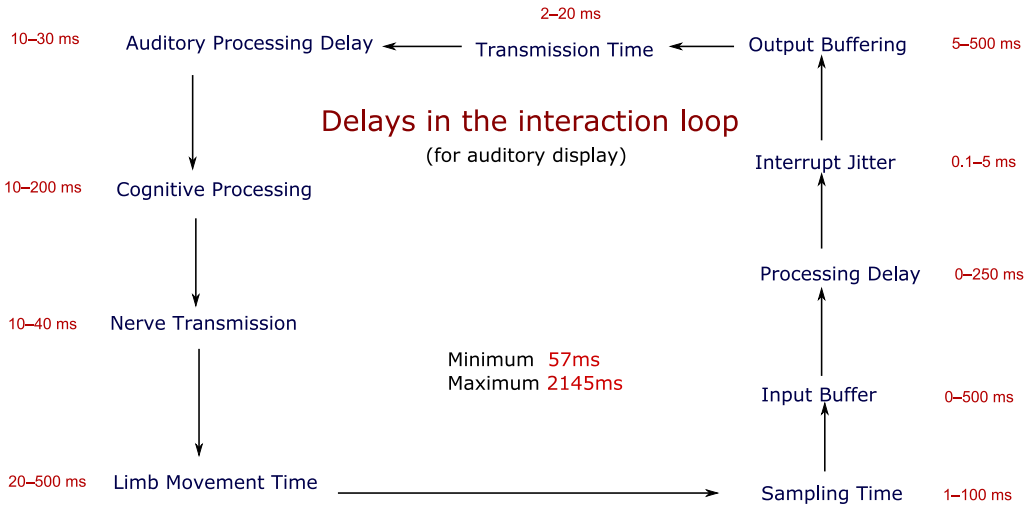


FIGURE II.8: Some roughly estimated delays in an interaction, for an audio display. The upper bound time is 2145ms, the lower is 57ms. For a visual display, the transmission time would drop to almost zero and the visual processing delay in the brain would increase by ∼20ms.

II.9 THE EVIDENCE SPACE

The sensor space is simply the space that the sensory inputs to a system represent. A point in this space represents the complete state of sensory measurements at a single instant. The inference mechanism transforms trajectories in this space into trajectories in the goal space.

The evidence space is an augmentation of the sensor space which represents the current state of the system in terms of values inferred from sensors over some time window. It is derived directly from the sensor inputs. The space encodes all of the information necessary to make a decision – but points in the evidence space do not represent particular goals. In particular, it represents at least part of the recent trajectory though the sensor space rather than the state at one observation instant. This could be a simple concatenation of sensor values (with some window) but generally representing the recent trajectory in a more compact form is necessary. This can range from simple properties of the signal (frequency content, derivatives, filterbank outputs, spline fits) to more complex inferred properties (factoring the space into sub-goals with a Bayes net). The distinction between the evidence space and the goal space is that the goal space represents only the probabilities of actionable goals; the evidence space represents all of the properties of the signal required to derive those probabilities. This can include arbitrarily complex inference to represent sub-states of the interaction.

This is the space in which dynamic systems can be introduced to create lower-level elements for the interaction loop, and thus mediates between the sensory inputs and the actual decision process. The temporal aspects of the sensory inputs can be coded in a manner which the user can directly control – there is a straightforward way of manipulating these states of the system. Feedback at this level can be combined with

higher level feedback from the goal states to create a system which is both controllable and transparent to the user.

II.9.0.1 The Necessary Size of the Evidence Space

The dimension of the evidence space determines the quantity of information required for the inference to be performed adequately. Short time windows may be suitable for high-DoF inputs, or when relatively few options are available. Longer term behaviour is required when noise levels are high, where the input channels have restricted bandwidth, or where a large number of possible alternative goals are possible. One way in which the space can be restricted is by *conditioning on the feedback*. Since the system knows what output it produced, under the assumption that the user is controlling the system based upon the feedback they are receiving, the possible responses can be restricted to those that relate meaningfully to the feedback produced recently.

II.9.0.2 Error Models – The Inverse Problem

For any given system a particular trajectory through the evidence space results in a trajectory through the goal space. Equivalently, some (but not necessarily all) trajectories in the goal space relate to a particular class of trajectories in the evidence space. The relationship between the goal space trajectories and the evidence space trajectories gives the tolerance of the system; where a large set of evidence space trajectories result in similar goal space trajectories the system tolerates significant deviation, but is rejecting much of the input information.

In some cases it can be useful to draw samples from the distribution $p(X|g)$, that is, the probability of a given trajectory through the evidence (or sensor) space given a particular goal. In particle filter systems, this is how inference is performed; samples are drawn from this distribution and compared with actual sensor measurements. Selection is based upon the measure of similarity between the postulated and actual measurements.

If a system supports such sampling, and a reasonable operator model is available, an objective measure of likely interaction quality can be obtained by computing the cost of trajectory generation for each sample drawn, and summing over the whole ensemble to estimate the cost of achieving a single goal. The text entry system described in Chapter V, for example, uses this approach to optimise the layout of characters.

II.10 Interface as Hierarchical Control Model

The interface can be viewed as a hierarchy of control problems, from the longest term intentions of the user to the most tightly-coupled physical interaction. The outermost loop in a system represents long term learning processes (or meta-control processes), which may be driven by the user (system adaption processes) or by the system (shaping processes). Inner level loops relate to the control of immediate intentions (the negotiation process) which lead to system actions. Inside this lie the control loops which mediate the flow of intention; control of the dynamic systems upon which inference is based. This is the level at which feedback can have most effect in changing the attributes of the control. Yet further inside lies the physical control of the user; controlling the physical state of objects in the physical world. At this point the problem leaves the software domain and any further changes to the control loop are hardware engineering issues. Beyond this lie the internal proprioceptive loops which maintain the motion of

the limbs. These are inaccessible physically, but can be shaped by the processes in the outermost loops: learning and shaping.[14] All loops, save for any behaviour shaping loops are controlled by the user – the system is passive in the sense it can be considered to have no intention with respect to the user's states.[15]

Section V.4.11.1 examines the Hex text entry system in light of this structure, and illustrates how the components of the diagram in Figure II.9 relate to components of the design.

[14]Powers [1973a] presents a breakdown of even these innermost processes into ever tighter control loops, down to the last neuron. Some of these internal loops are speculative, although some are backed by significant physiological evidence; but these issues lie outside the scope of this work.

[15]Chapter VI discusses how introducing "intentions" to a system – the intention to determine the interactor's intention – can be used to design adaptive selection interfaces.

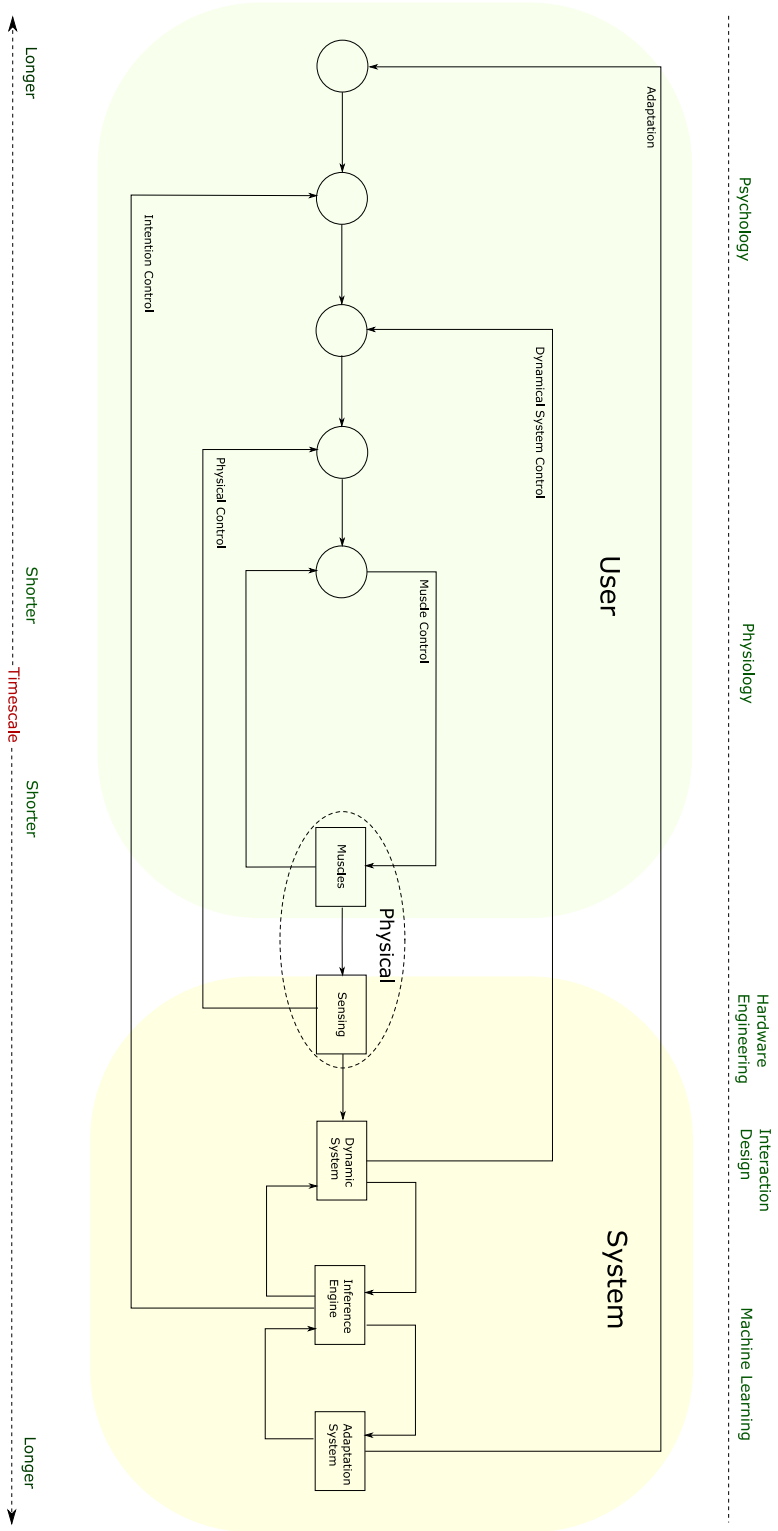


FIGURE II.9: The interface as a series of nested control loops. Layers move outward from physiological loops to complex learning behaviours.

II.10.1 Sources of Uncertainty

It is a contention of this work that uncertainty is a key factor in user interfaces. This uncertainty representation is necessary because the system cannot directly observe the hidden states of the user – only evidence for those states. The system attempts to extract the intention state of the user given the evidence it observes. The ultimate hidden variables for the system are user intentions; but even access to the physical measurements in the world is corrupted by noise. Probability theory provides an effective set of tools for dealing with these uncertainties in a principled and rational manner.

Smith [1997], for example, presents strong arguments for the importance of representing and propagating uncertainty in both measurements and models of dynamic systems. He notes:

> All theorems are true. All models are wrong. And all data are inaccurate. What are we to do? We must be sure to remain uncertain.

Propagation and representation of uncertainty is essential if inference and display are to be accurate, especially when predictive elements are introduced into a system.

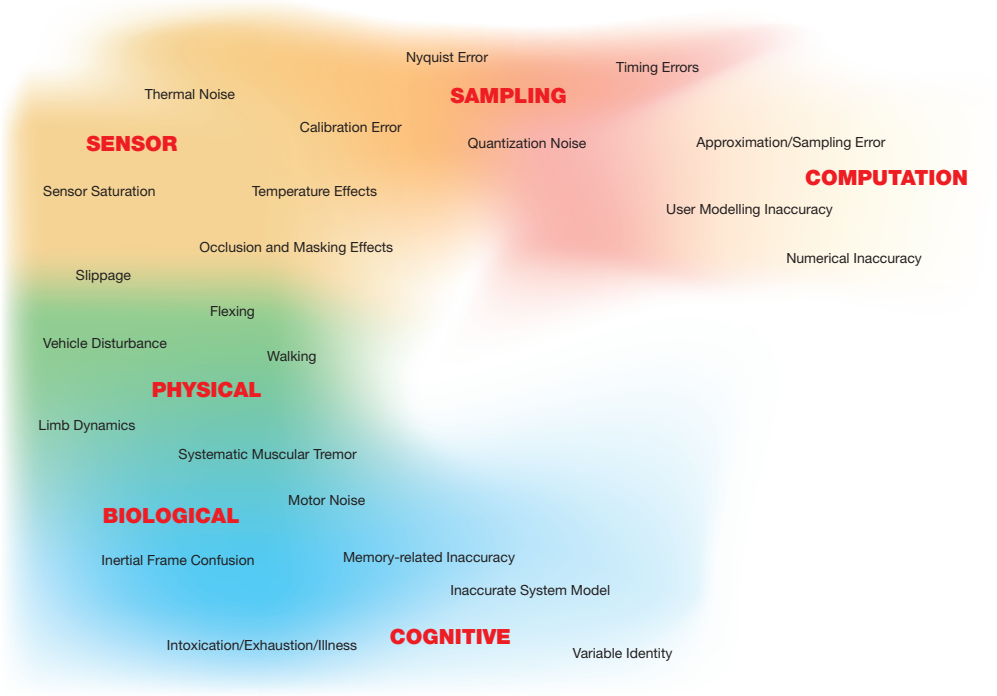


Figure II.10: Some potential sources of uncertainty in the control process.

Uncertainty in interaction arises from many sources. Some of these are physical, hardware limitations like electrical (thermal) noise, sensor mis-calibration, quantization noise and timing jitter. Others arise as modelling and computational issues, for example noise arising from Monte Carlo sampling from a distribution with insufficient samples, or from mismatches between an idealised model and the actual behaviour of a system. Figure II.10 illustrates some of these noise sources around the control loop.

II.10.2 Isomorphism Errors

Some "noise" occurs as non-communicative user action – states which are sensed by the system measurement devices but which are not associated with any intention. Simple examples in motion sensing systems include slippage of poses or disturbances to motion as a result of walking or travelling in a vehicle.[16] All of these will be sensed, but communicate little or nothing about the intention of the user. Many of these issues occur because of mismatches between the sensing hardware and the user's expectations of the sensing hardware. The user implicitly assigns one set of signals to a particular belief class, but the system assigns a different set of systems to the same belief class. For example, users rarely account for the interaction of orientation and linear acceleration when manipulating devices with inertial sensing features. The converse situation also occurs, where the user believes that a change in state communicates intention, but the change is either not sensed or not incorporated into the inference model. A simple example is an interface which quantizes its inputs, and thus small motions made by the user will only have an effect near quantization boundaries, which the user cannot directly observe.

This mismatch between user expectation and system expectation of the meaning of a signal is an *isomorphism error* – the system and the user differ in their belief over which classes of signals are equivalent. This error will always occur in any given system since belief classes cannot be communicated perfectly. It can be mitigated in four basic ways: more sophisticated sensing; better match between system inference and user belief; user training; and feedback of state. Adaptive systems update the inferential mechanisms during interaction to try and minimise this error, but this can lead to degradation of performance when they undermine user learned models of the interaction.

II.10.3 The Latent Variables

At the onset of an interaction, humans possess a large number of indirectly observable values; identity, emotional state, exhaustion level, level of intoxication, and so on, which may not be directly associated with their intentions with respect to the interface but affect what actions they will produce to communicate such intention. Although evidence for some of these states may be acquired during an extended interaction, the system must assume a distribution over these potential states.

All of these issues mean that the system's observations are significantly detached from the actual intentions of the user. It is this indirection that motivates the incorporation of uncertainty into the user interface. It is also the motivation for a control-based approach; a system in which the interactor is engaged in feedback-control of the "intention state" of the interface splits the inferential problem between the user and the system. Mismatches between interface interpretation and user expectation can be countered by the user interactively.

II.11 Conventional Interactions Interpreted Under This Model

This section reviews conventional interaction in terms of the theory laid out above. WIMP-style mouse interactions, button and keyboard interfaces, and gesture recognition systems are covered. More detailed analysis of the properties of various conventional interaction techniques is discussed in MacKenzie [1995]. Buxton [2002] presents a

[16] Obviously, these features can communicate information about context or state of the user, but the important issue is whether they communicate information with respect to the possible goals the system can provide.

comprehensive review of conventional interaction techniques; Hinckley [2002] is a more recent review of input devices and the laws governing their behaviour.

II.11.1 "Static Recognition", the Segmentation problem, and Button Pushing

In most existing interaction devices, there is a movement or positioning phase, which may or may not be sensed, and a terminal event which causes a discrete jump in the goal distributions. Mice have buttons for this purpose; keys only have this phase of interaction (the movement phase is not even sensed); joysticks have buttons; styli have contact/release sensing and in the case of touch screens, position cannot even be detected the movement phase (see Figure II.11). This feature is also present in many gestural interfaces, which use buttons to indicate the start and end of the gesture.

These jumps in goal space violate the information smoothness constraint given in Section II.7.2; they make an instantaneous reduction in entropy. The evidence, however, has accumulated over a period of time *before* the terminal event but is ignored or not even sensed up until that point. Great quantities of potentially salient information are ignored or delayed by such setups, primarily because the detection of intention is constrained to physical systems, where flexibility of response is limited. In such systems the user does not continuously control the beliefs of system; the user controls only motion in the physical world. Positions or sequences of positions are transformed to beliefs instantaneously; the action transfer problem is solved simply by handing it over to the user.

Figure II.11: Terminal events in conventional interaction devices. In each case, a single button style motion is used to trigger the action. Evidence accumulates throughout the interaction, but is not used until this action occurs, and when it *does* occur, it is treated as if it were absolutely certain.

II.11.2 Mouse Click Interactions

The mouse is now ubiquitous in desktop interfaces.[17] It translates two dimensional motions of the hand into corresponding motions on screen, as well as providing one or more physical buttons to perform specific actions. The windows-icon-mouse-pointer (WIMP) design paradigm is purely spatial. It almost completely eliminates temporal dependencies;[18] actions are atomic and isolated in time. The mouse buttons are used to define the points at which the position will be sampled. This makes it easy to learn – any control strategy which results in the appropriate final position will work – but severely limits the theoretical total bandwidth of the system. Section II.12 discusses

[17] This section deals specifically with mice. However, the ideas generalise to other continuous motion sensors which may be substituted, such as trackballs, joysticks, touchpads, etc.

[18] There are some notable exceptions, such as double-clicking.

how Fitts' law bounds the communication bandwidth in such cases. In terms of a goal space, intentions are mapped onto spatial densities with binary values ($p = 1$ inside the target, and 0 outside). The only way to code probabilistic models in such interfaces is to either make targets larger or bring them closer. Rescaling targets is straightforward, but there is only so much screen space available. Making them closer is possible with devices such as popup menus, which dynamically create spatial targets close to the current mouse position, but this is only possible to a limited extent.

The mapping from physical space to screen space is not a simple physical relation. The design of mouse interactions implicitly models the behaviour of operators, and is carefully optimised to increase the flow of information. The mouse motion (sensed by optical tracking or a rolling ball with rotary encoders) is *relative* – physical positions do not correspond to onscreen positions. This relative motion is transformed by carefully constructed transfer curves to improve the performance of the system – one of the clearest examples of dynamical system design in existing interfaces. Figure II.12 shows the curves used in the Microsoft Windows mouse drivers; Barrett et al. [1995] describes the design of transfer functions for miniature joysticks, as used in some laptops. The physical design of the mouse itself has an effect on the quality of the interaction (see e.g. Isokoski and Raisamo [2004]). Gamers are well known to have strong preferences for specific mice, even going to length of obtaining mouse pads constructed of special materials to improve the response. Techniques such as semantic pointing (Blanch et al. [2004]) alter the relation between the visual and sensed relations to improve performance, in recognition of the limitations of the standard dynamics of a mouse interaction.

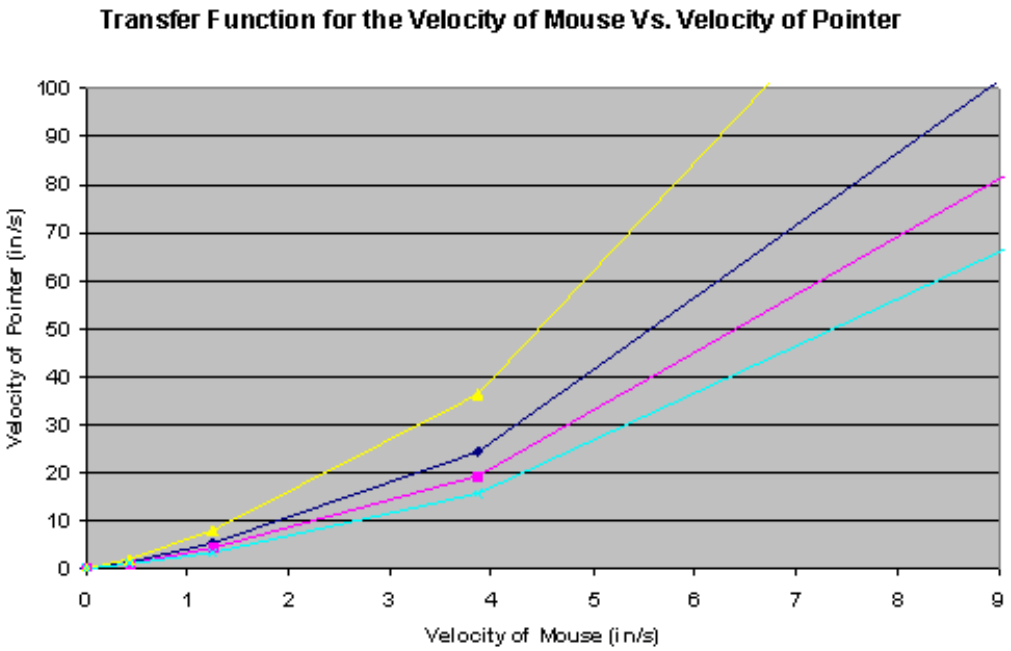


FIGURE II.12: The transfer functions used for mouse control in Windows XP. Taken from the Microsoft website (`http://www.microsoft.com/whdc/device/input/pointer-bal.mspx`, retrieved 10th of June 2006). The mouse control panel has four "speed" settings which select one of these transfer functions. This function is applied after any on-mouse filtering.

Mice are simple ways of transforming motion into intention. The GUI metaphor provides a world with which the user can interact. However, this world is primarily spatial and feedback is very simple (the position of a cursor with respect to targets). Actions are taken based on single, (near)-instantaneous physical measurements (button presses) rather than being taken when appropriate levels of evidence have accumulated. The flow of time is basically ignored. Users may proceed at any speed they wish – at the cost of very limited bandwidth. The dynamics of pointer motion are important to the operation of the mouse interface, but they are often ignored. Performance depends on the quality of the dynamics of the system with which the user interacts.

II.11.3 Interpretation of Button Actions

Physical button (or key) based interfaces are widespread in computing. They are intuitive to use; they *afford* pressing, and the feedback they present reliably indicates the state of the interaction. However, they can be clumsy, are expensive to manufacture, take up significant space, wear out, and have very limited communication power. The treatment of button presses in interfaces limits the communicative power of the interface even further.

Button presses are often considered to be discrete isolated binary events. An interface based around such events is (at least potentially) both easy to learn and easy to implement; for many years, keyboard-based interfaces were the *only* way to interact with a computer system. The entire interaction can be represented as a finite state automaton. Button pushes are assumed to be so reliable that the change of state can be assumed to be perfect evidence for an intention (or sub-intention[19]). In a button pushing interface, the state of a goal space cursor jumps from centre to vertex with no intermediate stage.

There is of course still a dynamic process by which intention is transformed to a communicable form. However, in button-based interfaces, the communication problem is pushed almost entirely into the physical world. The layout, tensioning, weighting, shape and labelling of the buttons are hardware issues rather than software ones. Feedback from buttons is mainly physical: tactile, auditory and visual feedback are all present.[20] Buttons are carefully designed systems designed to have dynamic properties that make communication easy. Unfortunately, these properties are unalterable after a hardware implementation is built. This significantly limits the power of such systems; the only modelling that can be brought into the process is *a priori* models of the likelihood and cost of actions.

However, button presses are not quite discrete binary events, as Figure II.13 illustrates. Significant effort is required to filter the oscillations of a button push ("debouncing") to make it fit with a state-machine model (either fully depressed or fully released). This, however, discards the characteristics of the push – different pushing motions lead to different contact patterns. All of this evidence is discarded by hard-wired filtering in conventional systems.

In hardware which simulates physical switches (e.g. with pressure sensing), this is even more pronounced. The entire force trajectory is available, but is quantized into a single event. Glimpse (Forlines et al. [2005]) is a rare counterexample, which uses variable

[19] That is, the intention to achieve a goal as part of communicating a higher-level intention, e.g. intending to enter a character as part of a word as part of a sentence as part of a document.

[20] Although sometimes additional software generated "key-click" audio feedback is used where tactile feedback is very weak, as was the case with the membrane keyboards of some early home computers.

pressure levels (a reasonable estimate of the probability of intention) to preview actions without committing them.

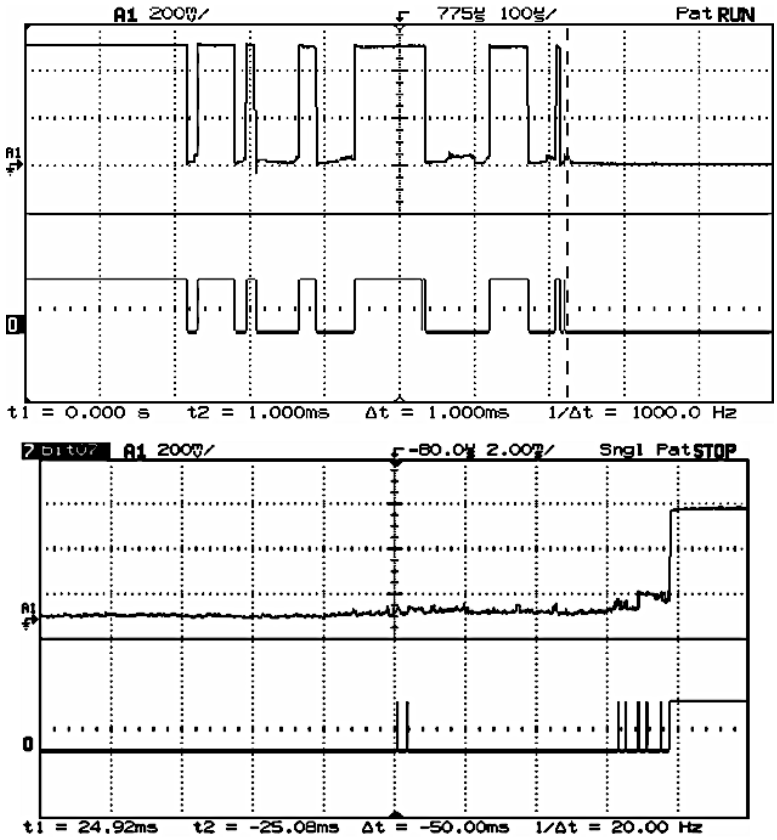


FIGURE II.13: Oscilloscope traces for two different buttons being pushed. Note that the behaviour is quite complex, with noise and ringing around the transient. Debouncing algorithms are needed to clean up these signals for button controls. The top trace in both cases is the raw signal values; the bottom indicates the corresponding logic states. These two images are taken from Jack Ganssle's debouncing guide (`http://www.ganssle.com/debouncing.pdf`, retrieved 15th April 2006).

II.11.4 GESTURES, SPEECH AND OTHER INTERFACES

One significant class of interfaces not falling into the categories laid out above are *pattern-matching* or *recognition* interfaces. This structure is common to gesture recognisers, speech recognition systems, and handwriting input systems. Generally, such systems observe measurements for a period of time and then estimate the best match between the measurements and models of possible inputs. These can be obtained from training data in learning systems, or they can be hand created by a designer. This is normally a static, segmented process, with blocks transformed instantly to belief and then to action.

This makes sense for speech, which provides almost no opportunities for external feed-

back control; the only important control loops for speech production occur inside the body. [21]

Gestural interfaces, however, could theoretically provide a great deal of feedback about the motions performed and their effect upon the system's beliefs, but rarely do so. Instead the user must repeatedly attempt to perform a motion which conforms to the pattern classes of the recogniser (without any effective method of obtaining awareness of the class boundaries). Many gestural systems also use button based segmentation to make it easier to identify the onset of meaningful behaviour – these buttons actually communicate a significant amount of information about the gesture. Systems will normally reject input trajectories which differ too far from any known class; this is better than the absolute certainty with which graphical user interfaces treat interactions. However, there is rarely provision for control as evidence accumulates, and lack of feedback about the process of performance makes it difficult to ascertain in what ways the motion performed differed from the motion which would achieve the action the user intends. Learning is thus impaired.

Static recognition interfaces have potentially much higher utilisation of the available bandwidth for control signals than spatially-based interfaces. However, difficulty in learning (partially because of poor feedback) tends to make them suitable only for niche applications.

II.11.5 Summary: Conventional Interfaces

Conventional interaction mechanisms implicitly model the likelihood of actions. However, it is difficult to bring new models of behaviour into an interface without entirely redesigning it; and it is often impossible to use belief models that dynamically update. Evidence for intention is often ignored, filtered out in hardware as "noise", and when it is accumulated it is (unreasonably) treated as certain. The onus is on the user to perform accurately, rather than on the system to interpret action well.

Conventional interfaces, like all interfaces must, communicate information via a mediating dynamic system. But such interfaces often have static, rigid systems which are well adapted but cannot be modulated according to changing contexts. Mouse interactions brought some of the interaction out of hardware and into the software world, but the process can go much further, as the work in this thesis illustrates.

II.12 Fitts' Law Analyses

This section briefly reviews Fitts' Law, a well-tested model on the information bandwidth of spatial interfaces based on pointing motions (and subsequently extended to certain trajectory tasks).

II.12.1 The Role of Fitts' Law in HCI

Fitts' Law (Fitts [1954]) is one of the most widely used quantitative theoretical models in human computer interaction.[22] In its most general form, it states that there is a logarithmic relationship between the time taken to hit a target, and the ratio of the size of

[21] Delayed auditory feedback (DAF) anti-stuttering devices are one minor exception. These disrupt stuttering behaviour by artificially delaying the audio feedback route, desynchronising it with bone conduction and proprioceptive feedback.

[22] MacKenzie, for example, lists three hundred and ten publications on Fitts' law as of 2002 (`http://www.yorku.ca/mack/RN-Fitts_bib.htm`, retrieved 9th of June 2006).

movement required to the target size. It is often stated as:

$$MT = a + b\log_2\left(\frac{2A}{W}\right), \tag{II.4}$$

with a, b constants, MT the movement time, A the amplitude of movement and W the width of the target. $\log_2(\frac{2A}{W})$ is often referred to as the *index of difficulty* or ID, measured in bits. There is a noticeable (and intentional) similarity to the information capacity of a (Gaussian) noisy channel:

$$C = b\log_2\left(1 + \frac{S}{N}\right), \tag{II.5}$$

where C is capacity, b is the bandwidth and S and N are the signal and noise power levels, respectively.

Fitts' original study examined one-dimensional rhythmic tapping tasks with stylus, where the subject had to alternately tap a pair of conductive bands on a strip. The above equation was found to fit the data very closely. Subsequent studies demonstrated that the equation holds across wide age ranges and over a variety of effectors (fingers, feet, arms, etc.) and under a number of other conditions (large scale and small scale movements, and even in imaginary movements (Decety and Jeannerod [1996]). Schmidt and Lee [2005] summarises these studies.

In the context of HCI research, the law has been found to hold with many input devices. Card et al. [1978] found evidence for Fitts' law behaviour with joysticks and mice, and a huge number of following publications, such as Jagacinski and Monk [1985] (head pointing devices), Marentakis and Brewster [2005] (pointing in 3D audio space), Hoffmann et al. [1995] (keyboards), found similar evidence. Plamondon and Alimi [1997] reviews a number of other studies. The law has also been found to apply in various interfaces with dynamic zooming, or with variable control-display ratios; see for example Guiard and Beaudouin-Lafon [2004b] or Blanch et al. [2004]. Guiard and Beaudouin-Lafon [2004a] contains a number of papers on recent Fitts' law work, and is a good review of the current state of the field.

II.12.1.1 Assumptions in Fitts' Law analysis

The massive evidence for the applicability of Fitts' law under many conditions suggests that it is an important relation in the behaviour of humans in targeting tasks, which are the basis for the design of a great deal of existing computer interactions. For spatial interactions, Fitts' law provides a good model of expected performance, once the constants for a particular system have been experimentally confirmed. However, there are several important questions which arise when considering the applicability of Fitts' law to new interaction techniques.

Fitts law does not generally hold in situations where limb dynamics are not significantly involved (as in isometric joysticks) (Mackenzie [1991]). Given this, it is unlikely that signals from inputs like EEG interfaces will behave in a Fitts manner. It also clearly does not hold in situations where the distance to traverse is long and the velocity of motion saturates; in such cases time to reach distant targets will be dominated by the linear element.

Fitts' law assumes that targets are binary; pointers are either on or off the target. Extending to systems which assign continuous probabilities to the space may be possible, but in this case the contour of the target is given by the posterior probability required to pass an action threshold. Dynamic probability updates which continuously infer goals

based on motion make it difficult to apply Fitts style analyses. A system in which evidence is gradually accumulated as the cursor moves over space is not directly amenable to Fitts' law analysis.

II.12.1.2 DIMENSIONALITY Fitts' original study focused on one-dimensional tapping tasks. There have been attempts to extend the paradigm to include higher dimensional pointing tasks. MacKenzie and Buxton [1992] is one example; this examined several possible *ad hoc* modifications of Fitts' law to fit two dimensional problems. Considering the distance to and width of targets along a vector from a starting position to target centre was found to be a reasonable fit.

Accot and Zhai developed a model of trajectory following based upon Fitts' law (Accot and Zhai [1997]). For trajectories which are constrained within a tunnel having some (possibly varying) width W, the problem of traversing that tunnel can be broken down into a series of target acquisition manoeuvres. Making these target manoeuvres infinitesimal, the following law can be derived:

$$T = a + b \int_C \frac{ds}{W(s)}, \tag{II.6}$$

for some trajectory C, having width $W(s)$ at point s. This law has been found to be experimentally valid for a number of tunnel-constrained path following tasks, including narrowing linear tunnels, and variable width smooth spiral formations. In Accot and Zhai [2002], line crossing tasks were shown to approximately in accordance with Fitts' like models, both for tasks where targets are collinear with the vector between them, and where the vector between the targets and the current position is normal to the vector between the targets.

Grossman and Balakrishnan [2005b] describe a probabilistic model for two-dimensional target acquisition, consistent with Fitts' law, approximating the distribution of terminal points near a target with a bivariate Gaussian. The model is shown to be consistent with experimental results, and also demonstrate that both dimensions of a two dimensional target (i.e. both the direction normal and parallel with the direction of movement from the starting position) have an impact on the movement time.

II.12.2 INFERRING A CLASS OF OPERATOR MODELS

Fitts' law is intended for quantifying pointing task behaviour. It is a model for analysis, not a generative model: it does not answer the question of what time series would be expected given an attempt to acquire a particular target. The law, is however, compatible with simple operator models.

Jagacinski and Flach [2003] show that Fitts' law is compatible with a second order response. Beamish et al. [2006] suggest that Fitts' law can be explained as a function of neuro-muscular delays. Card et al. [1983] give a (rather biologically-implausible) derivation based on iterative discrete approximations. Phillips and Repperger [1997] suggests that Fitts'-like responses are the result of transitions between proprioception-loop and visual-loop control:[23]

> In many cases, movements in single step tracking which must be both large and accurate will follow this two-phase process. The first phase of the movement is a rapid, open-loop ballistic movement, (as though a successive approximation strategy is predominant). However, the second final phase of

[23]The control referred to here as "open-loop" is still closed-loop, but the loop is proprioceptive rather than visual.

the operator response, is characterised by a slower visually monitored positioning movement (characteristic of the closed-loop strategy).

Schmidt and Lee [2005] discuss a number of other possible theories of motor control which explain Fitts'-like behaviour in targeting motions.

Identifying generative models of human tracking behaviour is more valuable than quantifying their effect. It permits the design of interfaces to match the possible responses of the user, rather than trying to optimise existing structures to satisfy a criterion. Chapter VI illustrates how this can be used to design entirely new interaction techniques.

II.12.3 The Applicability of the Law in General Interfaces

Fitts' law is a useful bound on the limits of spatial interfaces based on fixed activation regions. Distance to target or target size must be traded off to optimise the layout of interface. However, in more general interfaces, what is required is a generative model of the behaviour likely to be generated given an intention. Some simple operator models have been examined in the manual control theory literature (Section VI.4.1.1 reviews several of them). Such models make it possible to simulate behaviour in an interaction to quantify potential system performance.

II.13 Conclusions

The preceding sections have outlined the basic purpose of human-computer interaction – to communicate intention in a way that increases the utility of a system. To do this, a system must estimate the hidden intention states of the user. The problem can be viewed as a continuous control process where the interactor attempts to steer the system's beliefs about intention, forming a trajectory in *goal space*. Control in the goal space is not direct – the dimensionality is far too high – and instead the process is extended over time via mediating dynamics which an interactor can directly control. The interaction process can be viewed as a very general hierarchical control problem, with loops operating at a range of timescales. Feedback is an essential part of interfaces. It reduces the need for complex inference models in the system, and simplifies the models users must make of the system.

Conventional, spatially-oriented interactions can be interpreted sensibly under this model. However, it is apparent that such interfaces do not make the best possible use of bandwidth. In particular, the temporal aspects of interaction are largely ignored. They also fail to accumulate evidence rationally; although evidence for intention is often observed, specific terminal events are used to induce state changes whatever the state of the evidence.[24] The system itself may better be able to estimate the probability of intention as it varies over time. Part of the reason for the necessity of features such as undo is lack of control during evidence acquisition.

The signals that flow through the control loop are uncertain and are delayed by varying amounts. Systems must deal with uncertainty and delay; these can be reduced but not removed by better design. The following chapter will discuss how feedback can accurately represent uncertainty; the subsequent discusses how predictive displays can mitigate delay issues.

[24] Obviously such an event is evidence itself, but it increases the certainty of the system only by some level – it does not guarantee absolute certainty.

CHAPTER III

DYNAMIC PROBABILISTIC FEEDBACK

Clouds of sound to reveal the inferential mechanics.

> *But we don't really perceive sound as little bubbles coming into our ears, do we? Let's answer an easier question: Are there particles of sound production? Of course!*
>
> – Cook [2002]

III.1 SUMMARY

THIS chapter explores the issue of uncertainty in the display of results. The importance of real-time, relevant feedback is discussed, and, in particular, the role that appropriately uncertain display has. Granular synthesis is introduced as a suitable way of implementing uncertain displays in audio user interfaces. Structures for using these displays with existing inference mechanisms are described, and a number of implementation examples are discussed to illustrate the operation of the techniques.

III.2 FEEDBACK; A REFLECTIVE INTERFACE

It is almost always necessary to display the result of interactions, for otherwise little benefit can be gained from the interaction. But while this "display of results" is clearly vital, it is also arguable that *process feedback* is essential to a usable interface. Such feedback relates the changing state of the system purely for the purpose of improving the interaction; it provides no external utility. It is this continuous display which makes a control-based interaction model possible, binding the user into a negotiation loop with the system, and consequently splitting and sharing the inference problem between the system and the user. The benefit is twofold: the system can operate with less sophisticated inference (because the interactor has control before decisions are made); and the user can rely less on memorised sequences of behaviour to effect actions (as they can directly observe the result of their input in time to correct or adjust it).

Much research has been concerned with the development of transparent interfaces (Tanimoto [2004], Bardram and Bertelsen [1995]). The basic principle of this approach is to ensure that interaction is so direct that the result of actions is obvious, and a user intuitively understands the responses of the interface. In Tanimoto [2004], it is noted that:

> A transparent interface reveals some of the inner workings of a system. Transparency is important in computer systems that must engender trust, that teach, or that must rely on users to correct their mistakes.

III.2.1 Exposing the Interface Mechanics

One of the most significant difficulties with introducing "intelligence" into an interface is the loss of user intuition. As discussed in Section II.11, a button has a very simple physical manifestation, and directly mapping elements of the physical state to the intentional state of the system is obvious. It is, however, a suboptimal use of the possible communication bandwidth, and does not provide real-time *control* over the system.

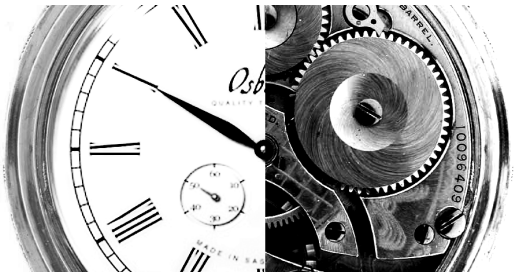


Figure III.1: Exposed mechanics. Good feedback should reveal the workings of a system in a way users can model.

In this thesis, it is suggested that making interfaces *reflective* – continuously displaying the system's beliefs about the user – can be of value in increasing usability. This chapter focuses on the process feedback in interactions, updating the user on the changing state of the interface inference engine. It specifically addresses the problem of uncertain display. Where inference produces an uncertain result – which it always should in an interaction task, for sensors never perfectly reliably indicate intention – the problem is how to display the range of possibilities and these change during an interaction.

III.2.2 Available Modalities and their Properties

There is a number of modalities available for the presentation of feedback to help the user control the system. In practice, only the visual, auditory, vibrotactile and force-feedback (kinaesthetic) channels are currently feasible for computer displays.[1]

Visual displays are the most widespread of all displays, and are often the only or the dominant way of receiving feedback from a system. They potentially have a very high bandwidth, and can easily display complex data. It is often natural to rely on visual display as the primary output medium, and there is a cornucopia of techniques for displaying data visually. However, in some contexts it can be a poor choice for feedback, such as where the eyes are otherwise occupied, or where a spatially oriented display is unnatural.

This is especially true in mobile situations where visual attention must be focused on the environment, rather than the interaction at hand. Such contexts comprise many of the scenarios in which novel devices are likely to be used; mobile phones or personal digital assistants, for example. It is also an unsuitable modality where low-latency is required, as visual responses take significant time to process (Rousselet et al. [2002] suggests a 150ms response), introducing long delays in the control loop.

[1] Although olfactory displays may be viable for display in the near future: see e.g. Washburn and Jones [2004] or Yanagida et al. [2003] for some examples.

Audio is a natural choice for display when visual display is unavailable, or for augmenting existing visual displays. It has a high potential bandwidth for communication, relatively low cognitive processing latency, and technology for audio reproduction is well developed. Audio is particularly suited to temporally-oriented displays where much finer grained temporal resolution is available than with a visual display (sub-millisecond resolution as opposed to $\sim$50 millisecond for visual display). An extensive literature is dedicated to the general sonification of data sets, displaying salient attributes via audio. Kramer and Walker [1999] reviews the field. Brewster [2003] is a more general review of auditory output for HCI. Sonification can have significant benefits even where visual displays are available, as Hermann and Hunt [2005] note:

> Regarding the latter point, auditory displays offer an interesting complement to visual displays. For example, an acoustic event (the audio counterpart of the graphical symbol) can show variation in a multitude of attributes such as pitch, modu- lations, amplitude envelope over time, spatial location, timbre, and brightness simultaneously. Human perception, though, is tuned to process a combined audiovisual (and often also tactile and olfactory) experience that changes instantaneously as we perform actions. Thus we can increase the dimensionality further by using different modalities for data representation.

The advantages of auditory display in user interfaces have been investigated in some detail; e.g. with earcons (Brewster et al. [1993], Brewster [1997]) or auditory icons (Gaver [1986], Mynatt [1997]). Brewster [1997], for instance, demonstrates the efficacy of auditory interfaces in mobile contexts, augmenting a conventional visual interaction by providing summative feedback on button press events.

There is somewhat less research on auditory interfaces in continuous control contexts (outside of musical applications).[2] Some classical examples are reviewed in Poulton [1974]; these are generally based around pitch modulation (e.g. Milnes-Walker [1971]) or interruption rate (for example, for error display as in Ellis et al. [1953]). An early complete integrated audio feedback system for aircraft control is described in Forbes [1946], which was sufficient to allow inexperienced pilots follow straight paths in a mechanical flight simulator. Vinje [1972] (cited in Sheridan and Ferrell [1974] p.263) developed an audio feedback system for V/STOL hover to augment conventional instruments. The heading, height and position of the aircraft were sonified; this is reported to have reduced the pilots subjective workload. The limitations of the technology at the time of these studies limited the sophistication of the resulting systems. With the introduction of affordable digital signal processing, much richer synthesis is practical. In more recent work, physically-modelled synthesis for continuous control of a rolling ball on a beam is described in Rath and Rocchesso [2005]; Rath [2004] has detailed analysis of the performance of the rolling ball system and the effects of the feedback on control behaviour. van den Doel et al. [2001] describe a real-time system for the generation of realistic slipping and sliding sounds, while Essl and O'Modhrain [2005] describe a friction based sonification for the perception of motion. Perry Cook's STK sonification toolkit (Cook [2002]) includes a number of models suitable for synthesis of continuous processes. All of these can be applied to display the state of continuously varying dynamical systems.

[2] Johannsen [2004], for example, contains a number of works on continuous control in a musical context.

III.3 UNCERTAINTY IN FEEDBACK

Section II.10.1 discussed the key role of uncertainty in user interfaces, and some common sources of noise in the interaction process. An interface should maintain and propagate uncertainty; it should also display uncertainty to the user to provide enough information for reliable control.

III.3.1 OPTIMAL CONTROL WITH UNCERTAIN DISPLAY

There is evidence that displays which represent uncertainty lead to appropriately regularised behaviour. Kording and Wolpert [2004], for example, showed that in a target tracking task, where a disturbance was artificially added to the user's control input, uncertain display (Gaussian point clouds) lead to behaviour that was consistent with Bayesian integration of the uncertain sensory inputs and prior beliefs about the system. Operator models, such as those proposed in Jagacinski and Flach [2003], which incorporate Kalman filters imply that uncertainty is modelled in user control behaviour. Section IV.4 demonstrates experimentally the regularising effect of uncertain display in orientation acquisition tasks.

Failure to incorporate uncertainty can certainly lead to poor decision making. The grounding of the *Royal Majesty* on the 9th of June 1995 was a direct result of the crew relying on a GPS navigation system which showed apparently accurate information (visual display as the ships position as a definite point, numerical display to six significant figures) despite not having accurate measurements (National Transportation Safety Board [1995], Degani [2004]). Unbeknownst to the crew, the GPS system was operating in dead-reckoning mode, and accumulated error rapidly as the ship travelled. However, the GPS continued to display the position with absolute accuracy. As a result, the ship ran aground on rocks which were clearly visible on the display – which the crew believed were a safe distance away. Don Norman (Norman [2006]) has argued that the fate of Southwest Airlines Flight 1248 (which ran off the runway due to misestimation of the available length) was due to similarly poor displays, giving apparently rock solid results despite known underlying uncertainty.

III.3.2 EXISTING TREATMENT OF UNCERTAINTY

Few interfaces provide any form of uncertain display. One of the few treatments of the subject in HCI is the studies of ambiguity in user interfaces given in Mankoff [2001] and Mankoff et al. [2000]. This focused around building interface toolkits for supporting correction and disambiguation of input from recognition systems. It does not deal with probabilistic interpretations of input, and is not aimed at continuous interaction interfaces; the interactions are focused around recognition/correction steps.

Rosenstein et al. [2005] and Rosenstein et al. [2004] discuss *intention* feedback as well as objective feedback in robotic displays, although they do not deal with uncertainty in intention estimates. Displays which show how the visible space corresponds to the goal space are examined; these consist of overlaid "funnel" images indicating regions of the space which the system will associate with a goal.

Displays of uncertainty have also been used for visualisation of static data. In static information visualisation contexts, vibration has been used as a cue for the display of uncertainty (Brown [2004]). There have been several attempts to introduce uncertain displays in GIS systems with attributes such as colouring, brightness or symbols representing uncertainty, and also with Monte Carlo sampling approaches (Hope [2005],

MacEachren et al. [2005], Leitner and Buttenfield [1997], Goodchild et al. [1994]). Grigoryan [2004], for example, discusses the visualisation of uncertain three-dimensional surfaces, using point clouds to represent areas of uncertainty. Griethe and Schumann [2006] and Johnson and Sanderson [2003] review a number of existing techniques for uncertain display in information visualisation.

III.3.3 Motivation of Particulate Representations

The true densities involved in interactions are often analytically intractable, except for the very simplest cases, and it is often more reasonable to draw Monte Carlo samples from the densities and work with these instead. In some cases the inferential process is itself at least partially particulate (e.g. in particle filters) – in this case a collection of samples completely represents the state of the inference. This representation has the advantage of being easy to feedback to the user in many modalities; in a visual case this might be presented a cloud of points instead of a single pointer. A direct auditory analogue of visual point clouds is *granular synthesis*, the main focus of this chapter. Granular synthesis can be extended to vibrotactile-tactile and force-feedback haptic channels (see Crossan et al. [2004]), but the specific asynchronous algorithms presented here may be less suited for such modalities due to the phase randomisation effects introduced by the granulation process. Synchronous techniques which emphasise phase relations or spatial distribution over spectral content may have better performance on such modalities.

III.3.4 Visual Uncertain Display

Adding uncertainty to a visual display is often simple. Points can be replaced by point "clouds", each a sample from a distribution. Greater numbers of points lead to more solid and less flickering displays, but require additional computational power, and may saturate the display. Alpha blending and anti-aliased drawing can be used to give a better sense of the density of particles (at the cost of some computation). The flickering motion effects of a Monte Carlo sampled visual display may also be useful cue. For situations where the point density can be evaluated quickly or pre-computed, the density can be regularly sampled and displayed directly as an alpha overlay. For simple densities, such as for the Gaussian case, probability isocontours can be displayed (e.g. the two standard deviation bound).

In some situations, for example in map navigation, there is a static world, and the relationship to that world is unknown (e.g. the position relative to contours). An uncertain display could show the position estimates as described above with point clouds, or alternatively the static world could be convolved with the position density before displaying (i.e. using the density as a point spread function), while the position remains a point display. [3] Figure III.2 compares these displays for visual map navigation.

[3] Although in practice some hazard or utility map should be made before convolving so that important but small details are not eliminated from the display – e.g. tiny rocks in an ocean navigation problem.

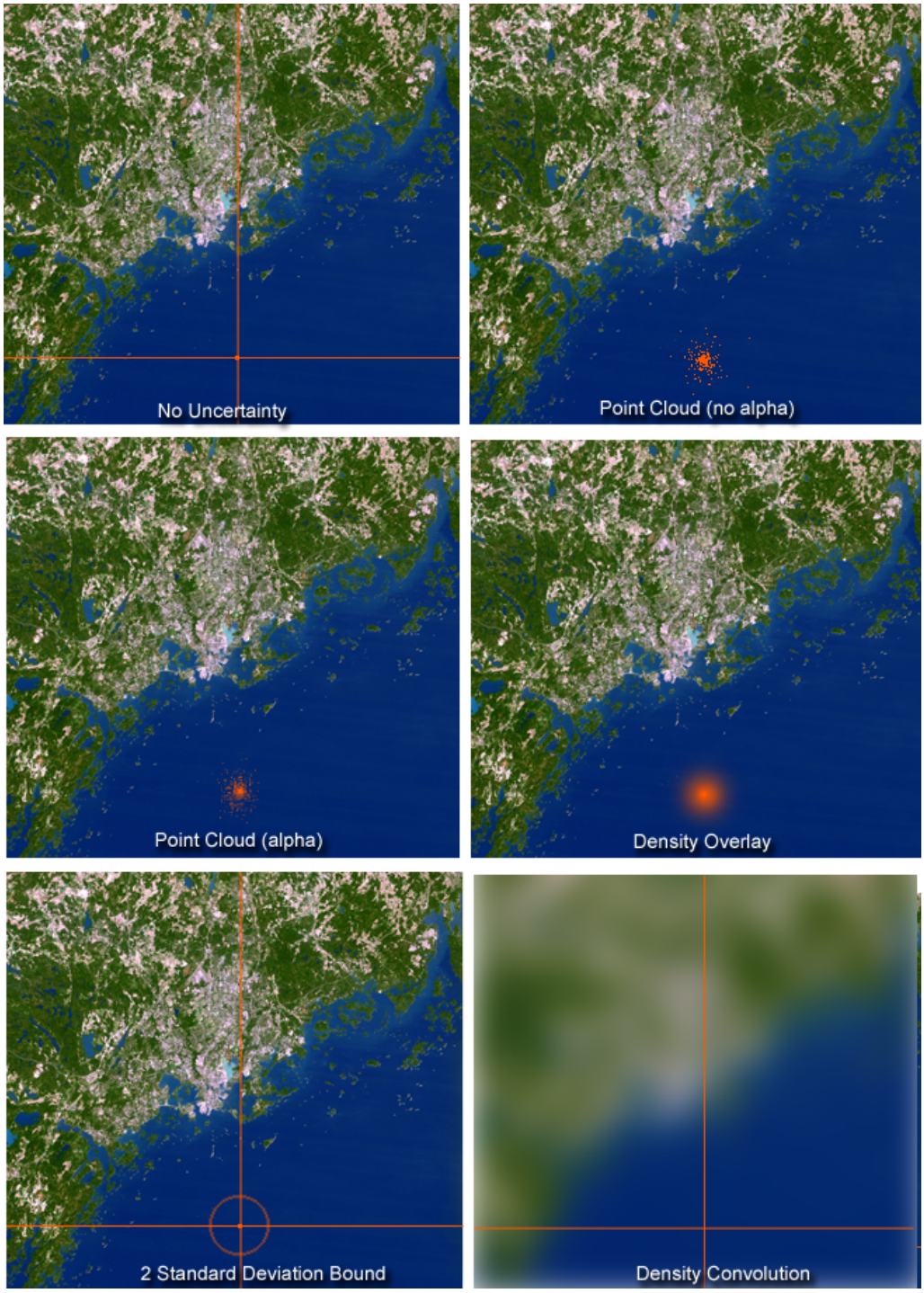


FIGURE III.2: Some postulated visual uncertain displays in a map navigation example. Left to right, top to bottom: Single point, no uncertainty; point cloud (no alpha); point cloud (alpha blended, anti-aliased); density alpha overlay; outline of two standard deviation bound; world convolved with position density.

In the visual domain uncertain displays are easy, if sometimes computationally expensive, to implement; however it is less obvious how to build uncertain auditory displays. The next sections show how granular synthesis can be used for uncertain display, *with respect to the goals of the user*.

III.4 Introduction to Granular Synthesis

III.4.1 Granular Synthesis; A Brief History

Granular synthesis was first described by composer Iannis Xenakis in the early 1950's (although the techniques are very similar to those proposed by Gabor for signal analysis in 1947 Gabor [1947]). Xenakis was interested in producing music by mathematical process; to this end he described a method of producing sound by choosing small sound segments and arranging them according to probability distributions. At the time, this was performed manually with magnetic tape. The details are described in Xenakis's book, Formalized Music (Xenakis [1971]). One of the best-known early works featuring granular synthesis is *"Concret pH"*, featuring extremely short samples of charcoal burning sequenced together into a four minute piece. This was originally played on an array of several hundred speakers in the Phillips Pavilion at the 1958 World Fair in Brussels, making use of the dense spatial textures granular synthesis can provide. Childs [2002] presents an analysis of Xenakis' *"Achorripsis"* which involves probabilistic granular synthesis, and is in effect one of the earliest examples of mathematical sonification. The technique is somewhat similar to the "micropolyphony" process employed by Ligeti in his famous 1961 work *"Atmospherés"* (the opening music for *2001: A Space Odyssey*), where the each member of the orchestra has a slightly different score, giving a dense, rich structure.

These techniques became much more practicable with the advent of affordable DSP units, and Barry Truax (Truax [1986], Truax [1988]) and Curtis Roads (Roads [1978]) implemented the techniques on computerised hardware and brought them to a wider audience. There are now a proliferation of variants upon the technique, almost all of which are focused on the production of novel musical textures. An comprehensive overview of the history and current state of granular synthesis related work is given in Microsound (Roads [2002]).

III.4.2 The Granulation Process

The basic thesis of granular synthesis is that a clear macroscopic structure can be created from a mass of microscopic events. These events are far too numerous to be specifically hand selected; they are instead generated by some mathematical process. Originally, (and in this work), these processes involved sampling from statistical distributions. The technique can, however, be extended to other parametric processes generating large quantities of unique but related events; in Miranda [2000] and Bowcott [1989], for example, cellular automata are used as a generating mechanism for evolving sound textures. Informally, the granulation process divides up some original sound waveforms into small sections, envelopes them with windows to eliminate clicking artifacts and sums a great number of these together continuously to produce audio output. The waveforms from which these slices are taken, the specific regions which are selected and parameters of any post-processing to be performed (such as resampling or spatialization), are all drawn from some distribution. Normally, there are a finite number

of source waveforms, which could, for example, correspond to goals in an interface.[4] Figure III.3 illustrates the basic elements of the process.

[4]Although if the grains are synthesised in real-time from some parametric synthesiser (e.g. from a waveguide model) the densities can be continuous over the synthesis parameters.

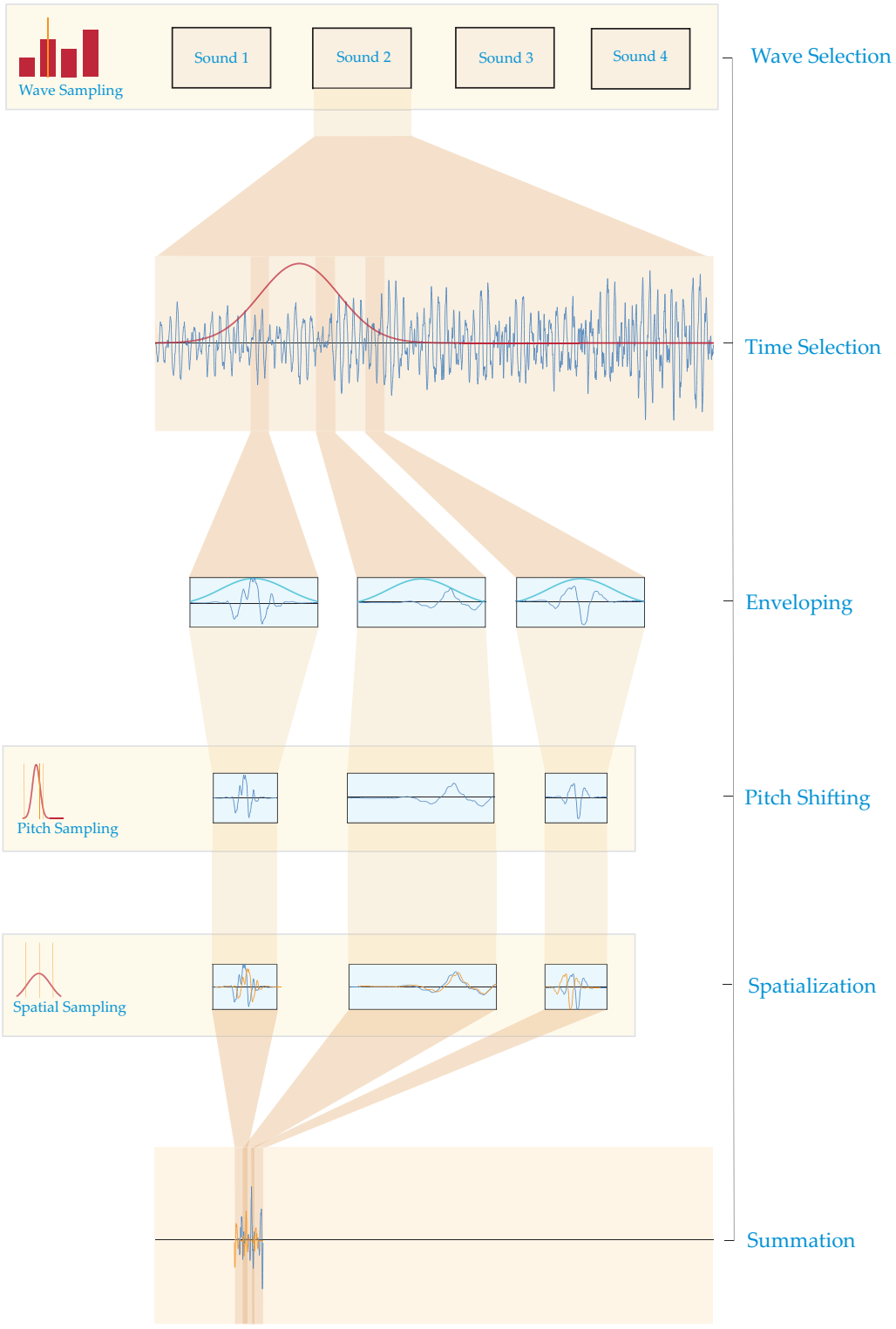


FIGURE III.3: The granulation process (shown here for three grains). A number of slices are taken from a set of original sounds. Sample positions are chosen by some process, here by sampling from a p.d.f. These slices are then enveloped (to eliminate clicking artifacts), pitch adjusted and spatialized according to the densities for each parameter. All of the slices are summed together to produce an output sound stream. In practice many hundreds of grains are active simultaneously.

More formally, the asynchronous granular synthesis process, as used in this work, can be defined as follows:

III.4.2.1 PER-SOURCE PARAMETERS

There are assumed to be a collection of source waveforms $w_1 \dots w_n$. For each waveform w_i, at some time t, there will be:

- an associated probability p_i;
- a density over time $t_i(x)$ which lies in $[0, l(w_i)]$, where l_{w_i} is the length of a waveform;
- a density over resampling rates (i.e over pitch shift), $s_i(x)$ over $[0, \infty]$;
- a density over spatial positioning, $a_i(x)$. For stereo, without loss of generality this can be defined to be a density over the range $[-1, 1]$ where -1 is hard left, 1 is hard right. For 3d spatialization this can be any density over $\mathbb{R}^3$. For other spatialization techniques (speaker arrays, as used in surround sound implementations), the density will be over an n-D v vector with each element in range $[0, 1]$.

One source, at a particular time, can be represented as the quintuplet $c_i = < w_i, p_i, t_i(x), s_i(x), a_i(x) >$. Very often the densities $t_i(x), s_i(x), a_i(x)$ will be chosen to be either mixtures of Gaussians or delta functions, simplifying the possible representations.

III.4.2.2 GLOBAL PARAMETERS

It is assumed that there is a single global window length l and a windowing function $v(t)$ over $[0, l]$, which applies to every waveform. It is possible to introduce these as a per-source parameter or a full per-source distribution (although a parametric form for $v(t)$ is needed), but this prevents a significant optimisation of the algorithm (see Appendix A for details). Some possible window functions are shown in Figure III.4.2.2, the Gaussian[5] and piecewise-linear windows being particularly useful. These are the windows implemented in the GSLib library (Appendix A). The length l is chosen so as to be long enough to give a sense of the timbre of the source, but short enough that the process does not become simple playback, and the textural aspects of the synthesis technique remain audible.

The other important global parameter is the number of grains involved. In the algorithm used here, there is an upper bound g on the maximum grains simultaneously played back and a probability of grain generation p_g; any number of grains less than g can be active simultaneously. p_g gives the probability of a grain being generated at any time point and thus defines an exponential distribution over the inter-grain-generation time. Increasing g and/or p_g increases the "smoothness" and continuity of the sound, but computational cost obviously increases. Very high densities lead to noise-like results.

III.4.2.3 GRAIN INFORMATION

The algorithm uses a set of grains $r_1 \dots r_g$, of which some number belong to the active set R_a. Each grain r_j encapsulates:

- its total length $l(r_j)$;
- its associated source $w(r_j)$;

[5]The Gaussian window is of course truncated to fit the grain length. Other similar windows such as the Kaiser, Hann or Hamming windows could be used instead.

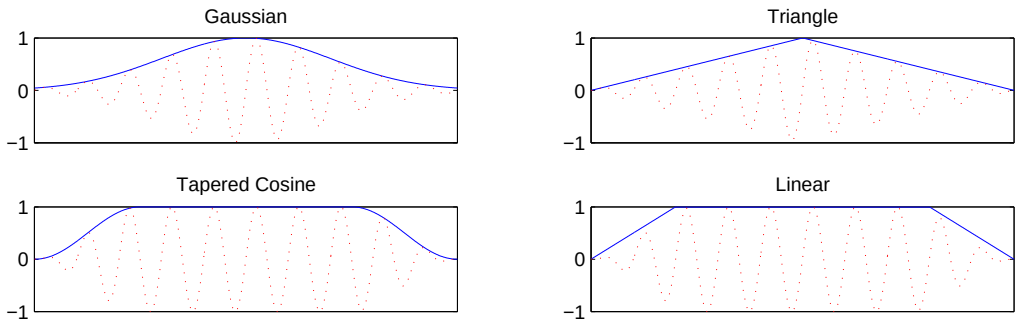


FIGURE III.4: Some possible enveloping windows for granular synthesis. These are applied to waveform sections to eliminate clicking artifacts. The effect of windowing a sine wave is illustrated.

- its current pointer into the waveform $pt(r_j)$;
- the source parameters $t(r_j), s(r_j), a(r_j)$ which were drawn from $t_j(x), s_j(x), a_j(x)$ at generation;
- the amount of the grain already produced $o(r_j)$;

III.4.2.4 ALGORITHM The synthesis algorithm can be summarised as follows:

1. An new buffer of length b is received and cleared.
2. For each grain r_j in the active set R_a:
   (a) For every sample i in b:
   (b) $pt(r_j) = pt(r_j) + s(r_j)$;
   (c) $b(i) = b(i) + w(r_j)[pt(r_j))] \times v(t)a$;
   (d) $o(r_j) = o(r_j) + 1$;
   (e) if $o(r_j) > l(r_j)$:
      i. remove r_j from R_a;
      ii. if $|R_a| < g$, with probability p_g:
         A. reset $o(r_j)$;
         B. draw a new source c_i from $c_1 ... c_n$, weighted by p_i;
         C. draw a new set of parameters t, s, a from the densities given by c_i;
         D. add r_j to R_a.
3. transform b to audio output buffer format.

III.4.3 EFFECT OF GRANULATION PARAMETERS

Complete discussion of the various variants of the synthesis technique and the general effect of parameters on the sound can be found in Roads [2002]. For the specific asynchronous algorithm described above, the following general effects may be noted:

- **Grain Length** Longer grains produce a smoother, cleaner sound with less of a distinct texture. Very long grains (>1s) include so much of the original that the granular effect is often lost. Shorter grains provide a dense textured feeling. Very short grains (<50ms) produce a distinct gritty texture and has little of the source character.
- **Grain Density** High grain densities imply smooth, thick sounds. Lower densities have a sparser feel, similar to that of a Geiger counter.
- **Envelope** Gentle envelopes mask some of the granular texture, and also reduce the output intensity; sharper ones emphasise the granular texture.
- **Temporal Distributions** Wide temporal distributions have an effect similar to that of long reverb, with a wide echoing sound; narrower distributions focus the sound more clearly.
- **Pitch Distributions** Wide pitch distributions produce a harsh, dissonant sound, that quickly turns into white noise. Mixtures of delta distributions (e.g. in a chord pattern) are much more pleasant to the ear.

III.4.4 Why Granular Display?

It may seem that similar effects might be had by associating a sound stream with a particular goal, and simply modulating its intensity by the probability of the goal. However, granular synthesis has a number of advantages over this approach.

First of all, granular synthesis makes temporal manipulations easy (see III.5.6), a technique which is hard to replicate without some form of granulation (although spectral methods like the phase vocoder – for example the stretcher given in Ravelli et al. [2005] – can be used in some circumstances). Many time-stretching algorithms are based on some form of synchronous granular synthesis (see, for example, Itagaki [2000] and Truax [1994]), often combined with specialised transient analysis to retain the structure of the fast onset events which are important in recognition of sounds. Pitch-synchronous-overlap-add (PSOLA) is a popular technique for time-stretching speech (Moulines and Charpentier [1990]); it is a synchronous granulation process with grain size locked to the estimated fundamental frequency of the audio. Granular synthesis can produce time distortions which are fully probabilistic in nature, an effect very difficult to achieve without granulation.

Secondly, granular output can be used for very low probabilities, producing sparse snatches of sound which are easily identifiable, due to the ears extreme sensitivity to the onset of events. Geiger counters, for example, can represent activity over a very wide range, from extremely sparse events to dense trains of pulses. Granular synthesis affords smoothly and intuitively changing *textures* of audio as probabilities vary, rather than simple intensity adjustments. In contrast, directly mapping volume to probability makes it very hard to hear low probability sources. It should also be noted that granular synthesis is scalable to very large numbers of potential sources, as the computation effort expended on each source depends on its probability. Low probability sources require little computation time. The naïve approach would require mixing of all streams with non-zero probability at all times.

Thirdly, granular synthesis provides a simple way of synthesising sounds with pitches distributed according to some p.d.f. This would be extremely hard to do with multiple continuous streams of sounds (a very large number of simultaneous streams would be required to get the effect – effectively granular synthesis with infinitely long grains). This can produce dense textures with interesting structure.

Finally, the granulation process has a number of parameters that can be configured to effect different textures of sound, ranging from clicking Geiger counter-style effects to dreamy, reverberant sequences. These offer the interface designer a wide palette of sonorities to tailor to a particular interface.

III.4.4.1 Direct Dissonance Models

One alternative to granular goal display is to display the entropy of the distribution via dissonance. This technique is described in Williamson and Murray-Smith [2002]. Briefly, each probability is mapped to a sound with a clear pitch, the intensity of the sound being related to the probability. The pitches are chosen so that they occupy a fairly narrow frequency band. As the entropy increases, more pitches are apparent within the output, and the dissonance increases, reflecting the increasing confusion in the inference.

This technique is in fact a special case of granular synthesis, with a single source whose pitch distribution is being manipulated. However, granular synthesis offers more sophisticated sonification, permitting more flexible pitch densities and densities over space, time and timbre.

III.5 Direct Mappings of Probability Distributions

For a simple case where there is a discrete number of potential goals, each goal can be paired with a source wave. The probability of each goal maps directly to the probability of drawing a grain from its associated source. A hierarchical menu can be decomposed in this way, each higher level element being a mixture of its sub-elements, weighted by the sub-element priors. As the user navigates down the tree, the sound converges from a *mélange* of elements to less dense mixtures of elements, until finally a single source remains. Any interface which features hierarchies of elements with varying probability can be sonified in this manner.

III.5.1 Example: Application to Traditional Gesture Recognition

A simple application of the type of feedback is in sonifying the output of a probabilistic classifier, such as an HMM, where there are a number of sub-models m_i, each of which has an updated likelihood $p(\theta|d)$ at every time step. This pattern often occurs in symbol recognition systems. It is, for example, a common structure in gesture recognisers (e.g. Wilson and Bobick [1999], Nam and Wohn [1996], Rigoll et al. [1998]). Figure III.5 shows the basic structure of a sonification of this structure. A full particle-filter based gesture recognition utilising this technique, along with temporal manipulations to represent phase display, is given in Section IV.5.

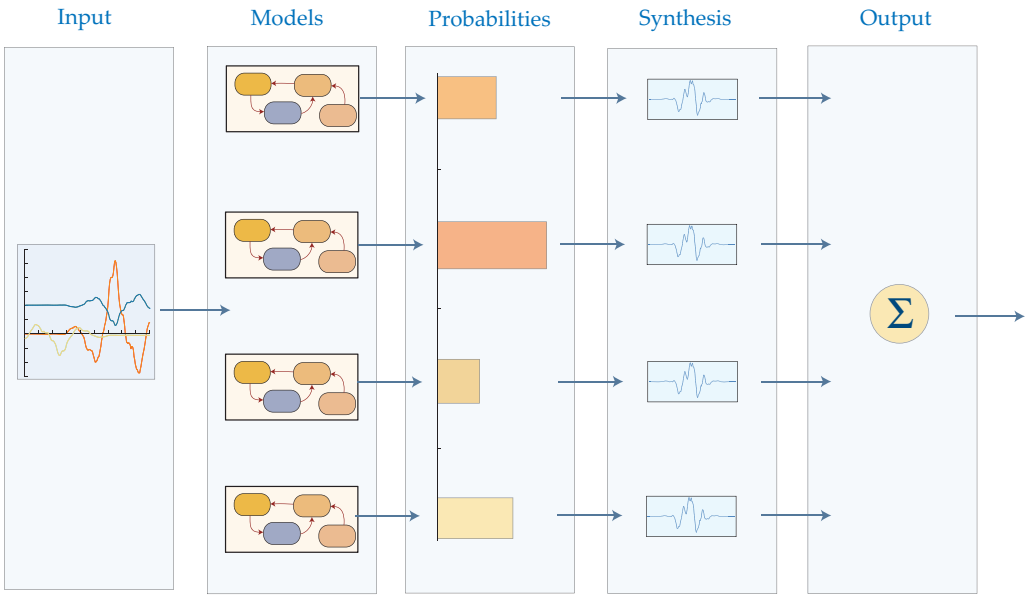


FIGURE III.5: Multiple models in a recognition task are directly sonified via granular synthesis.

III.5.2 IMPLEMENTATIONS: THE GRANULAR SYNTHESIS LIBRARY

The Granular Synthesis Library (GSLib) is a collection of C routines for synthesising asynchronous granular audio in real-time. It is non-system dependent – it simply fills a series of output buffers when required. It generates a continuous stream of grains, with adjustable grain length, density, and source waveforms. The distribution parameters for source selection, pitch, panning and time can all be adjusted in real-time. This makes it straightforward to add synthesis capabilities of the type described in this chapter to many systems with little effort, provided the existing system is already probabilistic in nature. The library is discussed more fully in Appendix A. This code was used to implement the examples described in this chapter.

III.5.3 STATE SPACE REPRESENTATIONS OF INTERFACES

As described in Section II.9, taking a state space view of an interaction, a set of n densities $f_1(x) \dots f_n(x)$ can be defined over that space. These are used to define a function from the state space $s \in \mathbb{R}^n$ to a probability distribution $p_1 \dots p_n$, a point in some goal space. These output probabilities map in an extremely straightforward way to the granular synthesis process; each density has an associated sound, and the probability of drawing a grain from waveform i is just p_i.

III.5.4 EXAMPLE: STATE SPACE MAPPING

As a demonstration of this type of approach, placing densities directly on the state space of a system, a simple targeting/exploration environment featuring granular synthesis has been constructed. This example places densities directly onto the sensor space, and produces audio based upon the activation of these densities as a cursor traverses the space.

A screenshot from the implementation is shown in Figure III.6. This is implemented with the granular synthesis library GSLib (see Appendix A). In this example, a number of Gaussian densities are distributed in a two-dimensional space, having means drawn uniformly across the rectangular area and identity covariances. Each density has a waveform associated with it, labelled at the modes on the diagram.

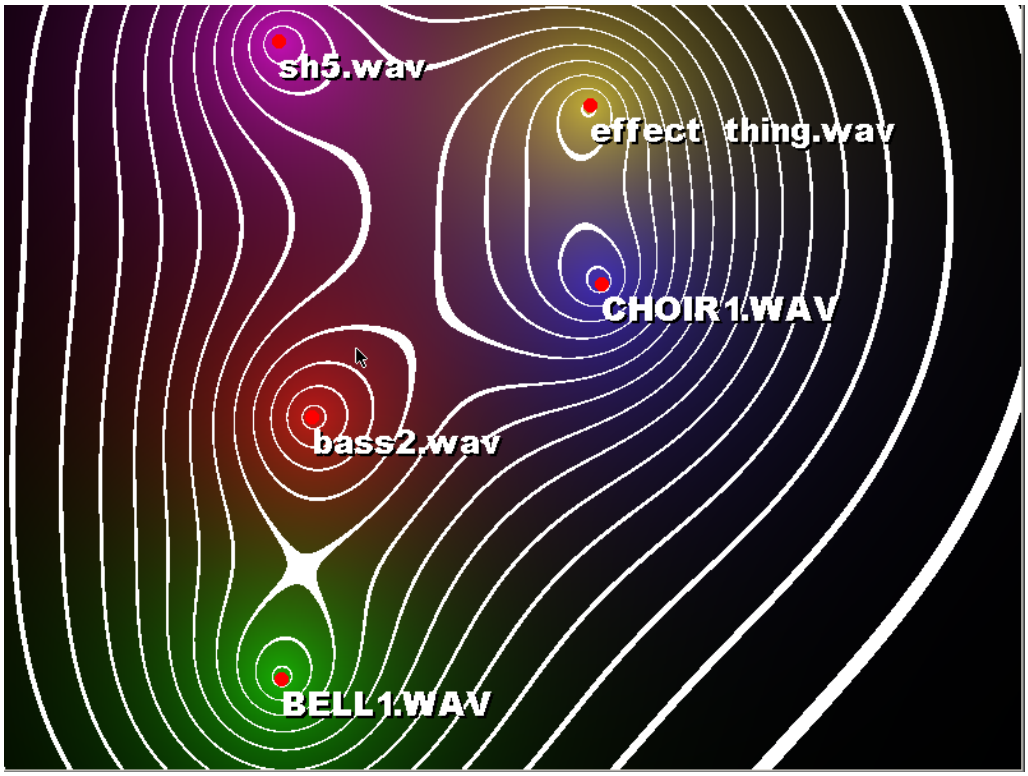


FIGURE III.6: Image from the mixture of Gaussians demo. A number of Gaussian densities are arranged in a 2d state space, each with its associated waveform. The probability of each source given the current cursor position is sonified. In this image, isoentropy contours are shown. The colours indicate the mixture of probabilities at that source.

As the mouse cursor is moved through this space, the probability p_i of each spatial "goal" is computed and normalised. These probabilities are fed to the synthesis mechanism, which produces a smoothly varying sound with a relatively constant texture. Temporal distributions in this example are uniform over the entire waveform, with no pitch or spatialization used. Unprocessed sections of music or sound effects are used as the input waveforms; the granulation process creates smooth textures from these automatically. See Appendix C for videos and audio demonstrations of these effects.

III.5.5 Display of Entropy vs. Display of Distribution

In systems such as the above example, where the complete distribution of goals is sonified, the prominence of each goal is related directly to the prominence of its associated sound. In cases where the distribution is does not have a single clearly probable value,

the sound diffuses into a mixture of all possible goals.

Consequently, the sonification emphasises the *entropy* of the distribution when entropy is high, and the identity of the source when entropy is low. When the waveforms have different pitches, for example, higher entropy manifests itself as more dissonant output. This is logically consistent with the desired behaviour of a probabilistic sonification system; since the user cannot hope to perceive the full distribution in all its detail, the sound simply becomes more blurred and noisy as the entropy increases. A single item, or even two or three, of high probability is still easily identifiable in the mix. The appropriate level of information for control of intention is produced automatically.

III.5.5.1 Importance of Change in Entropy

The change of entropy of the goal distribution is the rate at which the user is communicating with the system – knowledge of the dynamics of the system entropy gives the user awareness of the effect of their communication. From the user's point of view, one of the most important indicators in the interaction is how entropy changes over time: "Am I getting closer to a goal or not?" – "Am I communicating well?". A key function of an uncertain display is therefore to feedback the rate of effective communication to the user.

III.5.6 Probabilistic Temporal Audio Manipulation

One of the attractive features of this synthesis type is the immediate derivation of temporal manipulations which would otherwise have to be implemented separately. The apparent "speed" of a sound can be altered without distorting the pitch, in contrast to naïve adjustment of the sample rate. Time can be stretched, for example, simply by defining a narrow density on the sample indices within a waveform and translating its mean by some constant c at equally-spaced intervals t_n, of duration Δt where $c < \Delta t s$ (s being the sample rate). Time compression occurs when $c > \Delta t s$, and the situation where $c = \Delta t s$ is the *natural output rate*.

Simple "melody completing" type feedback for progressive feedback during an interaction task is trivial with this approach. It is in fact more flexible than the common method of synthesising a tone sequence from a predefined template (e.g. via a MIDI synthesiser), playing notes at specified intervals during a task. Instead, the granular technique produces a smooth, continuously changing sound, in which it is feasible to "hold time still" and still have meaningful output. Equivalently, apparent time rate can be fixed and the apparent pitch of the sound altered by resampling. This offers a flexible way of manipulating the audio stream, and can be used to introduce new timbral effects by employing complex pitch densities.

Since time is represented as a density, uncertainty about the current state of some time-dependent process can be displayed. In a gesture recognition system, for instance, the system may have uncertainty about the phase of the gesture. If each gesture model has an associated waveform, as in III.5.1, the density over samples in that wave is obtained directly from the phase density. This can be used to create feedback which relates both the distribution over classes and the distribution over phases in a classification problem.

A fundamental use of temporal distributions is in "motion blurring" highly time-compressed audio streams to preserve the audio flow and avoid introducing sharp pulsing artifacts in the output. If a very narrow density over samples is used, and the audio is compressed at some rate much greater than the natural output rate, sections of the audio are skipped due to the quantization of time into frames (see III.7(a)). This is analogous to the jerky effect seen in computer animated sequences where motion blurring is not

applied. By widening the time density so that it encompasses the entire inter-interval duration, the staccato nature of the sound can be minimised, producing smooth output at high time compression factors (Figure III.7(b)). This dynamic selection of overlap time can be used in situations where widely varying rates of a process must be sonified, preserving the audio "flow".

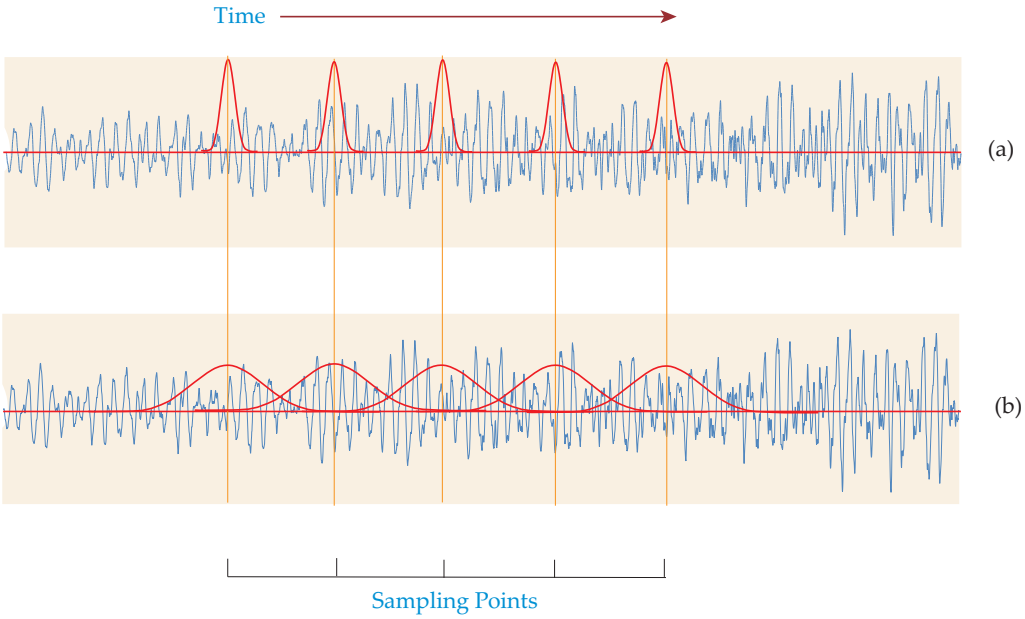


FIGURE III.7: (a) Audio densities are sparse and non-overlapping, leading to a jumpy sound. (b) Densities are adjusted so as to cover the entire inter-interval region. Sound is smooth and clean

III.5.7 EXAMPLE: ARC LENGTH MAPPING DISPLAY

A simple use of the time flexibility of the granular synthesis technique is to map time along curves in space, to produce a simple "completion" effect feedback (as in Müller-Tomfelde and Steiner [2001]) for spatial gestures. The implementation of such a system is described in this section.

In this implementation, a user can create a static path in a two-dimensional space from a number of points, through which a cubic spline is fitted. There is a Gaussian density around the path, and there is an audio source associated with the path, whose probability given is the density value at that point. The mean of the temporal density is mapped to the current arc length of the point nearest the user on the path. As the user moves away from the mean, this density expands, reflecting the uncertainty as to which part of the path the user is currently "at". The audio source in this case is a short musical excerpt; the music can be imagined to be written along the line of the path with the current time pointer sliding along it, reading the notes off (although the stretching and compression is smooth here, without note "jumping" effects). An image from the implemented system is illustrated in Figure III.8. See Appendix C for video and audio footage.

More formally, for the trajectory goal the waveform time samples

$$t \sim \mathcal{N}(\delta, \sigma_t), \tag{III.1}$$

where

$$\delta = \int_0^l \sqrt{\left(\left(\frac{dx}{dv}\right)^2 + \left(\frac{dy}{dv}\right)^2\right)} dv, \tag{III.2}$$

for a curve of length l and

$$\sigma_t = d\alpha, \tag{III.3}$$

where α is some constant defining the rate at which time blurs,

$$d = \sqrt{(n_x - c_x)^2 + (n_y - c_y)^2}, \tag{III.4}$$

c is the current cursor position and n the nearest point on the curve (subject to monotonicity constraint). The probability of selecting samples from the trajectory waveform is

$$p_1 = \frac{1}{2\pi\sigma_v} e^{\left(\frac{d^2}{-2\sigma_v^2}\right)}, \tag{III.5}$$

where σ_v is the trajectory width constant. The probability of selecting from the uniform background goal is just $p_0 = (1 - p_1)$.

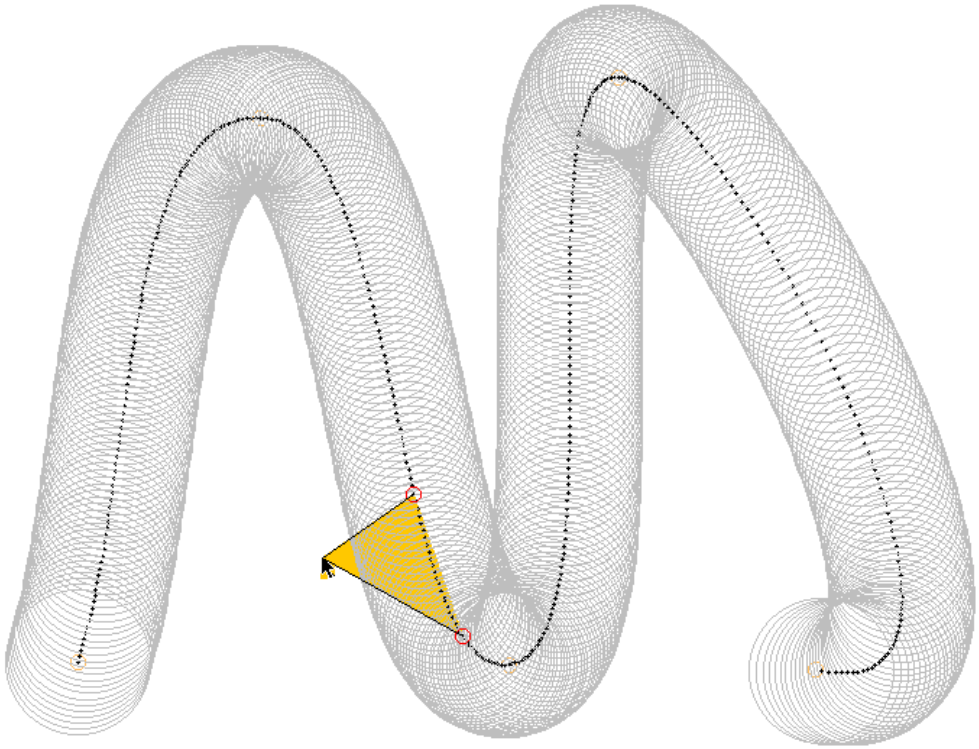


FIGURE III.8: An image from the arc length mapping demo. The dark path indicates the mean positions, while the outer circles indicate the one standard deviation bound. The current position along the path is given by the triangle extending from the cursor; as the mouse moves closer and further the potential positions contract and expand respectively. Audio is produced corresponding to the section of the curve highlighted by the two endpoints.

Extending this technique to more complex representations of phase is straightforward. Section IV.5 shows how the technique can be used with a particle filtering trajectory warping algorithm, displaying the ensemble of estimated phases in a gesture recognition situation.

III.6 CONCLUSIONS

Many interfaces can benefit from the adoption of appropriately ambiguous display; a user can only make rational decisions if the feedback perceived accurately reflects the beliefs of the system. Monte Carlo sampled approximations to uncertainty are a computationally convenient way of representing and manipulating uncertainty, and lend themselves well to display. Almost any interactive system can be augmented in this way. The particulate methods map easily onto the visual, auditory and even vibrotactile domains. [6]

[6] See Crossan et al. [2004] for haptic granular synthesis for probabilistic display.

III.6.1 Granular Synthesis As A General Method for Probabilistic Display

Granular synthesis is a powerful and flexible synthesis tool, and is a suitable technique for uncertain auditory displays, sonifying the time-varying states of a systems belief states. The nature of granulated approaches makes it easy to synthesise smooth audio textures with varying timbral characteristics and powerful control over the temporal flow of the audio. The process can be represented in continuous form, without relying on discrete alerts to inform the user. The asynchronous "clouds of sound" algorithm presented here can be adapted to many contexts where belief representation is relevant. Goal-directed feedback is produced by placing densities on the state space of a system (possibly combined via Bayes nets), transforming estimates of physically measured state to points in the goal space. Thus the feedback relates to the inferred goals of the user, rather than the state of interaction itself.

The following chapter extends these techniques, using Monte Carlo sampling to infer *future* potential states of the system and the user, propagating the uncertainty through time. The resulting distributions are then sonified via granular synthesis, and displayed with visual point clouds.

CHAPTER IV

PREDICTIVE UNCERTAIN DISPLAYS

Showing the possible futures.

> *Experience indicates that, by using a properly designed predictor instrument, a novice can in ten minutes or less learn to operate a complex and difficult control system as well as or better than even the most highly skilled operator using standard indicators.*
>
> – Kelley [1968] [discussing submarine control displays]

IV.1 SUMMARY

IN this chapter, the uncertain displays that were developed in the preceding chapter are extended to include predictive capability. These predictive displays can help mitigate the effects of delay in an interaction loop; the appropriately uncertain display ensures that the display remains "honest" and is not over-confident. The power of Monte Carlo predictive algorithms in interactions is discussed, and some examples of complex control problems are augmented with such predictors. Experimental evidence of the utility of uncertain display is presented. The natural extension from display of pure prediction to integration into the inferential process itself – using particle filters to estimate hidden states – is presented.

IV.2 QUICKENING: PREDICTIVE DISPLAYS

IV.2.1 THE NEED FOR PREDICTION

The function of a feedback mechanism to provide the interactor with sufficient information to control the system outcome so as to best match the interactor's intention. It must therefore display all variables necessary for effective control. To be most effective, these should be displayed *at the time when they are most useful*. In particular, the feedback should wherever possible take into account any delays in the closed-loop that would otherwise impair control. Humans have adapted to deal with the delays inherent in perception and motor control; this makes control in the physical world possible. Compensating for additional delays introduced by interactive systems is challenging, as was discussed in Section II.8.4. Where delays are unavoidable, predictive and preview displays can be used to mitigate the effects.

Displaying predictions is one way of reducing the cognitive load on the user. The system can take over some of the modelling that the user would otherwise have had to have

done to control the system. Since the system often has better access to the variables involved in an interaction task (particularly if there are very many dimensions), and much greater computational power with which to process them, these models can easily exceed the best performance of human predictions.

Delays are unavoidable in any system, and there are many such sources in human-computer interaction. Section II.8.4 outlined some of the important sources of delay and lag effects in interaction. Everything from hardware latency to muscle dynamics introduces delay into the control loop. Ignoring these delays and displaying only the "current" state pushes the work of prediction onto the human interactor.[1] This is far from optimal, both because the system should relieve the user of effort wherever possible, and because the system may well have better information as to the likely future outcomes. Systems can predict both their own responses and those of users interacting with them, though prediction of the subsequent states of a human interactor is only possible in the very short term – otherwise communication would be impossible. When delays reach the order of 250ms or more control generally degrades to an unacceptable extent (see Section II.8.4).

The purpose of predictive feedback is to help a user specify intention in a form the system can interpret reliably. In particular, it can be used to reveal how much effort will be required to drive the system towards the intended goal, and what classes of action will improve or degrade the beliefs.

IV.2.2 Existing Predictive Interfaces

Predictive controllers are widespread in industrial control (see e.g. Qin and Badgewell [1997]), ranging from sophisticated process models (Ziebolz and Paynter [1954] and Smith [1959] introduced simulation control) to the simple derivative component of the well-known proportional-integral-derivative (PID) controller. These controllers use models of future behaviour to influence the output applied to a system under control. In the presence of delays, this can significantly increase the quality of control. The same principle can be applied to manual control, where the human is acting as a controller. By extending the display to show (self) predictions of state, the user's level of control can be enhanced.

IV.2.2.1 Predictive Instruments for Manual Control

In manual control applications, predictive displays have been applied, especially in vehicle control and gunnery. In this field, such displays are often described as being "quickened". The most widespread examples are the rate displays in aircraft (e.g. the VSI indicates rise/sink rates). Kelley [1968] gives an excellent account of some of the various predictive displays suggested for submarine control and also has one of the first discussions of fast-time models for predictive manual control (see Figure IV.2); Hess and Gorder [1990] give a predictive display for helicopter pilots during hovering manoeuvres (illustrated in Figure IV.1); Birmingham and Taylor [1954] describe one of the first electronic quickened displays, and also coined the term "quickening". Almost all of these displays operate by augmenting the display with derivatives of a control variable. Kelley's quote, which opens the chapter, illustrates the power of predictive instrumentation.

[1] There is of course no "current" state from either the system or the user's perspective – evidence arrives at different times and is fused into a current estimate of the world in both the past and the future. The sensation of conscious "now" in humans may well be constructed *post hoc*, as experiments with the readiness potential (see Libet [1981] or Libet [2004]) indicate.

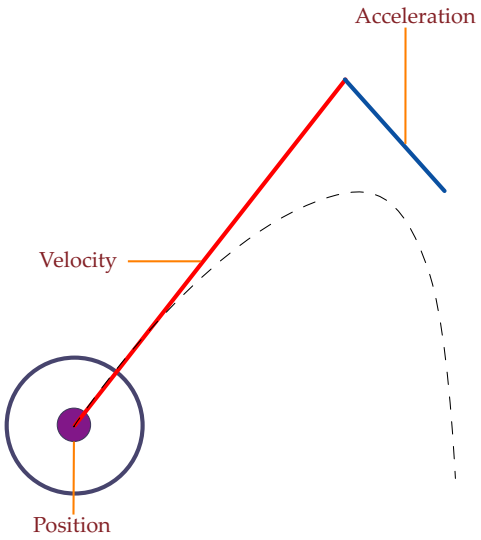


FIGURE IV.1: Quickened display for helicopter hover manoeuvres. Adapted from Hess and Gorder [1990]. The acceleration, velocity and current position of the aircraft are displayed on a map to aid in low speed movements. The dashed black line is not part of the display. It indicates the path an object would take given the velocity and acceleration vectors shown here (with 1 acceleration unit = .07 velocity units).

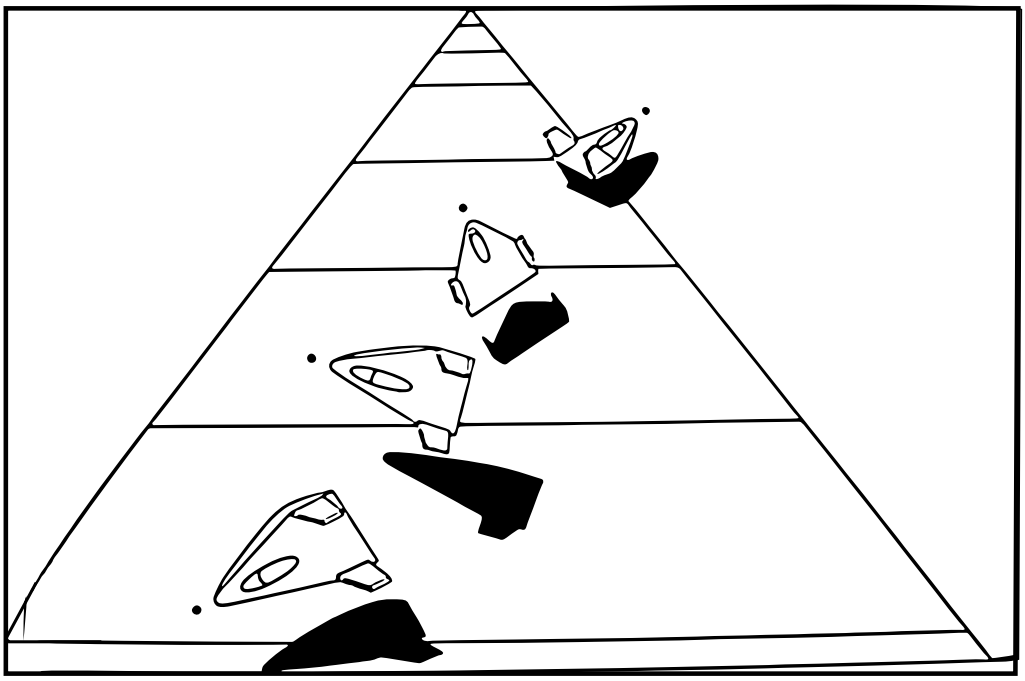


FIGURE IV.2: Kelley's postulated predictor model for spacecraft manoeuvres. This display is intended to be based on a fast-time model of the process; a simulation approach to predictive control, based upon Ziebolz's (Ziebolz and Paynter [1954]) work. Reproduced from Kelley [1968] p.147

IV.2.2.2 PREDICTION IN COMPUTER INTERFACES

Predictive displays have also been employed in virtual world navigation (e.g. Chapman and Ware [1992]) and especially in telerobotics (for example, the predictive robot arm interface described in Bejczy et al. [1990] or the video overlays for teleoperator control described in Noyes and Sheridan [1984] and the more sophisticated photorealistic predictive displays described in Barth et al. [2000] and Burkert et al. [2004]).

Predictive displays for mouse pointing tasks in desktop user interfaces based on velocity estimates are discussed in Asano et al. [2005], and in Murata [1995]. The system described in Asano et al. [2005], for example, attempts to ascertain the maximum of the velocity profile in a pointing movement, assuming a pointing model where trajcetories are straight lines with a smooth velocity profile, having a single well-defined maximum. It then uses this to predict the final goal destination, and displays the result with an animated "jumping" motion. The system was found to have some improvement in pointing performance for distant targets.

Glimpse (Forlines et al. [2005]) demonstrates a different style of predictive interface. In this model, intended for touchscreen based interfaces, gentle pressure causes the display to indicate what would happen at that point; pressing harder "pushes through" and the action is actually performed. This reduces the necessity of functions such as undo; users can perceive what the effect of their actions will be without irreversibly changing the state of the system.

IV.2.3 UNCERTAINTY

It is important to note that predictions are most beneficial when they include appropriate uncertainty. Without this, a display of a single future possibility with unrealistic confidence will lead to instability and inaccuracy; it is quite possible to build a predictive system which is much *worse* than a non-augmented system if uncertain display is not employed while the model is inaccurate. If estimates of uncertainty are available, then the display can reflect the full distribution of potential outcomes at the predicted horizon, becoming less informative as the relevance of information decreases.

Knowledge of the changing uncertainty of a situation is of importance when planning a set of actions to perform some goal; a certain but low-value outcome may be preferable to a risky but potentially high-value outcome in some cases. Dennett (Dennett [1984]), puts it thus:

> If an airline pilot is informed that there is a dangerous thunderstorm ahead, a storm so severe that were he to enter it he would risk **losing control** of his plane, he can divert the plane to another course in order to preserve or enlarge the margin of error he maintains [...] Recognising this, the pilot not only strives to control the plane at all times; he also engages in meta-level control planning and activity – taking steps to improve his position for controlling the plane by avoiding circumstances where, he can foresee, he will be forced (given his goals) to thread the needle between some Scylla and Charybdis. [emphasis in original]

Modelling of uncertainty should include the noise in the inputs to a system as well as uncertainty about the behaviour of that system.

IV.2.4 LINEAR PREDICTIONS IN GOAL SPACES

In the structure defined in Chapter II, there is a control loop in which the user controls the system's interpretation of user intention. The state of the system with respect to potential goals is the result of inference on some other control process. There are two basic ways of breaking down predictive models: models which attempt to predict new positions *directly in the goal space*, the focus of this section; and models which perform prediction in some portion of the state space and then project back into goal space. These are further discussed in Section IV.2.5.

It is possible to add some level of predictive display to any probabilistic control-based interface by introducing derivative augmentation to the goal space display. Under the assumption that a linear prediction is a reasonable estimate of future potential position in a goal space, the current state can be augmented, then sonified as described previously. Such an assumption is justified by smoothness properties of trajectories in goal space, as discussed in Section II.7.2, assuming that the system under consideration has rational evidence accumulation. A predicted value is obtained by taking:

$$p_i = \sum_{j=0}^{n} k_j \frac{dp_i}{dt^j} \tag{IV.1}$$

where the k's are weighting factors, and then normalising $p_1 \dots p_n$ such that $\sum_{i=1}^{n} p_i = 1$. In practice only a small number of derivatives can be used (estimating high order derivatives becomes difficult as noise rapidly dominates), in which case k_j will be zero except for the first few j.

Augmenting the simple targeting system described in Section III.5.4 with this approach produces an audio display with characteristics similar to that of the classical derivative-

enhanced predictive instruments. Approaching a target causes rapid increase in the intensity of the audio; overshooting causes very noticeable drops in intensity. In contrast to the classical instruments, however, more complex systems where the goals are not so directly linked to the sensor space can also be augmented. The augmentation is always with respect to the *inferred goals*, not to attributes of the interaction.

Although this technique is applicable to any suitable interface, it cannot provide accurate predictions for more than very short periods of time. A more sophisticated model would involve performing predictions in the system state-space and then projecting onto the goal space where these results are sonified or otherwise displayed.

IV.2.5 Sonification of Monte Carlo Propagated Uncertainty

The optimal feedback is the complete set of possible states the system could be in after some time t has elapsed and their probability. This will involve propagating the density around the current state forward in time, applying the (estimated) dynamics of the control loop.[2] These original densities should include all possible world states compatible with the observations. The optimal predictions should integrate over all possible system dynamics and user responses. In practice, these predictions are infeasibly complex and approximations have to be made to produce a tractable model. One powerful way of producing predictive displays is *prediction by simulation*; simulating the dynamics of a system and running the model faster than real-time. These simulations can be run under varying models and initial states, to obtain the performance of the system under different assumptions.

The first approximation considered is to represent densities by a large number of individual samples drawn from those densities, i.e. a Monte Carlo approximation. This is a flexible and easy to implement way of representing the densities in such a way that they can be transformed by any (computable) function. The accuracy of the representation is limited by the particle counts, and can be adapted when varying levels of computational power are available. Such approximations are also potentially parellizable.

The second is to constrain the potential user responses to a limited subset of the potential universe of responses. For this purpose, an operator model of the user must be obtained. There is a delicate balance between choosing a model which responds like the user (i.e. is complex enough), and a model which the *user can model* – the feedback will be useless if the user cannot determine what estimated behaviour was involved. For the models described below, the most useful approximation is that user input will be constant during the prediction time. Other approximations (see for example, Poulton [1974]), assume user input will return to some neutral state (centreing on a spring-loaded joystick is a common example), or attempt to integrate over larger possible actions with more complex operator models. However, the assumption of constancy provides the user with a way to explore the effects of changing the controls at the future time horizon in an intuitive manner. This provides benefits when the interactor engages in "dithering" to explore the local dynamics of the system; Kelley [1968] defines dithering as follows:

> Testing his internal model is often an important part of adaptation through inferred changes in the control situation. The pilot sometimes "dithers" the stick of an aircraft to see if the plane is responding in the way he expects.

[2] Obviously at least some of the control loop is unknown – otherwise communication would not be possible – but any short time period is likely to have states occupying a very small proportion of the possible states. The bandwidth restriction of the interface in fact implies that this should be true, since the rate at which the state can diverge is bounded.

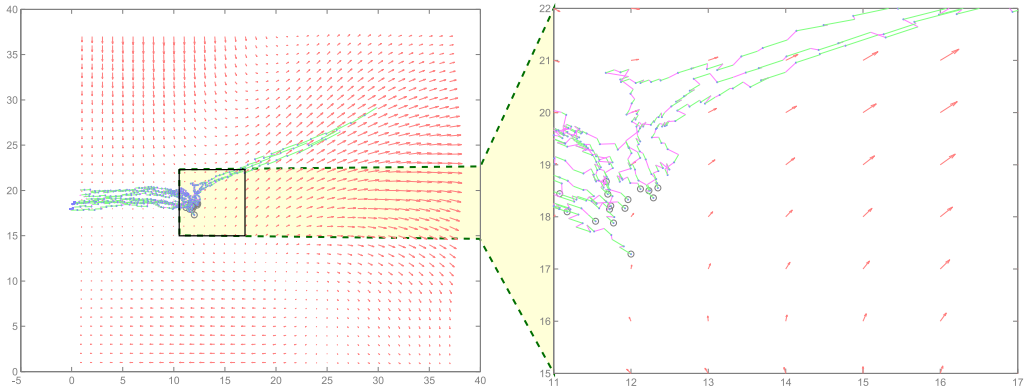


FIGURE IV.3: Monte Carlo propagation example. Shown are ten particles being propagated through a dynamic system (vector field shown in blue), starting at the points marked with black circles. At each step the deterministic dynamics are applied, followed by the noise diffusion process. The right hand side shows a close up of the process. Particle positions at each step are marked in red. The dynamics step is shown as a green line, the noise step as a magenta line. The initial condition is centred on a saddle point, leading to a strongly bimodal terminal distribution.

The third step is to replace the continuous dynamics with a discretised version which is amenable to recursive Monte Carlo updates.

Given a model of the dynamics of the interaction, a prediction model can be produced by sampling around the current state, and propagating Monte Carlo samples forward through the joint system-user dynamics to produce an estimate of the potential states at some time horizon. These estimates can be transformed to goal space via the goal densities and then sonified or otherwise displayed.

It should be noted that although this work concentrates on the Monte Carlo sampling approach, in some cases analytic approximations to the predicted density can be obtained without requiring sampling; see for example the Gaussian Process approximations used for time series prediction in Girard [2004]. These may offer more accurate estimates in some cases, but are reliable only under more restricted assumptions. For systems which are approximately linear there are classical techniques for propagating simple densities (Gaussians or mixtures of Gaussians) – unfortunately human-system interaction is very rarely linear over anything other than very short timescales.

An example of the Monte Carlo propagation procedure is illustrated in Figure IV.3, showing particles being propagated in a simple first-order nonlinear system (defined by a smooth vector field). The general algorithm is outlined below.

GENERAL MONTE CARLO PROPAGATION ALGORITHM

To approximate the effect of the system dynamics $h(x,t)$ on the density f over the time period $0 \ldots T$:

- For n samples:
  - Given a p.d.f. $f(\mathbf{x})$ around the current state $\mathbf{x}$, draw a new sample $\mathbf{x_0^i}$.
  - For each time step t until $t = T$:
    - $\mathbf{x_t^i} = h(\mathbf{x_t^i}, t)$ where $h(x,t)$ is the model dynamics.
    - Draw a new sample from the p.d.f. $\mathbf{x_{t+1}^i} = f_{x_t^i}(\mathbf{x_t^i})$.

The samples $x_T^i, \quad i \in \{0 \ldots n\}$ are the output samples at the time horizon. Given the goal densities on the state-space $g_j(\mathbf{x})$, the probabilities p_j are calculated and displayed.

IV.2.5.1 EFFECTS OF PROPAGATION

By correctly propagating the uncertainty involved in the prediction, over-confident results can be avoided. Section III.3 gives details for the non-predictive case; these generalise to the predictive case. In the predictive case, however, uncertainty rapidly increases as projections extend, and the need for uncertain display becomes greater.

This information allows the user to adopt a suitably stable control behaviour, as Section III.3.1 discussed. The uncertainty also implies that less information needs to be communicated to the interactor. The display can thus become less intense, decreasing the level of attention the user requires. This may seem counterintuitive (because often far more display "work" is required for a widely dispersed view), but diffusion of both visual particles and audio granules leads to a pleasing blurring effect.

IV.2.5.2 INTERACTIVE PROCESS EXPLORATION VIA TIME HORIZON MODULATION

In the preceding sections, a fixed prediction horizon has been assumed; however the prediction system can be enhanced by giving the interactor direct control over this time horizon. This facilitates interactive scanning of the future set of possible actions to provide the most pertinent information about the current interaction.[3]

Users may probe further to gain long-term understanding, or move back to short time horizons to engage in tight control. This allows the user to make a reasonable judgement about the trade-off between using a tightly-coupled closed-loop strategy and open-loop ballistic behaviour. It is also possible to display multiple time horizons with appropriate display techniques: in a visual (Monte Carlo) setting multiple particle "layers" could be displayed, or even the entire search particle "beam" (see Figure IV.6 for an example); in audio automatic, rapid, radar-style sweeps through horizons could be employed, at least for systems with reasonably slow dynamics.

Adjusting the time horizon in a Monte Carlo prediction process is rather like adjusting the angle of fire on an artillery piece; higher angles – up to a point – reach further but are more widely distributed (see Figure IV.4). Similarly, the angle of headlights on a car can be adjusted to provide either close range, focused lighting, or more distant but dim and diffuse. In implementations using this technique (see Section IV.3.1), the horizon

[3] It is interesting to note that it might be possible to link the the complexity of the terminal distribution to the time horizon and devise an automatic optimal horizon – "semantic autofocusing".

control is implemented as a head-tracking device which uses variation in head pitch to control the prediction distance, as an analogy to the artillery/headlights scenario.

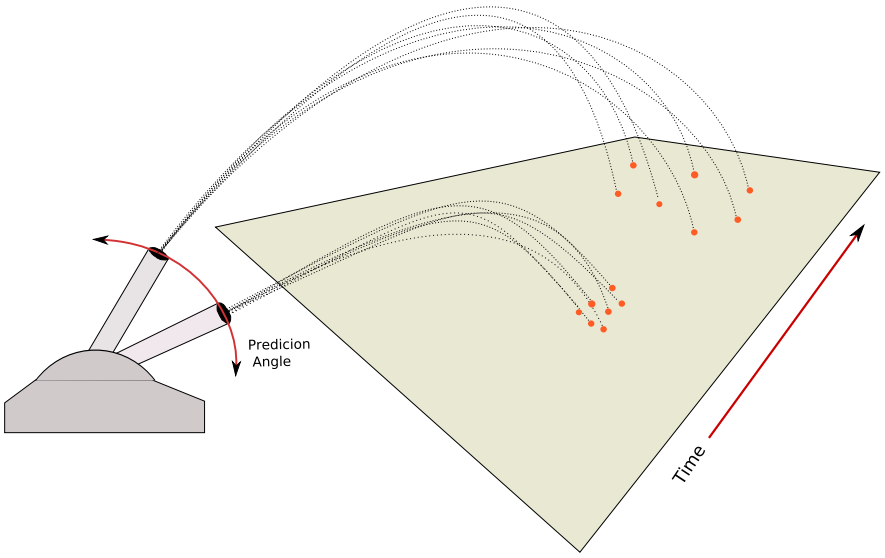


FIGURE IV.4: Artillery fire as an an analogy to adjustable time horizon Monte Carlo prediction horizons. Higher angles (up to 45^o) have greater range (in time) but increased diffusion.

IV.3 EXAMPLES

This section shows how predictive uncertain display can be used in practice. Dynamic systems which share some of the properties of interactive systems, but which are simple enough to demonstrate the ideas without undue complexity, are augmented with Monte Carlo predictive displays. The real-world problem of GPS navigation (with its variable quality of fix) is also tackled using uncertain predictive display. The use of the granular approach with particle filtering is also demonstrated.

IV.3.1 EXAMPLE: BEARING SIMULATION

As an example of the application of the predictive feedback to continuous control systems, the techniques described above have been applied to a simple physical system to demonstrate the applicability of the ideas. This shows how simulation models with propagated uncertainty can be created, displayed and sonified.

IV.3.1.1 MODEL The model chosen for the demonstration is a simulation of a small ball-bearing rolling over an undulating surface, with simulated gravity (see Figure IV.5). The orientation of the surface in this implementation is mouse controlled, leading to a second order control of the ball position (angle control leads to changes in the forces applied to the ball, i.e. changes in acceleration). The deformations in the landscape introduce non-linearities into the system dynamics. This model is a simplified version of the process used for prediction of navigation behaviour in the GPS navigation system of Section IV.4.

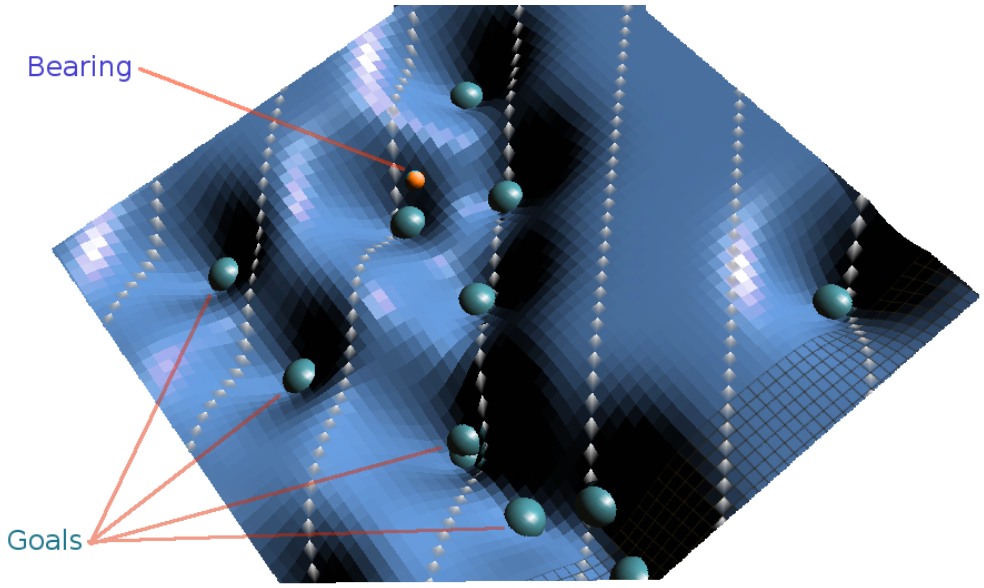


FIGURE IV.5: Simulated surface with goals distributed across it. The dynamics of the system are defined by the landscape. In this case, each goal has a smooth depression around it (leading to an attractor at that point). The solid blue spheres mark the centres of the goal distributions. Control is effected by virtually "tilting" the surface (i.e. by adjusting the effective gravitational vector).

The position of the ball is assumed to be uncertain. Prediction is performed by sampling from points around the current ball position, and projecting forward through the model dynamics. The surface also has uncertainty (representing *model* uncertainty); this is variable across the surface.

With this algorithm, the model uncertainty leads to trajectories which are unrealistically noisy (i.e. excessive high-frequency content) because subsequent samples from the surface distribution are assumed to independent; a better prediction algorithm would sample a smooth example from a class of surfaces and use it for the entire trajectory of one sample. A Gaussian process, for example, would be ideal for this task.

In this simulation, "goals" are associated with the attractor basins on the simulated surface; each goal is a Gaussian density with mean centred on the attractor. The activation of these goals is what is sonified. The possible trajectories of the ball are displayed visually. The visual display relates predictions of the evolution of the system with respect to

the simulated world; the audio display relates predictions with respect to the potential goals of a user.

IV.3.1.2 TECHNICAL ISSUES The surface here is a discretely sampled grid, with linear interpolation between points. The surface is formed from the sum of a number of (inverted) Gaussians. The system dynamics are integrated with the forward Euler method, one step per frame, and with a frame rate of approximately 40Hz.[4]

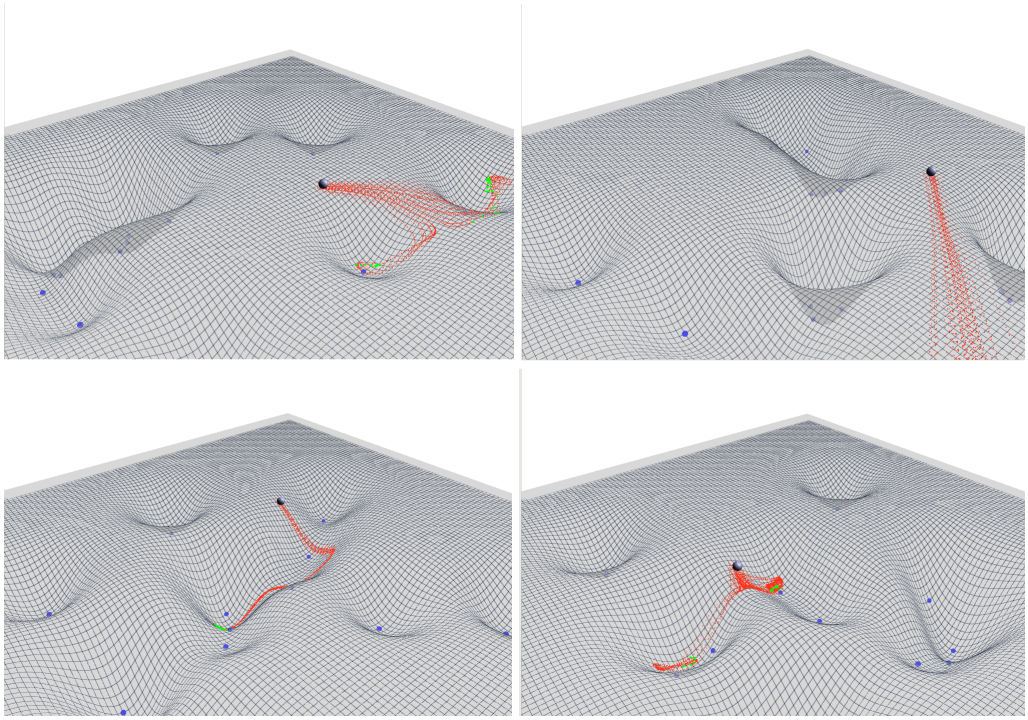


FIGURE IV.6: The ball-bearing simulation model. The ball-bearing can freely move over the surface. The mouse alters the orientation of the surface, and thus the effect of gravity on the motion of the bearing. Predictions are shown in red. The predictions at the time horizon are highlighted in green. There are a number of waveforms distributed on the surface; the centres of the density associated with them are shown as blue solid circles (at the centre of each depression).

[4]The visualisation is written in C, using OpenGL for visual display and the GSLib (see Chapter A) library for audio output.

FIGURE IV.7: InterTrax head tracker used for the manipulation of the time horizon in the rolling object demo. Tilting up and down extends and retracts the prediction horizon.

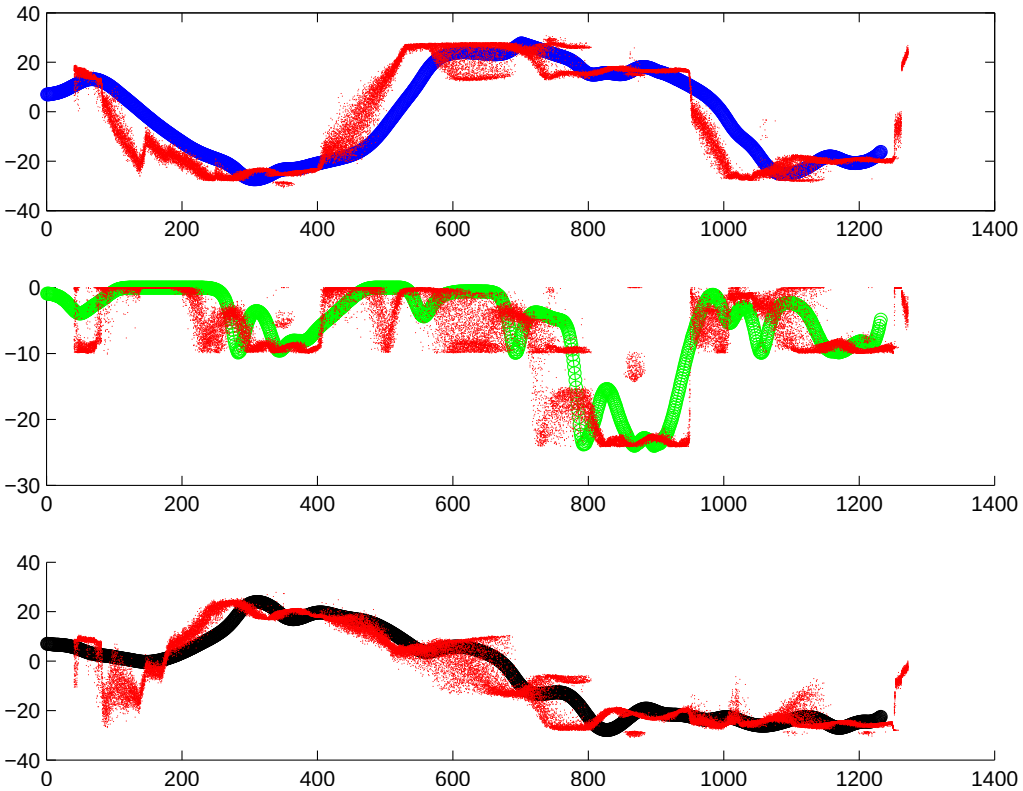


FIGURE IV.8: Predicted and actual states in the rolling ball simulation. Actual and predicted positions for the bearing are shown. This shows prediction at approximately two seconds ahead (frame rate is slightly variable but is around 40Hz. 80 step ahead prediction is used.)

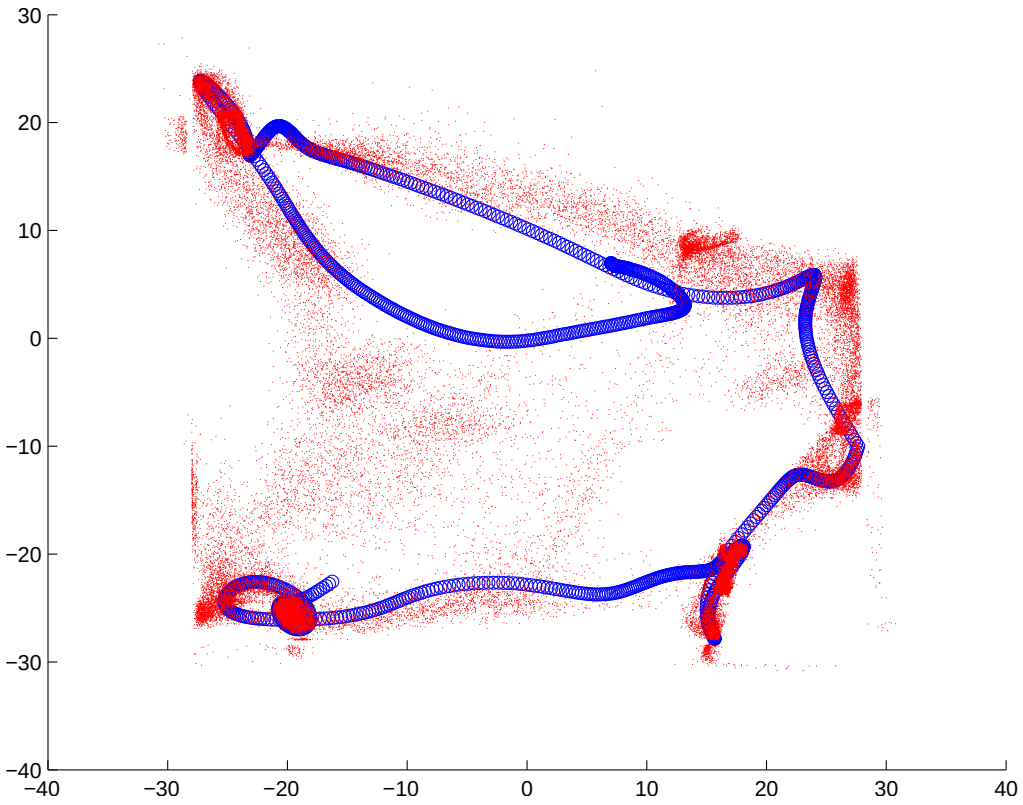


FIGURE IV.9: Predicted and actual states in the rolling ball simulation (2d projection of the trajectory).

IV.3.2 EXAMPLE: HELICOPTER DISPLAY AUGMENTATION

The following sections describe how the goal-directed audio feedback can be applied to a helicopter control task. This example is intended to illustrate how the techniques can be applied to a well-known control problem, which has many of the same characteristics as interaction tasks in difficult conditions. The presence of high-order control, delays and unpredictable behaviour make piloting helicopters a challenging task; many of these same issues are present (in less extreme forms) in interaction problems. Helicopter control, for example, is not so dissimilar to operating a brain-computer interface, in some respects; there are long delays (aerodynamic responses/ long classification windows), strong external disturbances (wind buffeting/spiking neurons) and unfamiliar actuation (four limb co-ordination/purely mental control).

IV.3.2.1 THE HELICOPTER PROBLEM

Control of helicopters has been extensively studied as an example of a difficult control problem. The difficulties of flying rotary-wing aircraft arise from several causes: the pilot must simultaneously coordinate all four limbs in a fairly complex interdependent way; the pilot must continuously monitor an array of instruments; the controls are high-order (e.g. cyclic pitch is a fourth-order position control); the helicopter is highly dynamically unstable, rather like a spinning top balanced on its point

– even slight disturbances tend to lead to rapid loss of stability; and there is a very significant lag – often many seconds – between input and reaction. The complexity of the task requires that pilots undergo significant training to learn the controls of the system, even where artificial stability systems are available. The concentration required makes flying extremely tiring for pilots.

Helicopter flight is therefore ripe for enhancement with predictive displays. Some experimental predictor displays have already been developed for aircraft (see Section IV.2.2.1 for one example). Some classical instruments, such as the vertical speed indicator (VSI), are actually basic predictive instruments, displaying the *derivative* of a control variable. An instrument applying the techniques that have been presented in this work should display the future state of the system with respect to *the goals the pilot is working to achieve*.

IV.3.2.2 Goal Feedback

In a flight task, there are several regions of the state space which are desirable. These include hover, level forward flight, acceleration and deceleration, low speed manoeuvres (for example pirouettes), banking turns, ascents and descents. These regions can be further subdivided into a hierarchy of goals; for example hover comprises zero linear velocity, zero angular rates and level pitch and roll. The goal feedback systems described in the preceding sections can be adapted to the flight task by defining goal densities over the aircraft state space, corresponding to these desirable regions. These regions can be combined to form joint variables representing higher level states.

Prediction can be initiated from the current state and projected forward to some time horizon; given an uncertainty model, the full distribution at this point can be computed (here the Monte Carlo sampling described previously is used to approximate this). The likelihood of each goal at this horizon can be computed by summing over the particles at the horizon, weighted by the goal p.d.f. ($p(g) = \frac{1}{n}\sum_{i=0}^{n} p(x_n)$).

IV.3.2.3 State Space

The aircraft state-space normally includes position[5] (x, y, z), orientation (ϕ, θ, ψ) and the derivatives of these. Taking just the first derivative, a simple linear state space can be represented as $[\dot{x}\ \dot{y}\ \dot{z}\ \phi\ \theta\ \psi\ p\ q\ r]^T$ (p, q, r are the angular velocities). Here the position and yaw are omitted as they have no direct effect[6] on the evolution of the system. The local dynamics of the aircraft can then be approximated as a linear system

$$\dot{x} = Ax + Bu, \tag{IV.2}$$

where A is the system matrix and B is the input matrix. u is the vector $[\theta_0\ \theta_{ls}\ \theta_{lc}\ \theta 0t]^T$ representing the control inputs to the aircraft; lateral cyclic, longitudinal cyclic, collective and anti-torque respectively.

IV.3.2.4 Prediction

The helicopter state can be predicted by computing a local linearization around the current state, and then iterating through a series of time-steps solving (IV.3.2.3). A value for u is required for each time step; this can potentially be a model of potential pilot inputs but simpler approximations are return-to-zero (this is the method that Poulton [1974] suggests) or constant inputs. The linear approximation is only reasonable for a short prediction period of time since true helicopter dynamics are far from linear.

[5] The position of the centre of gravity.
[6] Wind, ground effect, etc. are not considered.

IV.3.2.5 SIMULATION Laminar System's X-Plane [7] was used as a platform for the implementation of a predictive audio display for helicopter flight. This has a fairly realistic rotary-wing flight model and features external connectivity via UDP output. This communication layer was enhanced with a C library created by Andrew Ramsay (Ramsay [2004]). This setup allows real-time simulation with easy access to all of the important flight variables. The output from this was integrated into a modified version of granular sonification system created for the bearing example in Section IV.3.1. The goal densities were Gaussians placed in the state-space (the implemented example used hover and forward flight as goals, as well as sub-goals for each of these) with arbitrary associated sounds.

Linearization is achieved by rapidly perturbing the simulator state, one variable at a time, every 200ms[8]. From this the Jacobian (A) is estimated via central differences. The uncertainty in state is assumed to be Gaussian (with pre-set variance for each state variable) around the initial state. The uncertainty in the model is likewise assumed to be Gaussian (on the individual components of A), and each sample trajectory receives a constant sample from the model distribution. The particles are propagated forward via equation IV.3.2.3. No further perturbation is performed beyond the initial step.

Some sample trajectories for one helicopter state (in a simulation Bell 206) are shown in Figure IV.10. The time horizon is controllable adjustable via keyboard adjustments. Time horizons can range $\sim$50ms to approximately 5s. Beyond this the predictions are likely to be so divergent that they are not useful; computational cost also becomes an issue.

[7] http://www.x-plane.com

[8] It should be noted that this is not an especially accurate estimation of the linear parameters but the high frame rate of the simulator makes it a reasonable approximation.

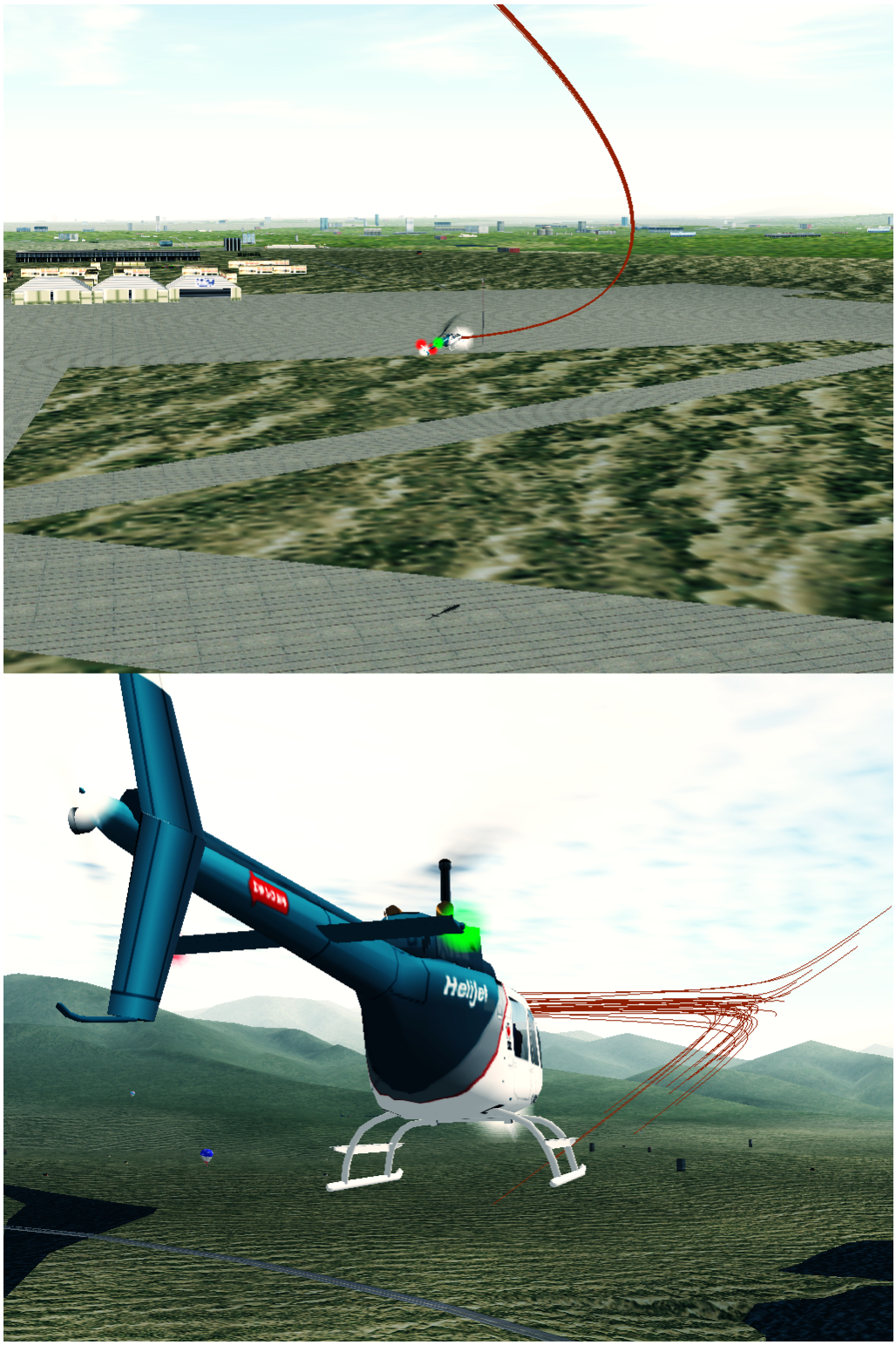


FIGURE IV.10: Visualization of the Monte Carlo sampled predictions in the X-Plane simulator. The "strands" extending from the helicopter body represent future possible paths for the aircraft.

Goal densities are defined on the aircraft state space, reflecting forward flight and hover behaviour, each of which are sub-divided into further sub-goals (e.g. zero yaw rate). Each goal and sub goal is associated with a target sound. Trajectories which terminate in these regions trigger the output of grains, exactly as in the previous bearing example. This produces a mixture of sounds which gradually form a chorus as the components of the desired behaviour are acquired. Figure IV.11 illustrates this decomposition. Additional flight modes can be added simply by defining new densities or introducing new joint variables (possibly extending the state space as necessary).

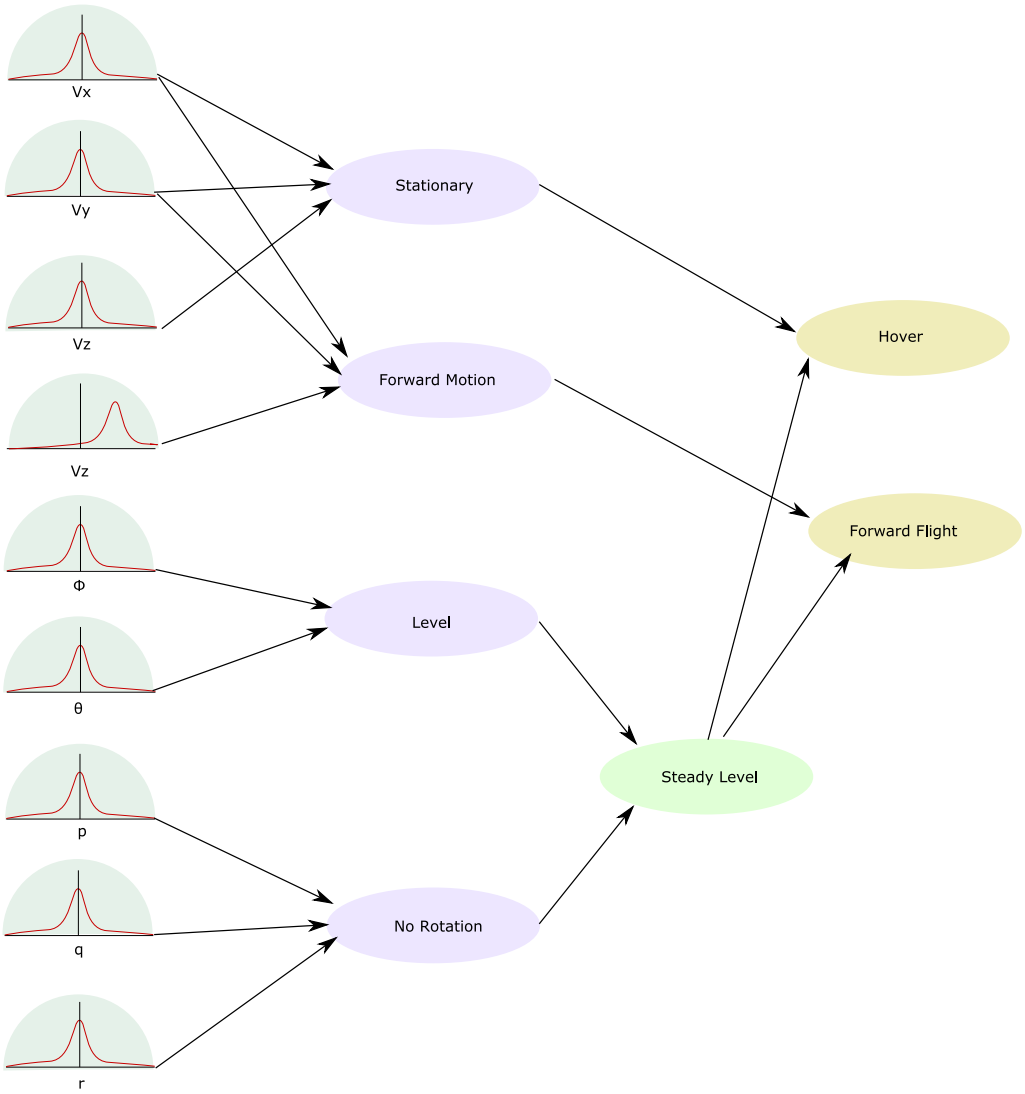


FIGURE IV.11: Dependencies and densities in the helicopter goal model. Arrows indicate dependencies. Densities are represented as filled semicircles. The oval objects are goals which are sonified. This graph is simple to evaluate as dependencies flow from left to right, and the left variables are known (they are samples from the Monte Carlo process).

IV.3.2.6 ANALYSIS Experimentally evaluating such a display is impractical; it takes a great deal of practice (at least thirty or forty hours) to learn to control a helicopter simulation to an extent that the display could possibly be useful. The display does, however, appear to show useful information, with the gradual build-up of goals forming a distinct pattern as the helicopter transitions into manoeuvres. Deviation from these patterns is quickly apparent. However, the prediction as it stands is relatively poor, and diverges quickly from the actual path of the helicopter, largely because of the lack of an operator model. The display as it stands only sonifies a small subset of the possible goals a pilot might consider during a helicopter flight. Extending to further goals (and their corresponding sub-goals) is a simple matter.

This example shows how goal-directed sonification can be brought into dynamic systems where uncertainty arises largely because of the inability of the user to model system behaviour over long delays. The idea can easily be applied to other domains where such unpredictable dynamics (from the point of view of the user) exist.

IV.4 A DETAILED EXAMPLE: GPS NAVIGATION

IV.4.1 THE GPS DISPLAY PROBLEM

The[9] global positioning system (GPS) is now almost ubiquitous in navigation systems. It does, however, suffer from accuracy problems, especially for activities like walking or cycling where small distances are important. Lack of satellite visibility in more extreme latitudes, slow update rates, and signal reflections and shadowing from buildings and other obstructions limit the utility of GPS for tasks such as navigating in cities. The navigation signal is subject to random jitter that can jump wildly (from a pedestrians point of view) even while the user is standing still; Figure IV.12 shows a typical GPS signal obtained while at a standstill.

[9]This work was conducted in conjunction with Steven Strachan at the Hamilton Institute, who ported the prediction code to the PocketPC and performed the experiments described here. This is an abbreviated version of the work presented in Williamson et al. [2006].

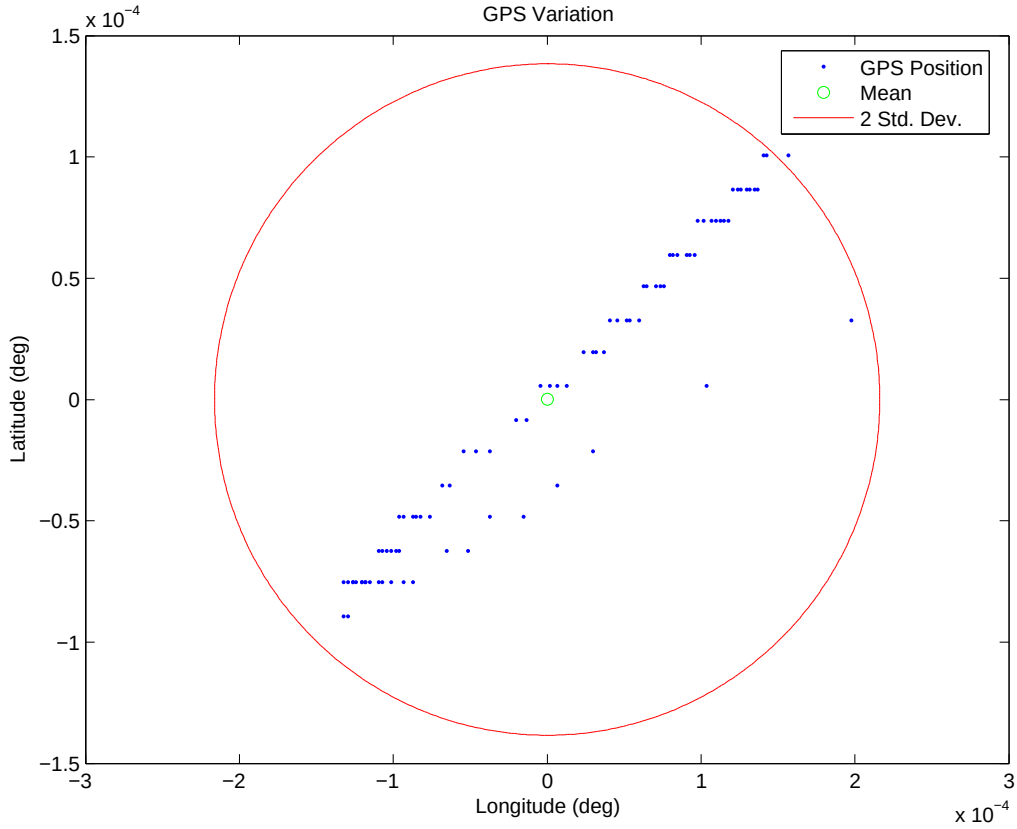


FIGURE IV.12: GPS position variation while the unit is at standstill in a two minute period. Units are degrees (the area shown corresponds to approximately 27 x 39 meters). This data is recorded in a clear area away from obstructions, at around 55 degrees north with the MESH GPS device (using a standard GPS chipset) and an external antenna. Both jitter and long-term drift are apparent. Noise would be even more pronounced in heavily built up areas.

One way of mitigating this problem is to give the user a more accurate display of what the GPS is truly sensing. The uncertainty on position can be displayed to the user, based upon a model of how GPS signals tend to vary. This can be combined with prediction to estimate where a user is likely to end up given the current information about their position and heading. A simple dynamic system can be substituted for the user, and fast-time simulations from this system can be used for display of possible future states (i.e. likely potential positions). If specific spatial goals, (e.g. the location of buildings of interest) are present, these can be displayed or sonified as predictions enter those regions of space, by placing goal densities on the geographical space.

IV.4.2 PROBABILISTIC PREDICTION DISPLAY: LIKELIHOOD MAPS AND SHADOW MAPPING

The methods for particle-oriented probabilistic display described in the previous sections can be applied directly to the map navigation problem. In this problem, there is a

significant amount of *a priori* information that can be used in the estimation of position, and the estimation of likely future movements of the user.

A given location has a particular signal uncertainty associated with it; this depends on the visibility of satellites, and other interference such as reflections from nearby buildings. Each location also has a likelihood of the user being that position. It is, for example, very unlikely that a user will pass through a known solid object. How likely such a position is depends on the nature of the obstacle and how sure the system is of the existence of such an obstacle. For a particulate display, the map of the certainty of the signal can be used to define the initial distribution of particles, representing positions compatible with sensor readings. The map of likelihood of observation can be used to define how the user is likely to move into a particular area.

Maps can be created based upon particular contexts of use; vehicle navigation is much more constrained than foot navigation, for example (see Figure IV.16(d)). Where context can be sensed (for example detecting cycling behaviour with inertial sensors), the maps can be updated to reflect the likely possible movements. Given a distribution over context variables, a total likelihood map can be formed as a linear combination of likelihood maps, weighted by the probabilities associated with the contexts the maps are designed for.

An estimate of the availability (and therefore quality) of a GPS signal can be obtained by *shadow mapping* (see Steed [2004]) – computing the visibility of satellites at each point in a map, based on data about the structure of local occlusions. Figure IV.13 shows an estimated shadow map for a region of the Maynooth campus in Ireland, computed by ray-tracing based upon a height map of the buildings. Satellite availability is sometimes also available directly from some GPS devices; this can be used to update the uncertainty.

The same height map can be used as a simple likelihood map (users tend not to pass through regions of very high gradient). This is the approach used in the examples below. Other more complex likelihood maps could be constructed, including obstructions such as busy roads and anisotropic likelihoods (such as in one-way streets).

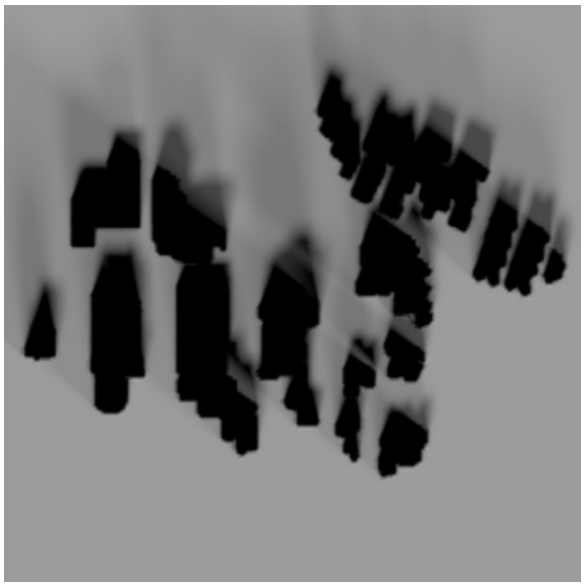


FIGURE IV.13: A shadow map computed for a region of the Maynooth campus in Ireland. This is estimated via a raytracing technique given the height of the surrounding area, and the angle of GPS satellites.

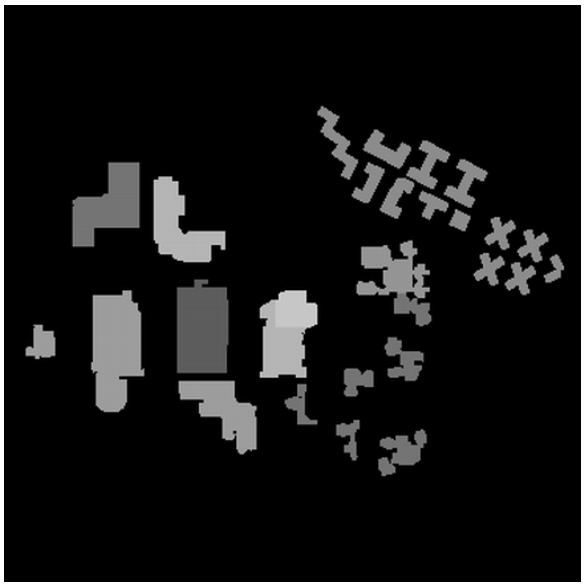


FIGURE IV.14: A likelihood map computed for a region of the Maynooth campus in Ireland. This is based on building positions. Entrance into buildings is not modelled in this example.

IV.4.3 A NAVIGATION APPLICATION

Figure IV.16 shows a GPS navigation interface with true uncertain display in operation. Samples are drawn from a position distribution, given a measured location and the estimated uncertainty in the signal (from the shadow map, and also from the "horizontal dilution of precision", the signal GPS devices use to indicate their estimated uncertainty). This is propagated forward through the dynamic system which acts as a proxy for exploration behaviour, causing the particles to flow through likely areas of the space.[10]

The update operates as follows:

- Draw samples $x^0 \ldots x^S$ from a distribution ε around the current state. This distribution represents the sensor uncertainty at the initial position (e.g. from the shadow maps described).
- For each step t until some horizon T:
  - $x_t^s = x_{t-1}^s + h + l(x_t^s) + \sigma(x_t^s)$ where $\sigma(x_t^s)$ represents the model noise at the new point x_t^s (a fixed Gaussian, in implementations), and $l(x_t^s)$ represents the derivative of the likelihood map at that point. h is heading specified by the user. $\sigma(x_t^s)$ can be a constant value (as in the implementations) or a more complex function; e.g. from a map indicating the resolution or quality of the likelihood map.
- Display the samples x_T^s

IV.4.3.1 IMPLEMENTATION

A navigation system implementing this algorithm has been built for the PocketPC (Figure IV.15), with the MESH(Hughes et al. [2004]) device used for inertial and GPS sensing, as well as for vibrotactile display. Heading is obtained from magnetometers and accelerometers.[11] Ten particles are used for prediction, with time horizon scanning via vertical tilt angle. A version with simulated GPS and heading signals (controlled via keyboard and mouse interactions) on desktop machines has also been constructed; this is used to produce the figures in Figure IV.16.

[10] The system is similar to the landscape surfaces of Section IV.3.1. More sophisticated substitute systems (and corresponding maps) could be built, but this suffices as a simple model of navigation behaviour.

[11] Accelerometers are required for tilt compensation for the magnetic field readings.

FIGURE IV.15: PocketPC augmented with GPS sensor pack. The sensor pack include GPS, 3-axis accelerometers, 3-axis gyroscopes, 3-axis magnetometers, capacitive touch sensing and vibrotactile display

Targets are sonified via goal-directed granular synthesis. Each target has an associated p.d.f. (Gaussian with fixed variance), defining a transformation from physical space into goal space. Grains are generated according to the probability of each goal given the particle positions at the time horizon and these p.d.f.'s. Each goal has an associated waveform. Vibrotactile feedback is also presented, with a short (250Hz sinusoidal) impulse generated along with each audio grain. Impulses are identical for all targets; the vibrotactile feedback merely indicates the existence of a target, not its identity.

In this implementation, targets are laid out in physical space. Users can explore the space in an obvious manner, scanning for targets by swinging the device around. The display incorporates appropriate uncertainty, and includes a model of how exploration is likely to proceed. This display is given in terms of the goals (physical targets) that the user may be interested in.

IV.4.4 EXPERIMENTS AND ANALYSIS

Field trials were performed to analyse the performance of the uncertain GPS navigation system, and in particular to determine whether uncertain display lead to more robust behaviour than a simple direct display of noisy sensor readings. These trials involved participants navigating the area around the Maynooth campus in Ireland. The participants attempted to locate four targets distributed in this space, using only the audio and vibrotactile feedback from the navigation application.

IV.4.4.1 EXPERIMENTAL LIMITATIONS AND METHODOLOGY

As the GPS signal in the test area was strong, with few occlusions or sources of reflection, the effects of GPS distortions that would be experienced in built-up areas were artificially simulated. Noise compatible with the effect of degraded GPS readings was combined with the true sensed position before the feedback as to target proximity and heading was presented to the user. This noise consisted of small offsets (Gaussian distributed with a standard deviation of a few meters) adjusted

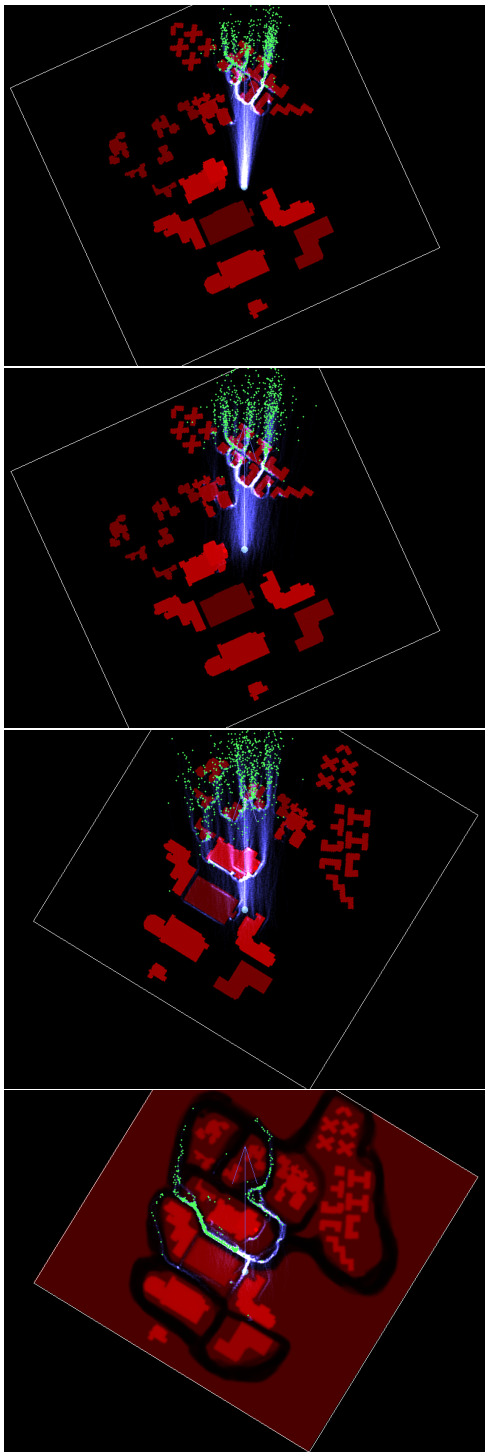


FIGURE IV.16: GPS Monte Carlo navigation application. This image is from a desktop mockup with simulated GPS input. The GPS enabled version runs on the PocketPC but does not have such visualisations. The first shows predictions with a likelihood map designed for walking behaviour. The second figure shows the effect of changing to a likelihood map suitable for bike navigation, where movement is more tightly constrained to pathways. The lower two figures illustrate the effect the shadow map has on prediction. The first of these two is in a relatively non-occluded area, and predictions begin, and remain concentrated. The second has poor satellite visibility; predictions are initially diffuse, but coalesce as obstacles reduce the likely movement paths. They subsequently re-diverge in open space.

every five seconds, producing regular disturbance in the apparent sensed position. Figure IV.17 shows the effect of the introduced noise, and the layout of the targets in the test area. The time horizon was fixed at a range of approximately 40m in the experimental sessions to reduce the complexity of the system.

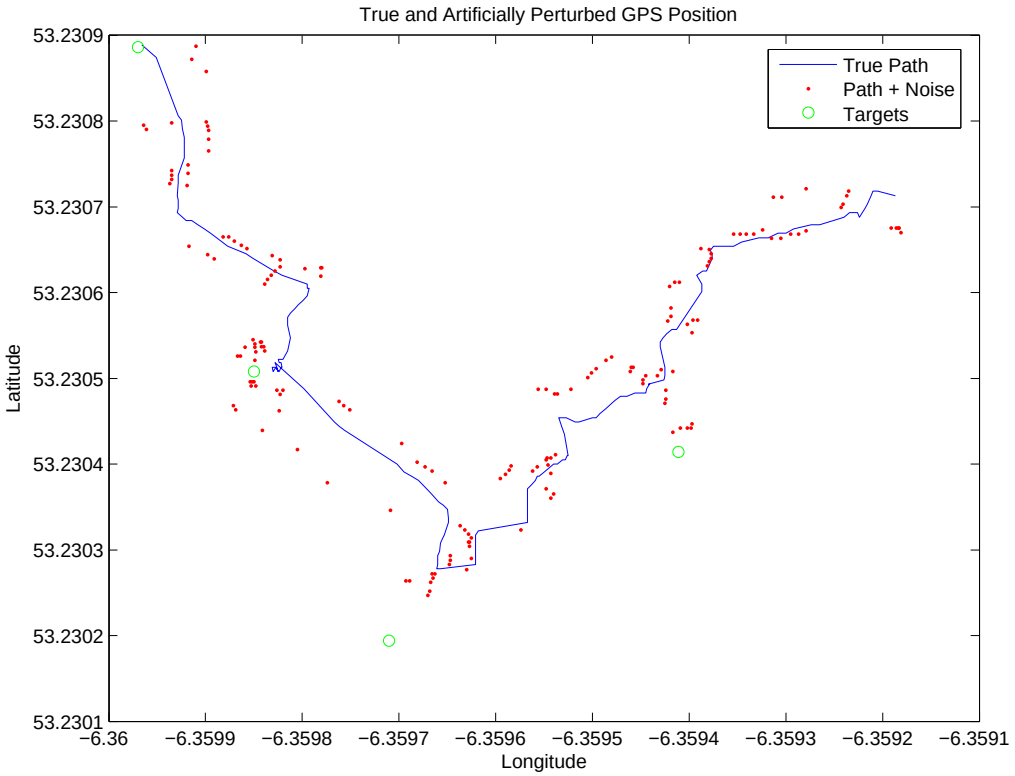


FIGURE IV.17: True GPS position and artificially disturbed GPS position during the navigation experiment. The true position is shown in blue; the readings after the introduction of noise are shown as red points. Hollow green circles indicate the targets the participant had to locate.

IV.4.4.2 CONDITIONS The experiment had two conditions: mean display and uncertain display. Mean display assumed that the GPS signal was certain, and feedback was generated directly upon the sensed+noise signal. The uncertain display condition represented and propagated the estimated uncertainty in the GPS signal according to the procedure laid out above.

The experimental design was within-subjects with counterbalanced presentation order. Participants received a short introduction to the system, along with a demonstration of the behaviour of the application. Participants had to navigate to the targets in order; once they were sufficiently close to the target the current target advanced. This proceeded until all four targets had been located. Five participants took part in the trials.

IV.4.4.3 RESULTS All participants were able to navigate to the targets with both uncertain and mean displays. A typical trajectory (showing also the heading sensed by the device) is shown in Figure IV.18. Time to complete the task was generally reduced under the uncertain display condition (Figure IV.19). The total energy of the heading signal (i.e. a measure of how much scanning activity the participants exerted during navigation) was also reduced for all participants. The energy results are summarised in Figure IV.20.

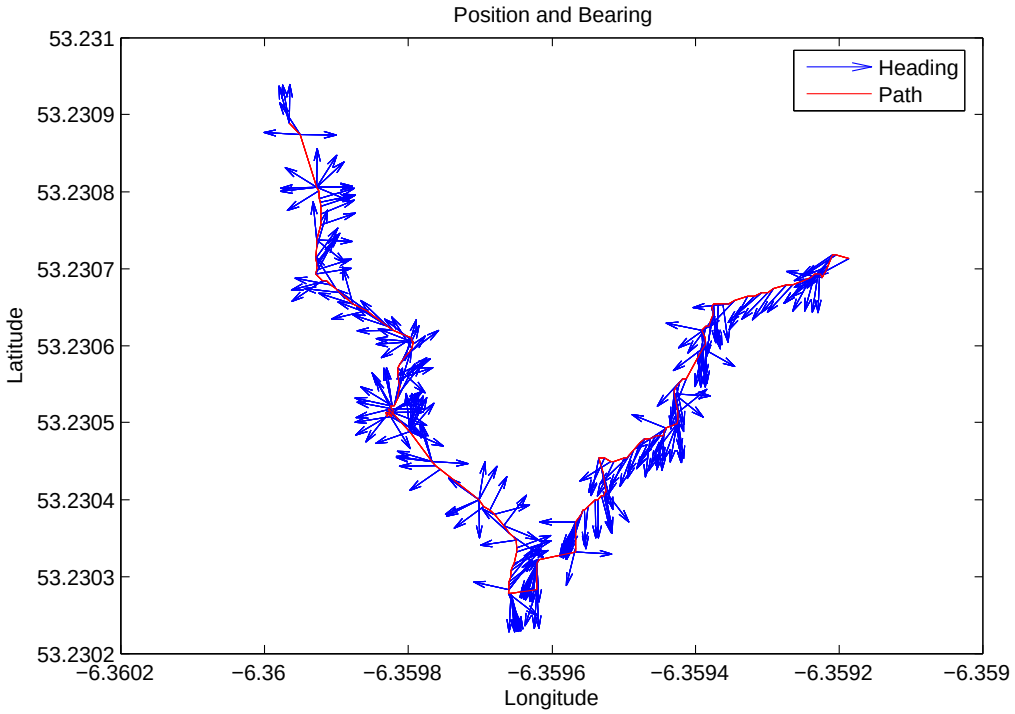


FIGURE IV.18: Position and heading of a typical user during the navigation task. Positions are shown as blue points; the arrows indicate the heading sensed by the inertial sensors.

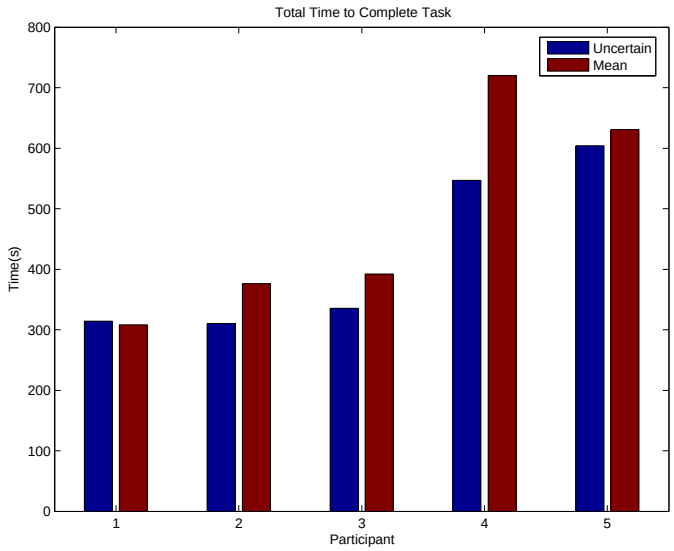


FIGURE IV.19: Time to complete task in mean and uncertain display cases. Time is reduced for all participants except the first in the uncertain display condition.

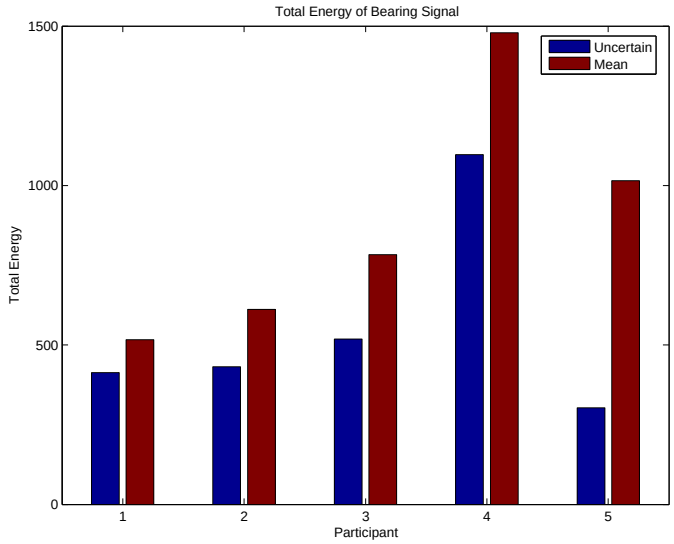


FIGURE IV.20: Heading signal energy in mean and uncertain display cases (energy is computed as $\sum_{t=0}^{T} \frac{dk^2}{dt} \frac{1}{T}$) . Energy is reduced in the uncertain display condition.

IV.4.5 Uncertain Orientation Display

The complete navigation system is quite complex, and it is difficult to control the parameters of an experiment examining the complete navigation system well enough to evaluate the effect of the uncertain versus mean display. A refined experiment was subsequently conducted to examine the effect of uncertain auditory display in orientation targeting tasks without the distraction of navigation in a physical environment. In this simplified setup, participants were asked to identify targets located in orientation space (i.e distributed around them at a constant distance). The magnetometer and accelerometer of the MESH device were used to ascertain the current orientation of the device relative to these targets. As in the previous example, the uncertain, particulate display was compared with a naïve mean display.

IV.4.5.1 Experimental Details

Five targets were laid out in a hundred and eighty degree arc around the participant. The targets were at a simulated distance of $\sim$71m. Each target had to be acquired three times for each condition. Acquisition was triggered when the participant held the device such that the measured heading was within 14 degrees of the true target position for a period of 5.4 seconds. Exiting the capture zone held the timer until re-acquisition occurred. As with the navigation display case, artificial noise was introduced to simulate the effect of degraded GPS signal quality. This noise was introduced to the simulated position of the participant, and was Gaussian distributed with a standard deviation of $\sim$4m. The noise offset was updated once every three seconds, leading to a square wave like signal.

The heading signal was filtered with a low-pass filter rolling off at 8Hz to eliminate high-frequency noise (especially tremor) from the signal before being displayed and recorded. The feedback was identical to that of the previous experiment, with both granular audio (giving the identity) and vibrotactile (presence of target only) feedback present. No visual feedback was provided, except for a counter to indicate the number of targets remaining in the experimental trial.

IV.4.5.2 Conditions

Four conditions were tested: mean display with additional noise, mean display without noise, uncertain display with noise, and uncertain display without additional noise. Eight participants took part in the experiment, which was within-subjects and had counterbalanced presentation order. The experiments were conducted indoors in rooms with minimal visual distractions. Each participant had a three minute introduction and a brief demo of the system.

IV.4.5.3 Results

Although all four conditions were tested, the variation between the mean noise and uncertain noise cases are of most interest, and will be discussed in detail here. Figure IV.22 compares the time to acquire targets for each subject in these two conditions. There is a noticeable reduction in the acquisition time when the uncertain display is employed.

Figure IV.21 shows the energy ($\sum_{t=0}^{T} \left(\frac{d\theta}{dt}\right)^2$) after 63% of the error has been reduced. This focuses the results on the period after the gross acquisition phase, leaving the feedback-dependent fine-adjustment phase (see Figures IV.24 and IV.25 for time series with the 63% points marked). The energy values are generally inconclusive in this case.

The variance of the error is illustrated in Figure IV.23. This measures the "average size" of deviations. Histograms of the error for a typical case are illustrated in Figures IV.26 and IV.27, where the reduced variance is clearly visible. There is a strong reduction of the variance in the uncertain case as opposed to the mean case; errors were more likely

to be small with the uncertain display. This is partially due to the "hovering" phenomenon where participants hold just off target in the uncertain case, receiving sparse feedback but not selecting targets. In the mean case, wild overshoots are more common (see Figure IV.24).

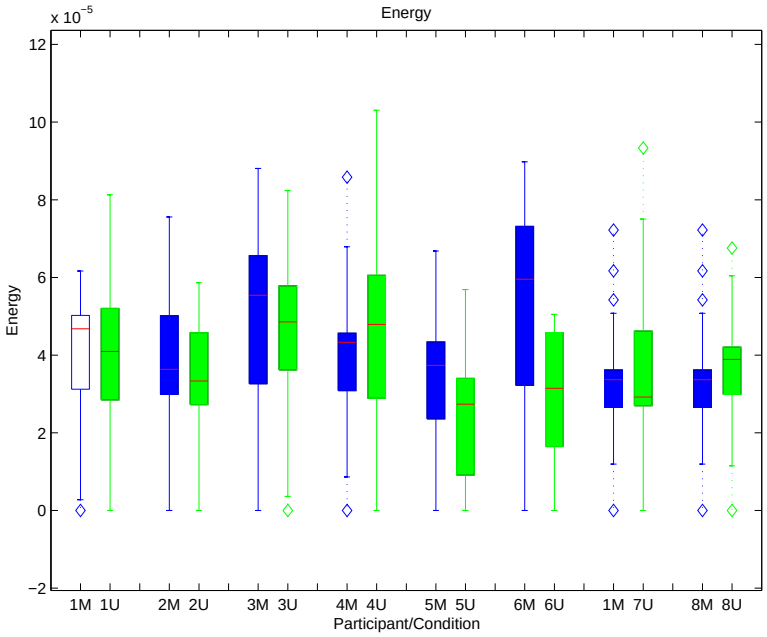


FIGURE IV.21: Energy in the 0.1–2Hz band of the heading signal for the period after a 63% reduction in error (i.e. the fine-tuning phase). The blue box plots show the energy for the mean condition, while the green show the uncertain condition.

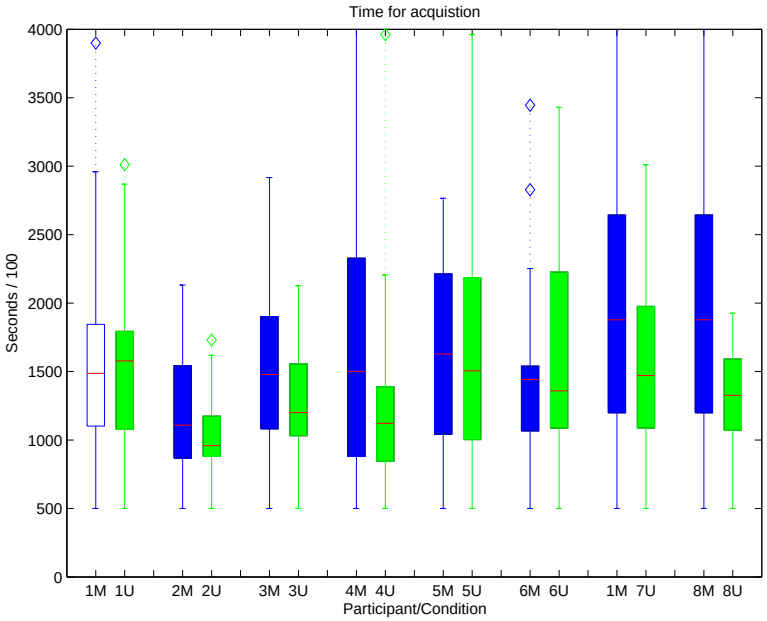


FIGURE IV.22: Boxplots of acquisition times for orientation acquisition. Again, blue represents the mean condition, green the uncertain.

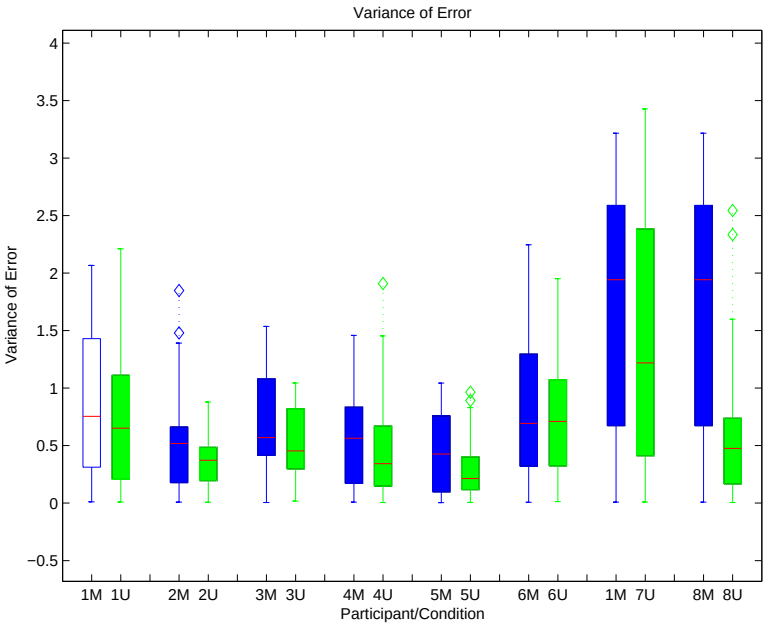


FIGURE IV.23: Variance of error for orientation acquisition. Blue shows the variance of the error (true angle-sensed angle) for the mean condition, green for the uncertain.

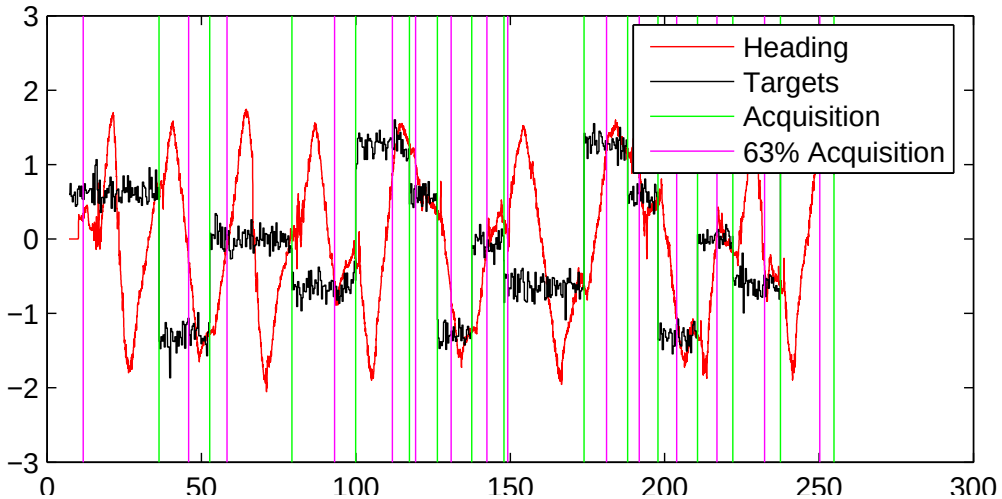


FIGURE IV.24: Time series for a typical experimental run in the mean noisy case. The start and end of the acquisitions are marked, along with the point at which 63% of the error was removed (this was the criterion used to determine the end of the gross movement phase).

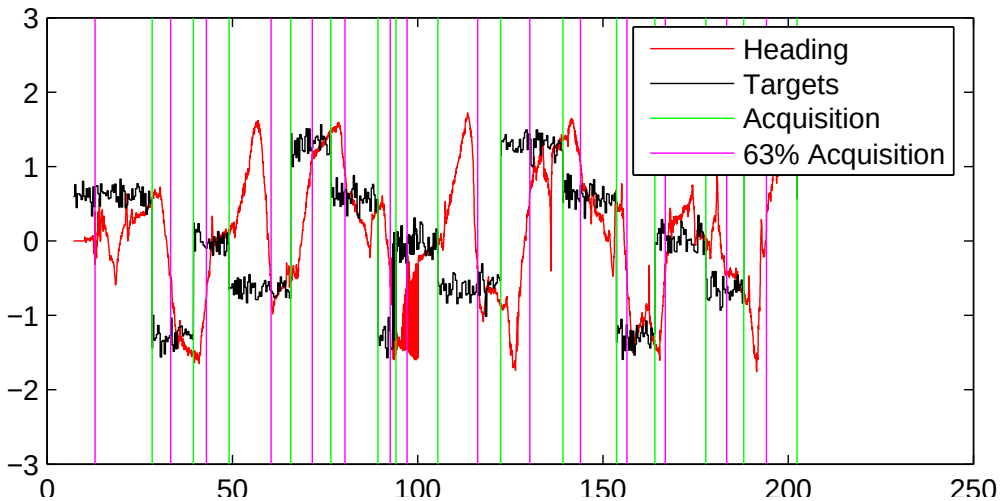


FIGURE IV.25: Time series for a typical experimental run in the uncertain noisy case. The start and end of the acquisitions are marked, along with the point at which 63% of the error was removed. Compared with Figure IV.24, there is less wild overshooting in this case, although participants occasionally "hover" just outside the capture zone.

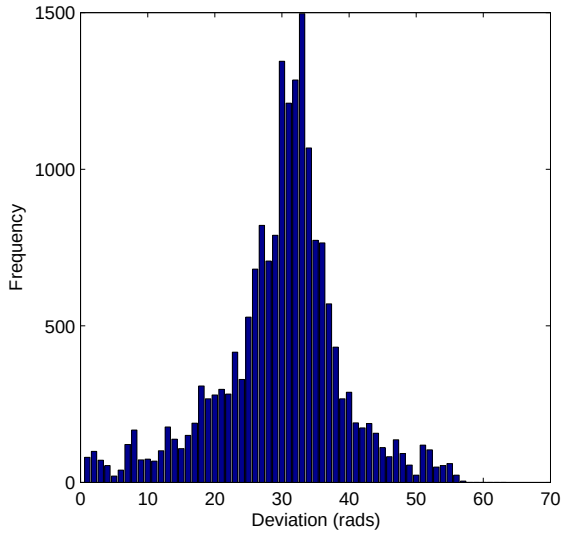


FIGURE IV.26: Histogram of error for one experimental trial in the mean noisy case.

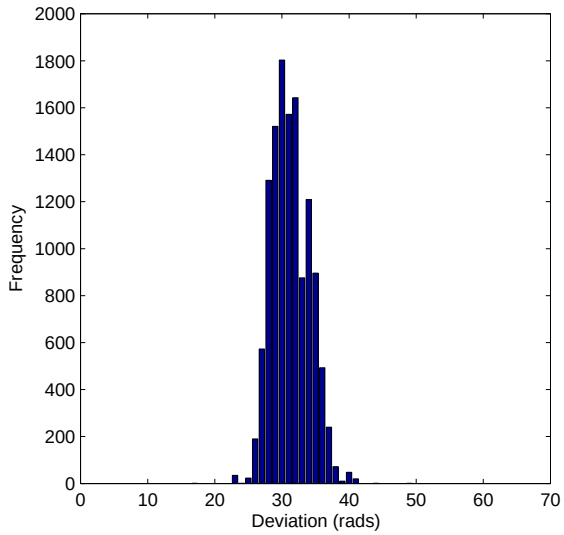


FIGURE IV.27: Histogram of error for one experimental trial in the uncertain noisy case. The distribution is significantly narrower than that show in Figure IV.26 for the mean case.

IV.4.6 Discussion

These experiments showed moderate improvement in performance when a simple Monte Carlo uncertain display was employed, versus the naïve display case. There was also a qualitative change in the behaviour the participants exhibited, with a reduction in overshooting and searching behaviour, but an increase in small, long-term errors. This is likely to be due to the production of feedback even when the orientation of the device is outside of the target capture zone, as a result of the limited modelling of the noise expected to be present during the task.

The relatively small amount of noise introduced to the signal, and the the very tolerant conditions for acquisition (nearly 15 degree funnel), may have made it excessively easy to select the targets in the mean condition. More realistic conditions, with a tighter acquisition requirement would probably favour the uncertain condition more strongly. The Monte Carlo distribution for the uncertain case was also not optimal for the conditions present; a better choice may improve the performance of such an exploration system. However, the experiments do indicate that even the simple uncertain display used here does not confuse users, and reduces negative aspects of their performance.

IV.5 Particle Filtering

IV.5.1 The Filtering Process

Particle filtering (See Ristic et al. [2004] and Doucet et al. [2001] for overviews of the field),[12] is a powerful Monte Carlo technique for the estimation of hidden variables in dynamic systems. It is effectively a generalised Kalman filter (see Kalman [1960] for the Kalman's original paper; Maybeck [1979] has a more readable and in depth discussion), without restriction to Gaussian noise or linear dynamics. It is used to recursively estimate the properties of a dynamic system where the measurements and dynamical model are held to be uncertain. It facilitates direct implementation of Bayesian inference; at each time step a prior is combined with evidence, and the posterior used as the prior for the next step. Monte Carlo sampling makes the use of a wide class of distributions possible – analytical approximations to the distributions are not required. It is reasonably efficient even in moderately high-dimensional problems, such as contour tracking in image recognition (Isard and Blake [1998b] introduced the technique – which they called "condensation" – into computer vision for exactly this purpose), and can be partially parallelised in some cases. It can be combined with Kalman filters for very efficient estimation of parameters, where some part of a system are linear with Gaussian noise, while others are nonlinear or have different noise properties. Dynamic Bayes nets can be estimated with particle filtering approaches (Murphy and Russell. [2001]); stochastic analogues of the HMM are, for example, possible.

The basic process of an importance sampling particle filter (the most commonly employed type – see the aforementioned references for more exotic variants) is: draw samples from some prior; apply system dynamics; weight the transformed samples by some measured evidence; normalise the weights; and finally draw new samples according to these weights (i.e. a survival-of-the-fittest process). In many algorithm variants, some random proportion of samples are redrawn from the original prior to prevent the distribution collapsing to a point. These samples are then used to form the new prior at

[12]The technique is also known as the condensation algorithm, Monte Carlo filtering, survival-of-the fittest filtering, bootstrap filtering and possibly other names. Particle filtering is the most widely used terminology.

the next time step.[13] This process is illustrated in Figure IV.28. The system dynamics and priors can be arbitrarily chosen, making it a very flexible way of implementing probabilistic models directly from their equations. Particle filters are also temporally flexible. They can be run forward without evidence updates to form predictions; they can also be run backwards (particle smoothing – Isard and Blake [1998a] give a concrete implementation) to re-estimate past states in the light of new evidence.

This technique is obviously similar to the processes described in previous sections for the prediction of future states of the system, with the addition of a selection step to the process. As such, it can be sonified and displayed in much the same manner – placing densities on the (hidden) state space, estimating the values for an ensemble of particles, and drawing grains according to the distribution of the ensemble.

IV.5.2 Implementation: A Sonified PF Gesture Recogniser

This section describes an example of the implementation of the sonification and particle display techniques in a particle filtering context. It demonstrates how uncertain, predictive displays can be combined with inference mechanism in a natural way.

This implementation uses a modified version of the Black-Jepson temporal trajectory matching algorithm (Black and Jepson [1998]) which implements a type of probabilistic time-warping matcher. In this implementation the algorithm has been adapted to track the initial offset of the gesture. All priors are Gaussian (or half-Gaussian for positive-only values). Gestures are two-dimensional and are performed using the mouse in this implementation. The system displays the particle class and current estimated parameters visually (by transforming the model parameters back into trajectories); it uses granular synthesis to display the class of the gesture (via waveform source) and its estimated phase (via the time index parameter). The time display is fully probabilistic; when uncertainty in the phase parameter is present, the sound becomes temporally blurred.

The display is also predictive; it shows possible evolutions of state which are achievable from the current assumptions. These are produced by running the particle filter forward without any sensory input (i.e. without the evidence weighting step). This continuously shows potential completions of the current motion; it is *self-revealing*.

IV.5.2.1 Estimation

The tracker estimates the following state variables:

- ϕ Phase within the trajectory, with an initial half-Gaussian (positive only) prior with mean 0, variance σ_ϕ and a Gaussian diffusion distribution $\mathcal{N}(0, \sigma_{d\phi})$;
- ω Phase rate (the increment of ϕ at each time step), Gaussian prior $\mathcal{N}(1, \sigma_\omega)$ and a Gaussian diffusion distribution $\mathcal{N}(0, \sigma_{d\omega})$;
- x, y Offset from start of gesture; initial uniform prior bounded to size of available window, diffusion distributions $\mathcal{N}(0, \sigma_{dx})$ and $\mathcal{N}(0, \sigma_{dy})$ which are usually equal;
- α Global scaling of gesture, with Gaussian initial prior $\mathcal{N}(1, \sigma_\alpha)$ and diffusion distribution $\mathcal{N}(1, \sigma_{d_\alpha})$;
- τ Gesture class with uniform discrete prior over all classes and a uniform diffusion distribution.

Comparison is made over a time window for estimating the likelihood of parameters, computing the SSE between the estimated and actual values for n samples.[14] Since

[13] The prior can either be represented *directly* by the samples, or can be smoothed by placing kernels on the samples and drawing samples from the resulting mixture (regularised particle filtering).

[14] At 50Hz sampling, $n = 40$ is an effective value, i.e. just less than one second time window.

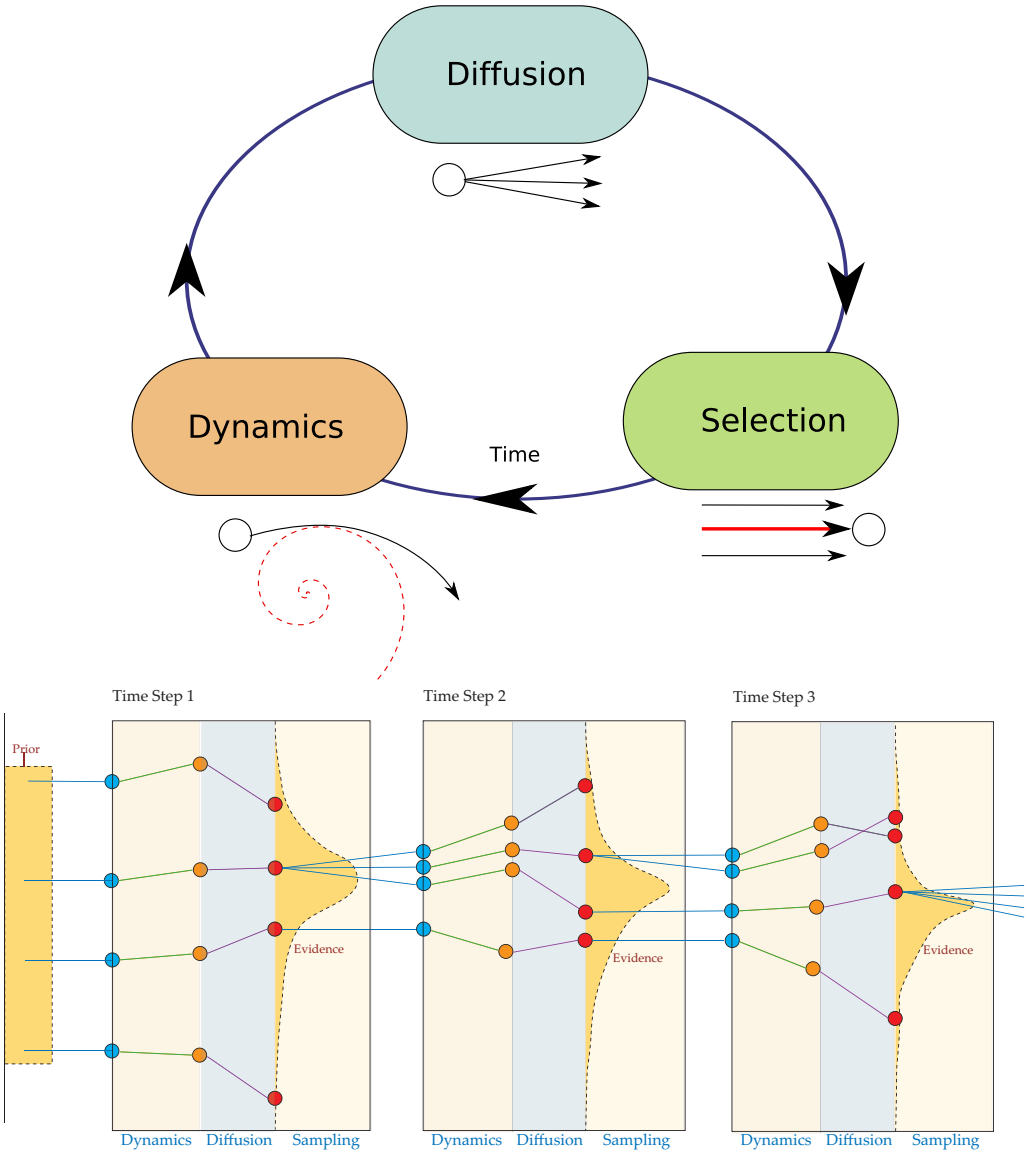


FIGURE IV.28: Illustration of the basic elements of a one-dimensional particle filter. Each time step involves deterministic dynamics, diffusion and importance sampling. The similarity to the Monte Carlo prediction methods is apparent. The major difference is the introduction of an importance sampling step.

the comparison is made over this window only, the "global" parameters are effectively local, and the system tracks these parameters as they change. For one dimension, given a history of sensor input $x_1 \ldots x_n$, a set of gesture examples $g(t)_1 \ldots g(t)_m$ a set of particles $s^1 \ldots s^k$, the comparison returns a value for each s^i equal to:

$$\sum_{j=0}^{n} s^i_\alpha g_{s^i_\tau}(\phi') + s^i_x - x(T-j))^2, \tag{IV.3}$$

with

$$\phi' = (n-j)s^i_\omega + s^i_\phi. \tag{IV.4}$$

The process is identical for other dimensions; the comparison weights are summed.

The recognition process proceeds as follows:

- Sample from the parameter priors for each particle;
- For each timestep:
  - Apply the system dynamics (in this case, just update the phase parameter);
  - Apply diffusion to all parameters according to their diffusion density;
  - Compare the predicted and actual values across the current time window, for each particle, getting the particle weights;
  - Normalise the weights;
  - Sample from the old population according to the new weights;
  - Replace x of the samples with samples drawn from the original prior (where x is around 5–10% of the original number of samples).

Approximately one thousand particles are used in the demonstration algorithm. The priors and diffusion parameters are manually tuned. This give reasonable performance for a small number of gestures shapes (< 10), even with a single example of each gesture class, and is quite sufficient to illustrate the display techniques. The implemented algorithm is a very simple – much more powerful (and reliable) inference algorithms could be used in place. The technique generalises easily.

Images from the recognition process in this demonstration are given in Figure IV.29, showing the coalescence of particles as recognition proceeds. Display uses alpha blended points for the current state, applying the transformation parameters for each particle to the original trajectories to obtain screen positions. Similarly, partially transparent lines showing the future evolution of particles are drawn extending from these points. Granular audio generation is exactly as in the previous examples, with class mapped to waveform and phase mapped to time index.

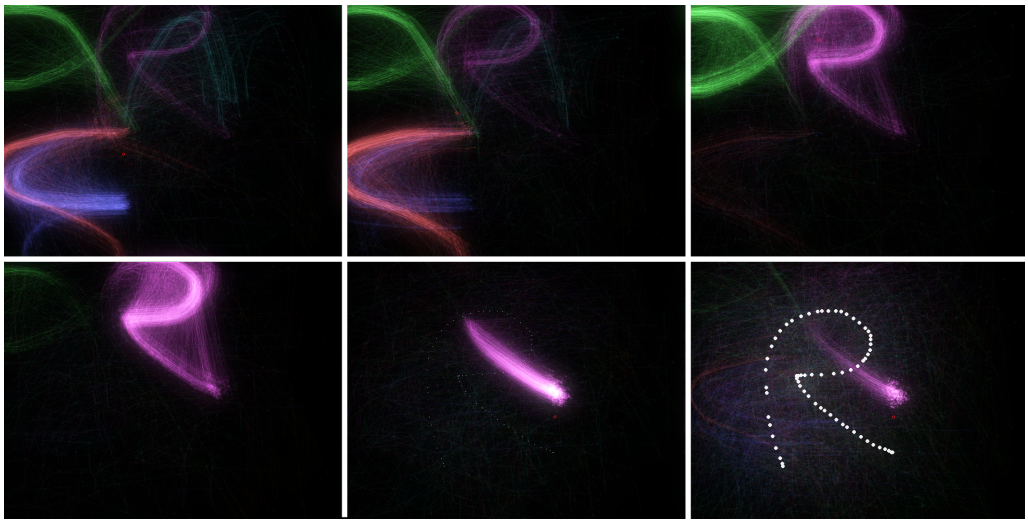


FIGURE IV.29: Sequence of screenshots from the particle filter gesture recogniser during the performance of an uppercase "R"-shaped gesture. The states of the particles are drawn as partially transparent circles, with lines extending showing the future state of the particle. Each gesture class (i.e. each letter shape) is assigned a different colour. The result is nebulous display of the distribution of particles.

IV.5.2.2 DISCUSSION

Particle filters are an extremely powerful technique for inference, and will become more so as computing power continues to increase. They can be applied in a wide range of contexts, and are theoretically sound. They are temporally flexible, and compatible with a Bayesian interpretation of the world. The display and sonification techniques described here generalise well to other particle filtering problems (at least where defined classes of model are present); they make it possible to build systems which have sophisticated inference and matching probabilistic feedback. Feedback of the type presented exposes the inference to the interactor. Interactors can observe their effect upon the inferential process, and, with predictive displays, how actions affect the behaviour of the system in the future.

IV.6 CONCLUSIONS

Delays can have a crippling effect on the the efficiency of an interaction – performance degrades rapidly as response time increases. Predictive displays (along with preview displays) can help improve performance in the presence of such delays, reducing the modelling load on the user. The previously introduced probabilistic display methods can be extended to cope with predictive power, and the process by which this can be done has been illustrated with a number of specific examples. The Monte Carlo processes described here are general enough to be applied to many interactive systems, and lend themselves well to display. It provides temporal flexibility in the display of results, and can remain appropriately uncertain. The quality of a particular display depends on the quality of the prediction model, and the quality of the sampling process.

The Monte Carlo predictive process is similar to the particle filtering process. Consequently,particulate audio and visual displays can be adapted to display particle filter state, including both current and future estimates of the state of the system.

CHAPTER V

AUGMENTED DYNAMICS FOR INTERACTION ENHANCEMENT

Making likely things easy.

> *The swimming fish has the quality of integrated skilled motion. The designer of the manual control system should strive to integrate the man into the system in such a way that this same quality is evident.*
>
> – Kelley [1968]

V.1 SUMMARY

MODULATED dynamics is presented as a powerful technique for interaction design. The basic principle is that the user should stay in control of a relatively simple, familiar dynamic system whose properties are gently manipulated according to the evolution of a more complex inference process. This is shown to be a unifying approach, and one which provides richer opportunities for extension than many existing *ad hoc* techniques. As an example of the method in practice, a text entry system for mobile devices is developed, which uses the augmented dynamics principle to bring language modelling into a continuously controllable system.

V.2 AUGMENTING DYNAMICS; CHANGING FEEDBACK

The control-oriented view of interaction presented in previous chapters requires feedback for meaningful interaction. In Section II.10, the breakdown of control loops into internal (not directly accessible) and external (within the domain of the interface) was described. This chapter focuses on how adjusting the dynamics of external loops can introduce intelligence to an interaction in a smooth and non-disturbing manner. The fundamental goal is that *likely things should be easy to do*; if there is already strong evidence for some particular action the user should have to expend less energy in confirming that hypothesis than some other less likely potential goal. Augmented system dynamics can be used to create a system with a simple, smooth structure that can also incorporate significant intelligence. Figure V.1 illustrates this idea.

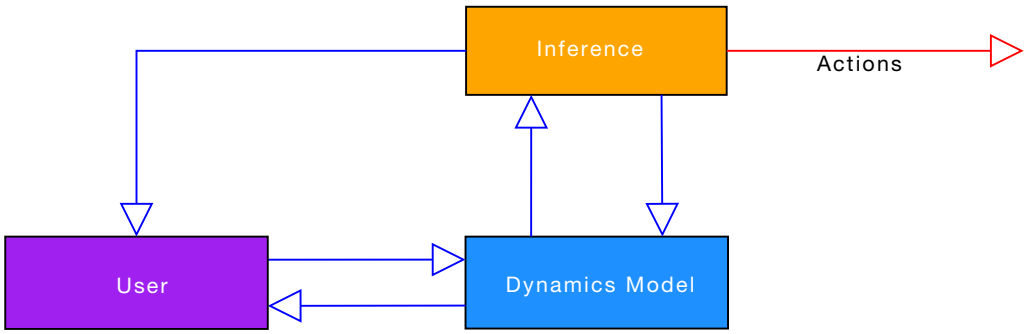


FIGURE V.1: Feeding the results of the inference back into the control loop.

The potential advantage of augmenting the dynamics over other approaches is that the process does not wrest control from the user and thus make decisions for them before sufficient evidence has been accumulated (as opposed to the overzealous autocompletes present in some interfaces). Instead, it smoothly adjusts the interface properties to make the input space trajectories required to reach parts of the goal space less costly from the user to perform. The continuity of process means that the user's knowledge of the evolution of ordinary dynamical systems is still valid within the realm of the interaction process; unpredictable jumps of state are minimised. This is the advantage of the modulated-dynamics approach over discrete automatic completion systems. It is particularly powerful when combined with a probabilistic predictive display, such as those described in Chapter IV.

In Hermann and Ritter [1999], "model based sonification" was proposed for the exploration of data sets. The authors note:

> ..why not sonify data spaces by taking the environmental sound production in our real world as a model? Nature has optimised our auditory senses to extract information from the auditory signal that is produced by our physical environment. Thus the idea is: (i) build a virtual scenario from the data, (ii) define a kind of virtual physics that permits vibrational reaction of its elements to external excitations, (iii) let the user interactively excite the system and listen.

The augmented dynamics approach applies the same basic principles to the entire interaction problem; by building dynamic systems which behave in some way like the physics to which humans are adapted, rich interaction can be implemented in a way that users can relate to. Because the dynamics of the software-created world are flexible, long term "intelligent" behaviour can be brought into the virtual physics without disrupting the basic behaviour of the system.

The key idea is that long-term behaviour (from the inference model) is incorporated into the evolution of the current state without requiring a very high-dimensional display. The extension of the input backwards in time – with respect to the goals – is the advantage of this approach. The effects of long-term behaviour, stretching for the user's actions in the last few milliseconds, right through the complete history of interaction with the system can be brought into the interaction in an appropriate and controllable manner. Rather than defining specific patterns of movement which are to be achieved, the movements required to indicate intention develop as evidence accumulates. Con-

text variables, sensed from the environment or from the user, can be smoothly brought into the interaction without requiring discrete (and disturbing) mode-shifts. As a simple example, a system which can infer whether an interactor is moving or is stable can adaptively dampen particular frequency ranges in its response; this can, for example, be used to adaptively minimise gait noise without switching the user into a special "walking mode".

The extent to which this extension can be done without incomprehensible displays depends on the ability of the user to model the system. High quality real-time feedback, as discussed in the previous chapter, is one of the things that can maximise the benefit of these methods.

In order to implement this augmentation, a suitable definition of "easy" must be made. This corresponds to the cost-per-bit of communicating (as described in II.2.2). This cost is derived from the physiological and cognitive constraints of the user. Some important constraints among these are limb dynamics, response times and control error models. Limb dynamic limitations are extensively discussed in Schmidt and Lee [2005] and Latash [1993]. Flash and Hogan [1985] describe the minimum-jerk model, one of the most widely used models of human motor control in trajectory tasks. Human control models and response times are discussed in Jagacinski and Flach [2003], Schmidt and Lee [1999] and in Kelley [1968].

V.2.1 Semantic Pointing and Other Interface Enhancements

There have been a number of recent attempts to improve performance in pointing tasks by dynamically altering the response of the system. These methods include semantic pointing (Blanch et al. [2004], Blanch [2005]), object pointing (Guiard et al. [2004]) and "bubble-cursor" pointing (Grossman and Balakrishnan [2005a]). All of these attempt to reduce the effective distance to a target in a pointing task, based on some estimate of likely user behaviour, thus making interaction more efficient. These attempt to ameliorate the effect of selection on a surface where only parts of the display have functional value (for example, icons lying on a desktop).

V.2.1.1 Object pointing

Object pointing simply warps over unused (non-information bearing) regions of the display space so that targeting requires less movement, at the cost of making the structure of the space less intuitive. The motor space image of the workspace is a dense tessellation of targets, with minimal space in between, while the visual image remains undisturbed. Guiard et al. [2004] found that for some test interfaces, this significantly improved selection time.

V.2.1.2 Semantic pointing

Semantic pointing centres around adjustment of the *control-display ratio*. This defines the gradient of the linear relationship between sensor inputs and displayed response of a cursor. The ratio gain is adapted so that more commonly used or more useful parts of a conventional GUI have lower gain and vice versa; the cursor "sticks" to likely GUI controls and slips quickly over less useful areas. This effectively adjusts the motor space image of the interface such that regions are scaled according to their utility (see Figure V.2). In Blanch et al. [2004], faster targeting times and reduced error rates were observed in the experimental setup, which presumably had accurate identification of areas of of utility.

FIGURE V.2: The visual space (a) and motor space image (b) of a dialog box with variable control-display ratios. The "save" button, which has higher utility to the user, is enlarged in motor space. From Blanch et al. [2004].

V.2.1.3 AREA CURSORS Bubble pointing (Grossman and Balakrishnan [2005a]) uses a cursor with non-zero effective area (other area based cursors are described in Kabbash and Buxton [1995]); the cursor can be "over" multiple regions of interest at one time. The "bubble" is a circular object which represents the region of activation associated with the cursor. Objects are selected when they are under this area and activation (mouse click) occurs. Disambiguation is achieved by selecting the nearest target to the centre when multiple targets lie under the cursor (this is also displayed to the user). This effectively increases the size of targets for selection. While this can provide improved performance (as the authors demonstrate), it does not increase the richness of selection. A model where the activation region was a density instead of an activation area, and targets had associated priors would allow simple probabilistic inference to be brought into the process. The bubble size (in bubble pointing) is also fixed and is not responsive to changes in input dynamics (e.g. higher velocity might imply a greater tolerance for larger selection areas along the axis of motion).

V.2.1.4 PSEUDO-HAPTICS Pseudo-haptics (Lécuyer et al. [2004], Lécuyer et al. [2001], Lécuyer et al. [2000]) involves the modulation of the control-display ratio as a display mechanism rather than to improve the quality of interaction quality. Lécuyer et al. [2004], for example, demonstrates that simply adjusting this ratio based upon some map of simulated surface height permitted interactors to perceive apparent textures in the surface (bumps and ridges). The disruption between the kinaesthetic feedback from the hand motion and the visual feedback is strong enough that the perception of the contours of the virtual surface are apparent.

V.3 MODULATED DYNAMICS: UNIFICATION AND EXTENSION

It is clear that the basic, unaided, spatial pointing metaphor is not the optimal way of interacting, despite its omnipresence on desktop computers. The basic dynamics of pointing devices, as described in II.11, have been extensively tuned to increase performance, with carefully calibrated acceleration curves and similar enhancements. However, this fails to address the underlying inefficiencies in the pointer-based interaction methods. The aforementioned augmentation techniques seek to improve this performance by distorting the motor space so as to be more efficient, while retaining the intuitive visual

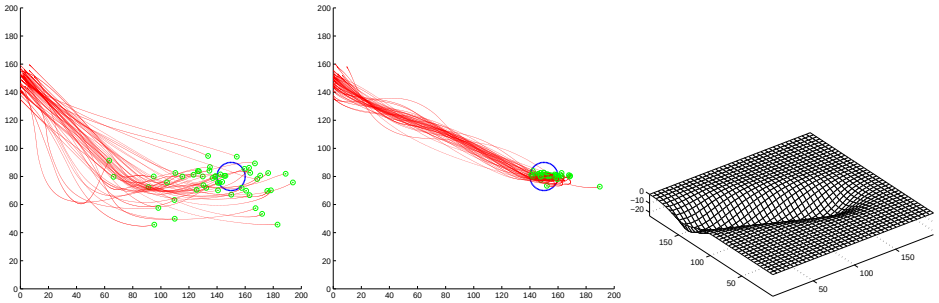


FIGURE V.3: An example of how introducing "helping" dynamics into the system can improve control performance. This figure shows a simulated acquisition task in which an operator – modelled as PID controller controlling a second order system with saturation, strong drifts and Gaussian noise – attempts to acquire a target (the circular region to the right). In the unaided case (left), many of the acquisitions miss the mark (only ∼25% acquire the target). When the simulated surface shown at the far right is introduced (a smooth "dimple" around a path heading towards the target), the acquisition is significantly cleaner (centre figure), with more than 80% of the acquisitions being succesful. This is equivalent to an increase of target size of ∼ 360%.

appearance of existing displays. This motor space distortion is a very specific example of the modulated dynamics approach to the design of interfaces. By adjusting the input-to-cursor gain so that response is more sensitive in regions of little interest, these can be quickly skipped over to reach relevant targets. The interactor's control is supported such that less effort needs to be expended to achieve likely goals. Figure V.3 shows how introducing dynamics can regularise operator behaviour and increase precision.

In the systems described above the inference is as simple as could be possible: the designer assigns prior probabilities to regions of activation (which may for example, be uniform across a class of GUI widgets). The augmented dynamics approach seeks to unify these techniques, and extend them to include both more complex, dynamic inference and more powerful dynamic systems, which offer greater opportunities for shaping user behaviour.

All of the enhanced pointing techniques have been shown to improve performance in spatial selection tasks. However, they attempt to fix the limitations of the spatial pointing metaphor, a sub-optimal way of distributing information across sensory inputs, without addressing the underlying flaws of this method. The metaphor has advantages in terms of familiarity, but the failure to incorporate the temporal richness of the observed signals dramatically limits the efficiency of any such selection mechanism. By squeezing all of the evidence used for the selection into the last instant (the terminal cursor location), or a very simple model of the dynamics (e.g. using just the estimated cursor velocity), much of the relevant data about the user's intention is lost. They also fail to incorporate probabilistic models of likely user actions; all goals are either assumed to be equally likely or to have *a priori* likelihoods (not dynamically updated given the evidence observed).

Conventional "gesture recognition" systems use (or can use) all of this sensory information in making decisions; but the prevailing action/response methods make gesture input frustrating and challenging to learn for anything but the simples gesture repertoires. The aim of the approach described here is to bridge this gap, maintaining the controllability and apparent simplicity of spatial or physically-inspired models, with-

out excluding the rich spatio-temporal communication possibilities that gesture-based systems offer.

V.3.1 A Goal Space View of Optimal Dynamics

In terms of the goal space view presented in Section II.7, a system with optimal dynamics requires the least effort on the part of the user to move across the action boundaries in the goal space such that effort remains proportional to the product of the probability and utility of the potential outcome. The designer of a system should ensure that suitable dynamics are implemented in an interface. This involves trading off the "intelligence" of the system against the controllability of the result. A system which cannot be accurately modelled and thus controlled by an interactor will fail to be maximally effective even with the most sophisticated inference models. By introducing smooth feedback from an inference model which estimates the long term goals of the user to the immediate-term control system, such dynamics can be created while maintaining separation between the dynamical design and the inference mechanisms.

V.4 Hex: An Example

This section illustrates the application of these ideas to a real-world problem: text entry on mobile devices. The existence of well-developed inference models for text entry makes it a suitable example for the application of the theoretical principles of dynamical design. The result is Hex, a text entry interface running on a mobile platform augmented with inertial sensors. These sections reviews some of the issues involved in entering text on mobile devices, some competitive solutions to the problem, and the basics of inertial interaction. Hex is then described in detail, and its performance is analysed.

V.4.1 The Mobile Text Entry Problem

Entering text on small mobile devices is a serious practical issue. Mobile devices now pervade most peoples lives, from the ubiquitous mobile phone to personal digital assistants and even "wearable" computers. Entering text, which is one of the most fundamental tasks a user might want to do, is still awkward on these devices, despite huge research efforts to develop better alternatives. Almost all SMS text messages are entered using the numeric keypad approach (usually with Tegic's T9 predictive text system, see below), a difficult, slow and error-prone process. Even given these practical limitations, SMS messaging is enormously popular. With better interaction, the flow of messaging could be made significantly more fluid.

There are several important issues in the design of mobile text entry systems: obviously a system must support reasonable input rates; it should not require disproportionate learning time (this, for example, is the major reason for the non-adoption of chording keyboards despite their superior performance and compact size); it should not be excessively frustrating; it must be feasible to operate within the limited feedback constraints of mobile operation (e.g. partially eyes-free); it should avoiding "colouring" the user's input by strongly favouring certain sequences over others; and it must make effective use of the sensory hardware available on a mobile device (it should not, for example, require power-draining or bulky sensors, and should be able to cope with noise sources such as would be generated by walking or travelling on a vehicle). These requirements place fairly stringent constraints on the design of new interfaces. In particular, the feedback, sensor size and noise rejection issues are of great importance in mobile interface design.

System	Rates	Lang. model	Geometry	Traj. type	Input
Quikwriting	none	none	octa.	2 stroke	stylus
TCube	~ 20	none	hex.	2 stroke	stylus
Dasher	20–34	PPM	tree	cont.	mouse/eye^{+}
Shark	45–70	bigram*	hex.	cont.	stylus
SonicText	none	none	hex.	2 stroke	thumbstick
Cirrin	~ 20	none	circle	cont.	stylus
TiltText	~ 13	T9	list	scroll	acc.+buttons
Unigesture	1–4	lookup	hepta.	7-way tilt	acc.
TUP	6–7	trigram	circle	tap	touchwheel
T9	10–25	lookup	grid	button	buttons
Hex	~ 17	modified PPM	hex.	cont.	acc./EEG

Table V.1: A comparison of the attributes of the text entry methods listed in this section. Rates are given in words per minute. "cont." indicates a continuous trajectory system, while "2 stroke" indicates a two stroke entry method (two individual stroke motions). "acc." indicates accelerometer input. *Shark uses a smoothed bigram model. $^{+}$Dasher runs on a number of of other devices, including breath control and buttons.

V.4.2 Competitive Approaches

There is now a vast proliferation of text-entry methods for different devices and contexts, although few have gained widespread acceptance. This section gives an overview of existing entry methods suitable for mobile devices, especially those featuring continuous gestural movements. A review of some methods not discussed here is given in MacKenzie and Soukoreff [2002]. Table V.1 gives an overview of the various methods reviewed in this section.

V.4.2.1 Quikwriting

Quikwriting (Perlin [1998]) is a continuous gestural input method for stylus input devices. It uses gestures based on moving back and forth to sections of a input area, which is divided into eight parts. No online language model is used, although the layout is hand-optimised for quick entry. No data entry rates for the system are available. The system requires two handed interaction and a touch-screen or other stylus input.

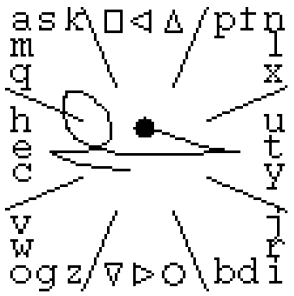


Figure V.4: The Quikwriting text entry system. Text is entered by moving back and forth from the octagonal sections. Taken from Perlin [1998].

V.4.2.2 TCube

T-Cube (Venolia and Neiberg [1994]) is a text entry system for stylus based devices based on flick motions on an eight way pop-up pie menu. Each character requires two sequential motions to select. The layout of characters is stable and does not involve any language modelling (the ordering is sequential). Rates of up to 20 wpm are reported after nine thirty minute training sessions.

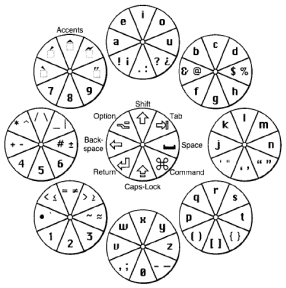


FIGURE V.5: TCube text entry. Entry is based around pie menus, which the user can pop up at any position on screen. From Venolia and Neiberg [1994].

V.4.2.3 DASHER Dasher (Ward et al. [2000], Ward and MacKay [2002]) is a probabilistic text entry system based on continuous one dimensional movement. The interactor "flies" through an expanding array of letters which move from left to right. The area associated with each letter is determined by its probability, as given by a PPM prediction model. Users are able to communicate at input rates of approximately ∼20–34 words per minute with Dasher. Originally controlled by mouse and eye-tracking input it has been extended to support breath control, and also a hybrid speech recognition mode (Vertanen [2004]). Although Dasher is a very efficient entry system, especially where control is difficult, or the language or other structure is unfamiliar, the dynamic layout of letters reduces the opportunity for learning highly optimised strategies for entry, and requires continuous attention to control.

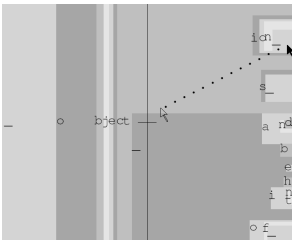


FIGURE V.6: Dasher continuous text entry system. Text is entered as a continuous one-dimensional gesture. The sizes of the targets are adapted according to a probability model. Taken from Ward et al. [2000].

V.4.2.4 SHARK Shark (Kristensson and Zhai [2004], Zhai and Kristensson [2003]) (now rebranded ShapeWriter) is a soft keyboard based entry system. Entry is performed with continuous gestures drawn with a stylus onto a soft keyboard. The path through the keyboard is optimised according to a language model so that trajectories close to those of likely words are warped into the correct shape (this is performed after the completion of a word gesture). The authors suggest an impressive speed of around 45 words per minute is achievable with the system – with short sequences reaching up to ∼70 wpm. The use of word length gestures allows the language model to have a greater role in the decision process, but at the same time reduces the amount of direct control the user has over the operation of the interface. The system offers substantial room for users to improve their performance through learning, as the optimal trajectories for particular words are unchanging. Shark also displays the optimal word trajectory after warping has been performed, showing exactly how performance could be improved. Shark, however, requires two handed interaction, and a reasonably sized touch-sensitive screen.

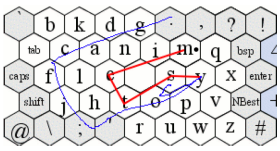


FIGURE V.7: Shark gestural text entry system. Words are entered as continuous gesture on the virtual keyboard display. The system then uses a language model to deform the path to the path for the most likely word given the gesture. Taken from Kristensson and Zhai [2004].

V.4.2.5 SONICTEXT SonicText (Rinott [2004], Rinott [2005]) is a mobile text entry system using a spring-centred eight way joystick with audio feedback. It is intended to be usable without visual attention (e.g. in a pocket). Symbols are coded similarly to Quikwriting (Perlin [1998]); movements consist of a directional movement followed by an optional rotation motion, and then a return to the centre position. Each symbol thus requires either one or two movements. Audio feedback representing the symbol currently acquired is produced during the interaction. The feedback has two modes: phonetic (sound of the letter) or synthetic (abstract sounds representing the letter). Since acquisition is only completed once the user releases the joystick, returning it to the sensor, the feedback can be used to explore the space. No entry rates are given for the device, but it is claimed that entry is possible without visual attention.

FIGURE V.8: SonicText. The joystick can move in eight directions; each letter requires either a single directional movement, or a directional movement followed by a "roll". Audio feedback is presented during the interaction to reduce the need for visual attention. From Rinott [2004].

V.4.2.6 CIRRIN The Cirrin interface (Mankoff and Abowd [1998]) is a gestural text entry method, based on continuous gestures weaving in and out of a circular array of letters. It has no support for online language models (although the fixed layout of letters is pre-optimised via simulated annealing). It is intended for use with stylus interfaces, where entire sequences can be entered with a single gestural stroke. One user is reported to have achieved 20 wpm with the system. The lack of any dynamic language modelling reduces it potential for high input rates, but does make for a stable interface which can be mastered with experience. Cirrin requires a two-handed, touch screen interface, making unsuitable for some applications.

V.4.2.7 TILTTYPE AND TILTTEXT TiltType (Partridge et al. [2002]) is a simple text entry method for extremely compact devices. It uses a four buttons and a tilt sensing accelerometer. The device can be tilted in nine different ways which, combined with three of the four buttons, allows for 27 characters. The fourth button functions either as a shift key (to access numerals and punctuation) or as a backspace key, for quick correction. The user can hold down a button and scroll through letters until the desired one is acquired; releasing the button causes the letter to be stored. No rates are given for entry, but the authors quote the speed as "slow". It re-

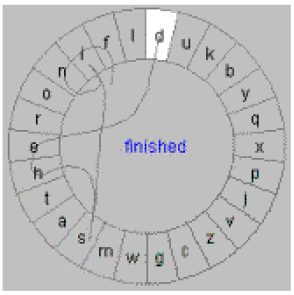


FIGURE V.9: The Cirrin text entry interface. Text is entered as a continuous movement, entering and leaving one segment of the circle to produce a character. The layout is chosen to reduce user effort for likely words. From Mankoff and Abowd [1998].

quires two hands for use. TiltText (Wigdor and Balakrishnan [2003]) is a similar system, which uses a full 12-key mobile phone keypad, and uses tilt to disambiguate letters. It effectively implements T9, but uses the accelerometer in place of the "cycle possibilities" button. It is two handed, and speeds of up to 13wpm are reported (somewhat slower than true T9 entry *without* accelerometer control).

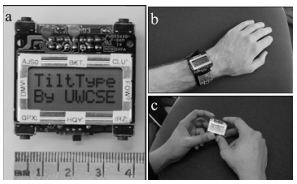


FIGURE V.10: TiltText Tilting text entry with accelerometers. From Partridge et al. [2002].

V.4.2.8 UNIGESTURE

Unigesture (Sazawal et al. [2002]) is another tilt-based approach to text entry. In this model, orientation space is divided into eight regions (seven for characters, one for space). Users tilt into one such zone to select a character; each tilt motion is recognised as an atomic gesture. When the space character is reached, the word is disambiguated by a language model. Users then push a button until the desired word is displayed. Words not in the system dictionary cannot be entered. This method is one handed, except that buttons must be pressed for disambiguation in the prototype system. Speeds of 3–8 seconds per character are quoted for the prototype; this corresponds to about 1–4 wpm.

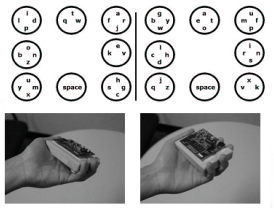


FIGURE V.11: Unigesture tilt text entry. Orientation is divided into seven zones plus one for space. Users tilt the device into a zone for each character; the system disambiguates the letters at the end of a word. From Sazawal et al. [2002].

V.4.2.9 TUP TOUCH WHEEL

The TUP touchwheel interface (Proschowsky et al. [2006]) is a text entry system for mobile devices which is also based on a circular arrangement of letters. However, the system is based around either tapping or rotating a wheel, rather than continuous strokes. It incorporates a probabilistic language model, as well as a model of the probability of a letter given the position of the interactor's finger (smooth densities around displayed character position). This makes for reasonably reliable entry even when the accuracy of the input is low (large fingers on a small wheel device). However, the current versions of this system do not use information about the dynamics of movement to improve interaction

and, since the model is effectively static, there is no way to shape the user's behaviour by modulating the feedback. Input rates are reported as 6–7 wpm for inexperienced users.

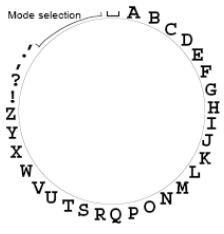


FIGURE V.12: The TUP touchwheel text entry system. The user can type by tapping on the circular pad. The system estimates the most likely letters given the position of the tap and a language model. From Proschowsky et al. [2006].

FIGURE V.13: Keypad text entry. Letters are coded as sequences of digits. In the T9 system, each letter is coded as a single digit, and the entire word decoded at the end.

V.4.2.10 T9 – THE BASELINE Nearly all mobile phones now on the market offer text entry via the numeric keypad. Although the "multi-tap" method for entry, which directly codes letters as sequences of digits is often available, it is very inefficient, and most phones offer Tegic's T9 prediction system for increasing entry rates. In this system, each letter is entered as a *single* digit, until the end of a word is reached, whereupon the possible words compatible with that sequence are decoded. The user can cycle through the possible words to disambiguate as required. James and Reischel [2001] report that T9 can achieve speeds between 10 wpm and 25 wpm (when all words to be entered are in the system dictionary). Numerical pad entry methods usually require two hands (single thumb operation is possible but clumsy), as well as careful finger co-ordination. This becomes a significant issue as the size of mobile phones continues to decrease.

V.4.3 TILT CONTROLLED INTERACTION

Since the topic at hand relates to the modulation of continuous dynamics in a control context, continuous sensing devices are of interest. Inertial sensing, where the orientation and motion of a device is directly sensed, provides rich input with small, cheap sensors. The availability of MEMS technology means that extremely small solid-state sensors can be mass produced. Accelerometers, gyroscopes and magnetometers are three major inertial sensing device types.

V.4.3.1 ACCELEROMETER DEVICES Most notable among these devices are linear accelerometers, which are widely available commercially. These sense linear acceleration on one or more axes (3-axis accelerometers are common) with a range of a few G's. They are sensitive to the effect of gravity, and as

such function as basic tilt sensors, measuring the orientation of the sensor with respect to the gravitational vector. Because of the dominating effect of the gravitational force, smaller translational accelerations are much more difficult to measure on such devices. MEMS accelerometers also suffer from calibration drift due to temperature changes. These factors rule out simple modelling of translational movements via double integration – the changing effect of gravity (due to small tilt variations) and calibration drift cause position estimates to drift wildly.

These accelerometers are, however, ideal for tilt sensing applications, and can be used with very little signal processing. They have been employed in a number of mobile tilt based interfaces. See, for example, Harrison et al. [1998] (tilting for scrolling), Eslambolchilar and Murray-Smith [2004], Eslambolchilar et al. [2004] (speed-dependent automatic zooming with tilt control) or Partridge et al. [2002] (simple text entry via tilting) for some examples of accelerometer-based tilt interfaces. Jang and Park [2004] discuss gesture recognition with mobile devices augmented with accelerometers. A number of mobile phones (such as the Nokia 3220 and the Samsung SCH-S310) already feature accelerometer devices.

V.4.3.2 Gyroscopes, Magnetometers and other Inertial Devices

As noted, gyroscopes (which measure angular rates) and magnetometers (which measure magnetic field strength, and can be used as electronic compasses) are also usable inertial sensors, but are more expensive and more difficult to work with than accelerometer devices. MEMS gyroscopes, which use tiny vibrating elements to detect motion, suffer from rapid drift and are sensitive to temperature changes. Magnetometers give estimates of orientation with respect to the Earth's magnetic field, but are disturbed by metallic objects in the vicinity – severely limiting their use indoors. Fusing the inputs from two or more sensor types, can, however, be used to gain much greater tracking accuracy, and to separate orientation changes from translational movements (see Luinge [2002] or Kim [2004] for more details).[1]

V.4.3.3 Advantages and Disadvantages of the Inertial Approach

Inertial sensing provides rich sensory input augmented cheaply and with very little support hardware. The input is direct and natural, in the sense that the device itself is simply moved around to effect changes in state, and sampling rates are high enough that fluid, real-time control can be achieved. Inertial sensing platforms also suitable for one-handed use, a major advantage over other input device types. However, they can lack precision, can induce light reflection issues when used with screen based displays, and have no inherent feedback – unlike a button array or even a screen, which constrains motion to definite surface. All feedback about the sensed state either comes from direct proprioceptive/visual information about the position and orientation of the device or from the artificial feedback generated by the system. The design of appropriate feedback is therefore critical in such applications.

Inertial sensors are also subject to various noise sources (e.g. tremor, gait, vehicular motion, etc.) which have to be accounted for. The sensing of these states can provide rich context data, but this requires significant modelling and processing to be of use. The inertial approach is also limited to the ways in which the physical device itself can be manipulated – a cumbersome device will lead to poor interaction. Other input devices, such as touch screens, which use two handed control, suffer less from the effects of the

[1] The "residue" after such fusion operations is an interesting source for additional context information, such as the presence of metallic objects.

physical design of the device.

V.4.4 Probabilistic Language Modelling

For any input system a model must be constructed of possible user goals; the more effective this model is in reducing the redundancy of the system, the greater communication bandwidth can be achieved. For text, where the canonical representation is a sequence of symbols (the alphabet and other assorted characters), there exists a great deal of redundancy. "Standard" English has an estimated entropy somewhere between $\sim$0.8 and $\sim$1.5 bits per character (Teahan and Cleary [1996], Shannon [1951], Cover and King [1978]).[2] Brown et al. [1992] suggest a maximum upper bound of 1.75 bits per character (modelling all punctuation and capitalisation), computed for the Brown English corpus. This is much less than a straightforward coding of the symbols, which requires 6–8 bits per character. Greater compression can be achieved in situations where the language is more restricted or regular, such as in technical or legal contexts. Efficient language coding systems exploit these redundancies to minimise the necessary bandwidth for storing or communicating text – this is the basis of efficient text entry. The important issue is to make use of these models in an interaction; a model which has very good compression properties is of little use if it cannot be reliably controlled by an interactor.

In this work, text coding models which can estimate the p.d.f. of possible text sequences given an input are of particular relevance. These probabilities can be fed back into the dynamical model to create the dynamic handling qualities outlined in the previous sections.

V.4.5 Transition from Novice to Expert: Learnability

A further issue in the design of a text entry system is the *learnability* of the system: how quickly a user can reach a particular bit-rate. This depends on how well the user can model the behaviour of the system in response to inputs, which in turn depends on the long term stability of the system, and on how well the control inputs and feedback of the system are matched to the user in the context of operation. A system which requires input as controlled and nuanced as required to play the violin, for example, will be unusable without years of training. Equally, a system with a sophisticated model which depends strongly on changing context variables can become so (apparently) complex that learning the response of the system to even trivial changes of inputs is very challenging.

Adaptivity is a major issue in the learnability of a system. Allowing a model to adapt to frequently used phrases, for example, can greatly increase the representative power of the system. However, if in doing so it significantly disturbs the model the user has made of the system, the effect is likely to be detrimental despite the improvement in the language model.

A key aim of the modulated dynamics approach is to allow the construction of systems which are locally smooth – from the perspective of the user – but can still incorporate sophisticated models for improving the bandwidth of communication. These smooth changes in behaviour should maintain the originally learned behaviour while gently introducing opportunities for richer control and correspondingly higher communication rates.

[2] Cutting edge compression engines like PAQ8H (`http://www.cs.fit.edu/~mmahoney/compression/`, retrieved June 10th 2006) achieve these ratios in practice – with $\sim$0.9 bits/character for the CIA world factbook and $\sim$1.4 bit/character for the Calgary test corpus.

V.4.6 BASIC HEX ENTRY MODEL

The Hex system aims to implement a mobile text entry system based on inertial sensing, which demonstrates the use of the modulated dynamics approach to increase usability and communication rates. The aim is that Hex should support the flexibility and power of gesture recognition systems, without require rote memorisation of abstract sequences; the intuitiveness of the spatial pointing technique can be preserved. By choosing the dynamic system carefully, the interactor should be able to "bootstrap" from tightly-coupled control behaviour to faster, semi-open loop control. Hex constructs a virtual world that supports this transition by providing pseudo-physical indicators of the structure of the inference mechanism.

V.4.6.1 IMPLEMENTATION PLATFORM

The Hex system is implemented on an PocketPC device, with an external tri-axis accelerometer P^3C from XSens devices, which measures linear accelerations from -2g–2g. The peripheral is built around ADXL202 accelerometers. Tilt can be obtained from the effect of gravity on the sensor. The setup is shown in Figure V.14. A version for desktop PC's controlled via the mouse has also been implemented.

FIGURE V.14: PocketPC augmented with XSens miniature linear accelerometer.

V.4.6.2 HEXAGONAL TESSELLATION MODEL

In the Hex system, text is produced as a sequence of characters. Each character is encoded as a pair of transitions in a hexagonal grid; first a transition into a character group, and then a transition into the specific character. The hexagonal tessellation is space-filling (it has no "wasted" regions of the space), makes it possible to code characters in only two motions and requires only six basic directions of movement, which is quite achievable with accelerometer-based tilt input. Other tessellations either waste display space, require implausibly accurate tilt control or need longer sequences for character encoding; hexagons are good compromise between these criteria.

The two transition model provides thirty-six possible characters. Transitions back from a group backwards to the original state have no effect (a sort of limited undo), removing six possible characters. The final arrangement looks like Figure V.16. Alphabetic characters, backspace and space are shown. Additional punctuation and numbers can be added by introducing "shifted" three-transition sequences, but are not present in the Hex prototype. Transitions can be quickly computed by finding the nearest (Euclidean distance) point to a cursor on a hexagonally-arranged grid.[3] The accelerometer input is used to control the movement of the cursor through the grid (see V.4.7.2).

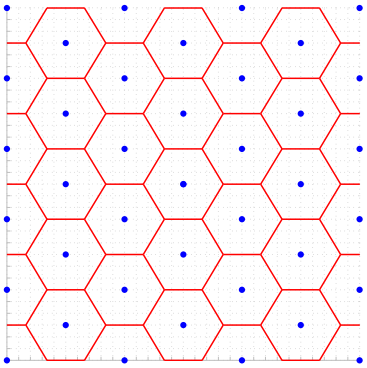


FIGURE V.15: The hexagonal grid is formed by taking the Voronoi tessellation of the hexagonally-arranged points (heavy black points). Transitions detection is simply a matter of observing changes in the nearest hexagon centre point.

The boundaries of the Voronoi tessellation of these points are hexagonal (see Figure V.15. Some example words and the trajectories which would generate them are shown in Figure V.18. A series of images from Hex in action is shown in Figure V.17.[4]

Every sequence of characters in Hex corresponds to a set of two-dimensional trajectories. It should be noted that these trajectories are *stable*, in contrast to adaptive layout systems like Dasher (Ward et al. [2000]).

[3]Such a grid can be obtained from a regular square grid, having vertices at $(x,y) \quad x,y \in Z$ by keeping only points where $(x \equiv 0 \bmod 10 \wedge y \equiv 0 \bmod 6) \vee (x \equiv 5 \bmod 10 \wedge y \equiv 3 \bmod 6)$.

[4]Hex features a spherically distorted display to maximise the visible area for fine control, while still presenting enough information to plan trajectories. This also leaves unused room at the corners, which can be used for other status information.

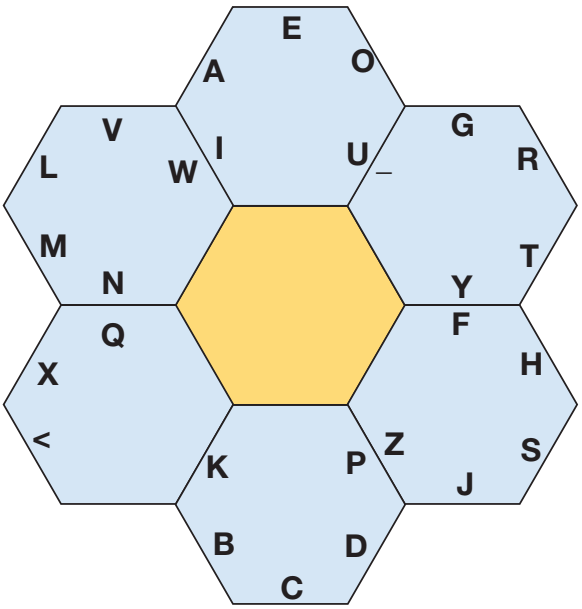


FIGURE V.16: The default layout of letters in Hex.

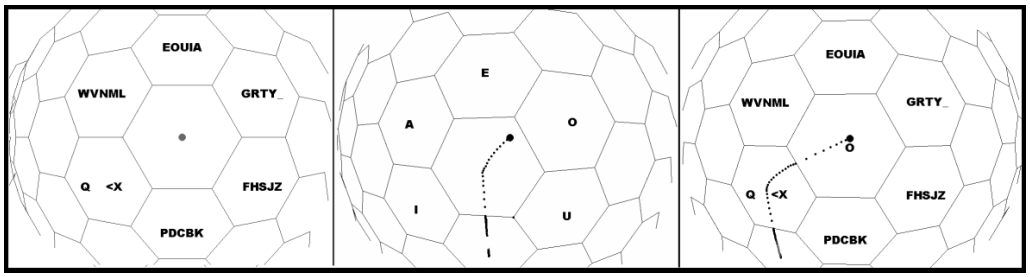


FIGURE V.17: A series of images from Hex in action. "O" is being entered. The cursor history is shown as a series of black dots.

V.4.7 BASIC DESCRIPTION OF THE HEX DYNAMICS

Hex involves the simulation of a dynamic system. The Hex model can be described as follows:

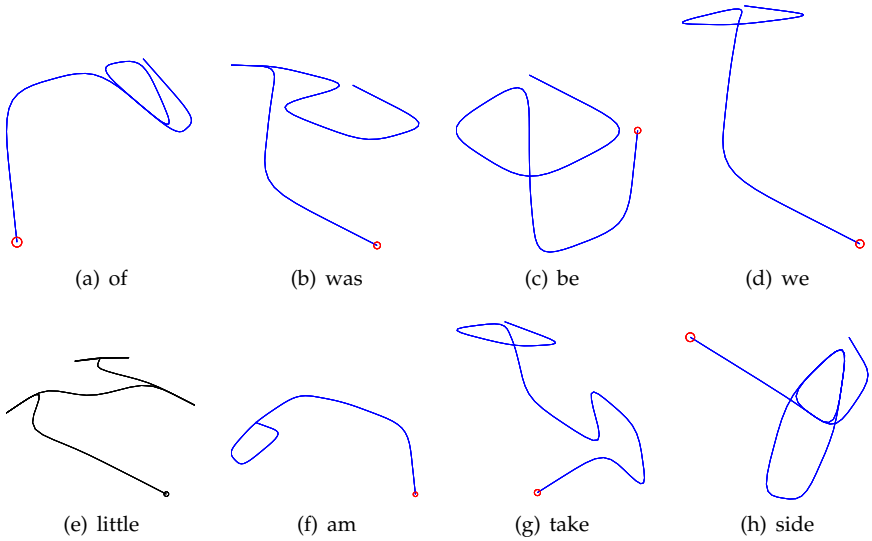
$$\dot{s} = h(u, s, l), \tag{V.1}$$

where h transforms the control inputs, depending on the current state s and the current language model state l

$$h(u, s, l) = u + k(s, l), \tag{V.2}$$

k represents the effect of the language model. u represents the control input.

$$u = \omega \left(\begin{bmatrix} \theta \\ \phi \end{bmatrix} \right), \tag{V.3}$$



(a) of (b) was (c) be (d) we

(e) little (f) am (g) take (h) side

FIGURE V.18: The trajectories corresponding to words using the layout given in Figure V.16. These are created by fitting a cubic spline through the medians of the transition edges which would produce the word.

where ω is a transfer function applied to the tilt angle estimate (θ, ϕ). The following sections outline the details of this process.

V.4.7.1 TILT TO VELOCITY

The cursor velocity is controlled by the angle of tilt of the device, as measured by the accelerometers. Assuming that the device is being tilted without significant external (non-gravitational) linear acceleration, the orientation ϕ, θ of the device can be found from the measured x, y, z accelerations via:

$$\phi = \tan^{-1}\left(\frac{y}{x}\right), \tag{V.4}$$

$$\theta = \cos^{-1}\left(\frac{z}{\sqrt{x^2+y^2+z^2}}\right) \tag{V.5}$$

V.4.7.2 STABILITY OF CONTROL: DEAD-ZONES AND TRANSFER FUNCTIONS

The velocity control is mapped so as to have zero at a user-calibrated point (auto-calibrated at start), reaching a maximum at $\pm\frac{\pi}{2}$. Practically, ranges of about $\pm\frac{\pi}{4}$ are achievable with wrist movements (variability of wrist movements in tilt-based interface are described in more detail in Crossan and Murray-Smith [2004]). To transform this to a more comfortable range of movement, a nonlinear transfer function (shown in Figure V.19) is applied. This combines a dead zone (see Jagacinski and Flach [2003]), which acts to reduce small oscillatory deviations near the zero point and slow drift effects, with a $\tan^{-1}$ response which increases sensitivity in the comfortable range of control.

The controller also incorporates mode switching behaviour. While the control input maintains a value inside the velocity dead zone, the system acts as a position controller, and forces from the probability model (see next section) are not applied. This mode

switching behaviour allows stable small-scale adjustments without restricting the total range of movement. Blended-mode controllers are found in advanced aircraft control systems, where very different areas of the flight envelope have to be controlled with the same input device. Shanks et al. [1996], for example, describes the use of blended control modes (vertical rate to pitch rate, based on airspeed) in a proposed flight management system for the VAAC Harrier aircraft, to deal with switches from jet-borne to wing-borne flight. Pilots control the most relevant variable across the aircraft flight envelope.

V.4.7.3 Transfer Function

$$u_1 = \begin{cases} 0 & |\theta| < \alpha \\ \tan^{-1}(\beta\theta) & |\theta| < \alpha \end{cases} \tag{V.6}$$

where α gives the dead-zone width and β the sensitivity of the control. For u_2, substitute ϕ for θ. Note that when $u_1 = 0 \wedge u_2 = 0$, the entire control mode switches to position control, and the total Hex dynamics can be written as:

$$x = x_e + v, \tag{V.7}$$

where x_e is the position where the dead-zone is entered, and

$$v_1 = \upsilon \tan^{-1}(\psi\theta), \tag{V.8}$$

(and similarly for v_2) with ψ, υ defining the response within the dead-zone.

This particular arrangement is effective, but more complex transfer functions which take into account the full variation of wrist movements available could be applied (for example, separately treating the θ and ϕ dimensions to get a fully two-dimensional transfer surface). Hinckley et al. [2000], for example, found that a circular dead zone was significantly more effective than a square one in mobile accelerometer interactions.

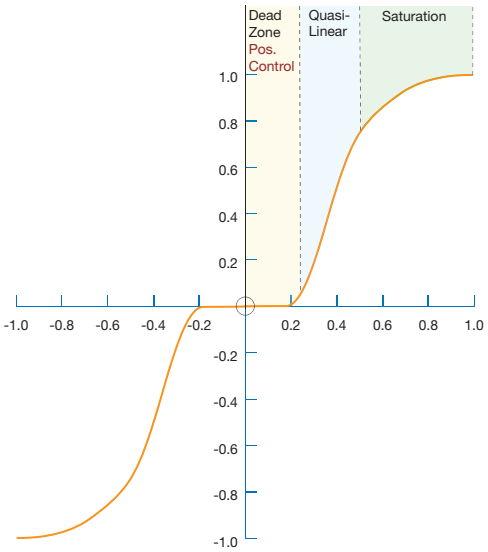


Figure V.19: The transfer function applied to the control input in Hex.

V.4.8 PROBABILITY MODEL

To maximise the potential input rate, Hex employs a probabilistic language model to infer the probability distribution of new characters given a prefix. This model is used to alter the handling qualities (the response of the system) of the system so that character sequences which are more likely are easier to perform.

Importantly, the dynamics have *no effect* on the shape of the optimal gesture for a particular character or character sequence; they only affect trajectories which deviate from these paths, the effect being stronger as deviation increases. This maintains the learnability of the system, while smoothly providing the increased communication benefits of an probabilistic model. They also offer the potential for rich feedback as to the state of the inference model, with respect to the user's low-level (kinaesthetic) goals as well as higher level cognitive goals. Feedback can be produced describing the varying gradient of the surface as the cursor moves over it; this informs the user as to how the language model is distorting the dynamics and what effect should be expected as a result. It is hypothesised that this could facilitate faster development models of the system behaviour in the user's mind.

The augmented dynamics in Hex can be visualised as a time-varying vector field on the plane, as shown in Figure V.21. This can be simplified to a simple surface for most of the calculations (but see V.4.8.1). The movement of the cursor can be imagined as a ball bearing rolling over this continually warping surface as the device is tilted around (see Figure V.20), although the tilt dynamics in Hex are first-order rather than a true second-order tilting model.

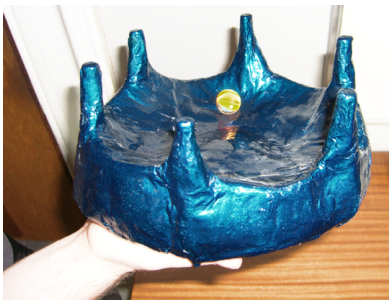


FIGURE V.20: A real physical model showing the design of Hex; rolling a marble (the cursor) across the undulating surface defined by the language model.

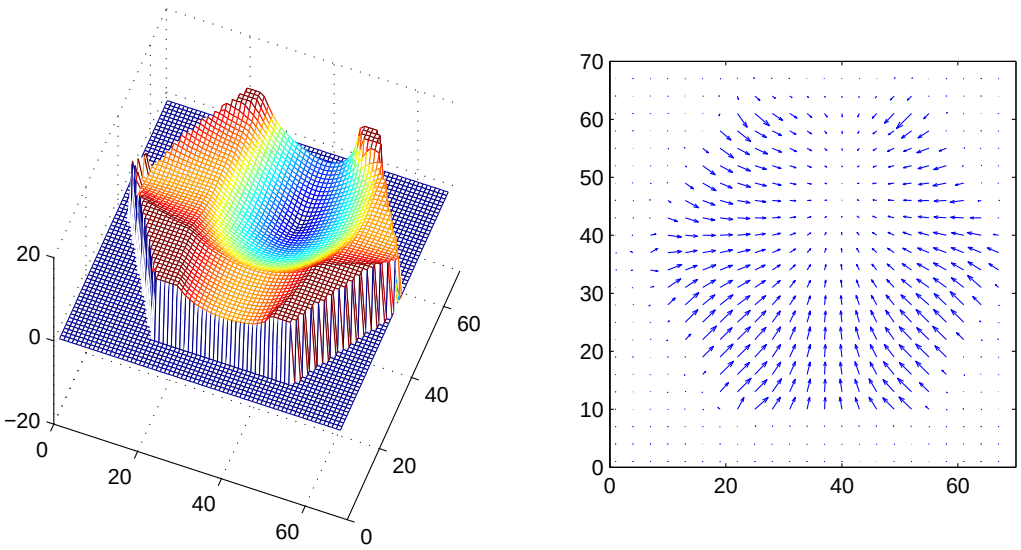


FIGURE V.21: A vector field, illustrating the virtual forces applied to the cursor in Hex

The effect of the dynamics is shown in Figures V.22,V.23, and V.24 where sample words have been performed multiple times. The control inputs and the output trajectories (as observed by the user, and used for the character entry) are shown, both in spatial and time series form. The regularising effect of the dynamic system is quite visible.

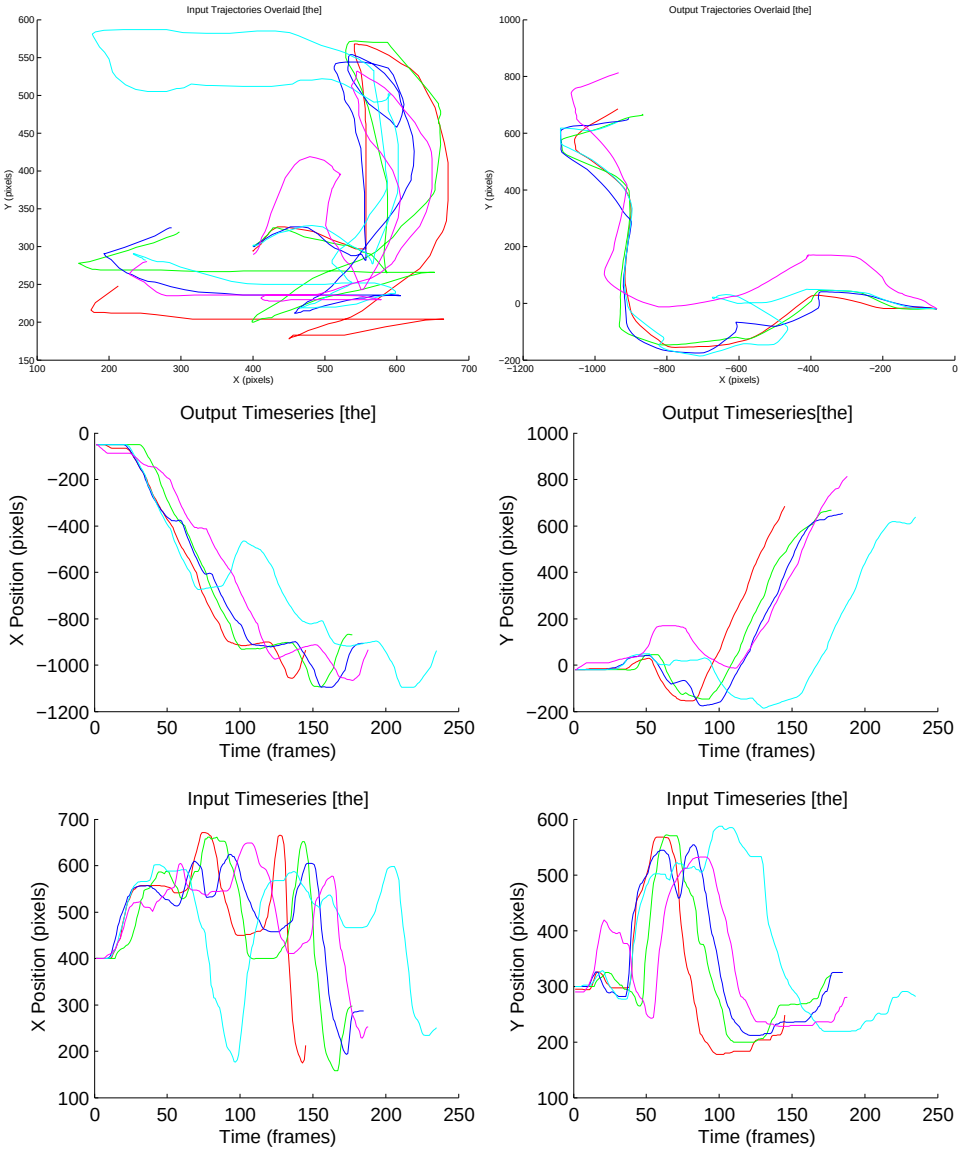


FIGURE V.22: Input and output trajectories for the word "the". Overlaid spatial plots are shown in the top two images; the lower quadrants show the time series.

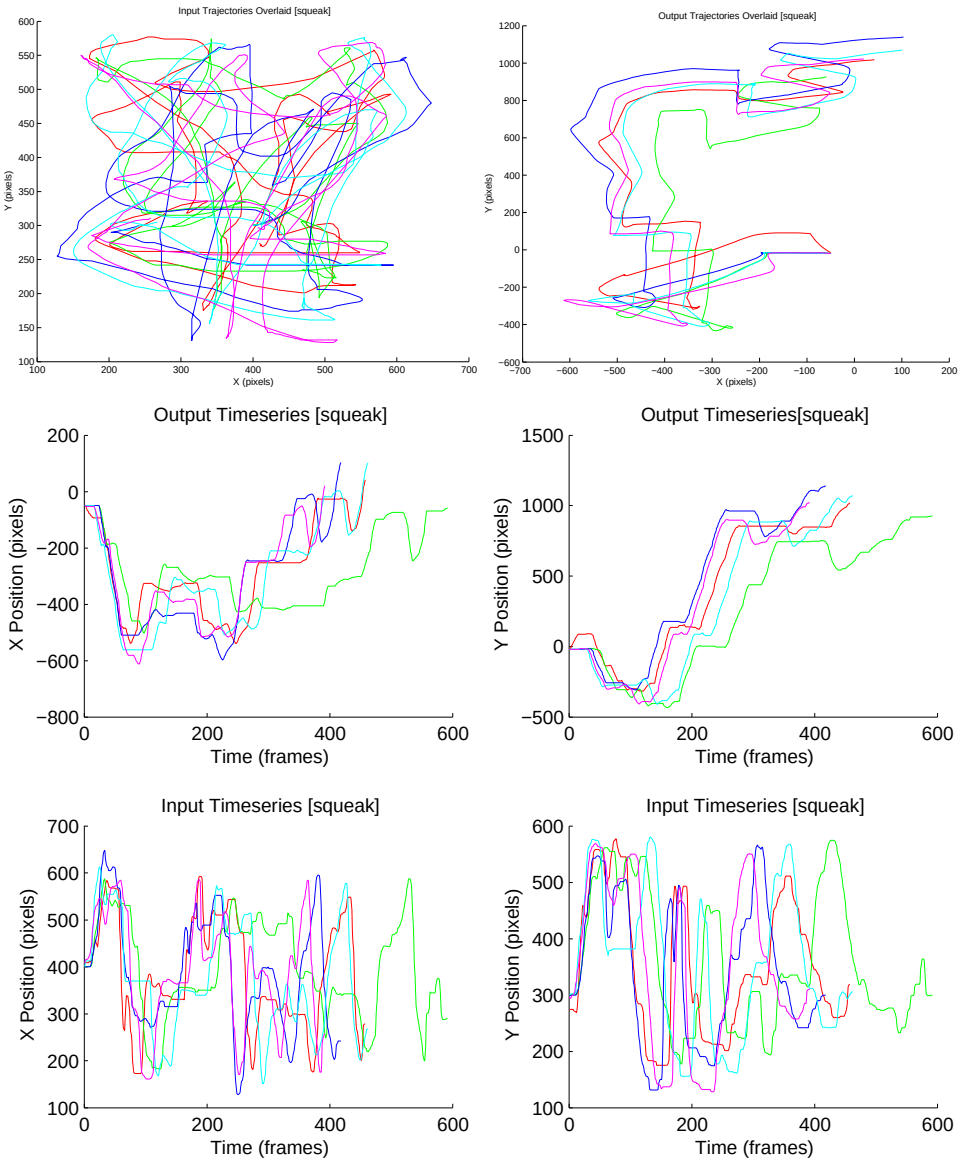


FIGURE V.23: Input and output trajectories for the word "squeak". Overlaid spatial plots are shown in the top two images; the lower quadrants show the time series. This is significantly less likely than "the" (Figure V.22), so the signal complexity is correspondingly higher.

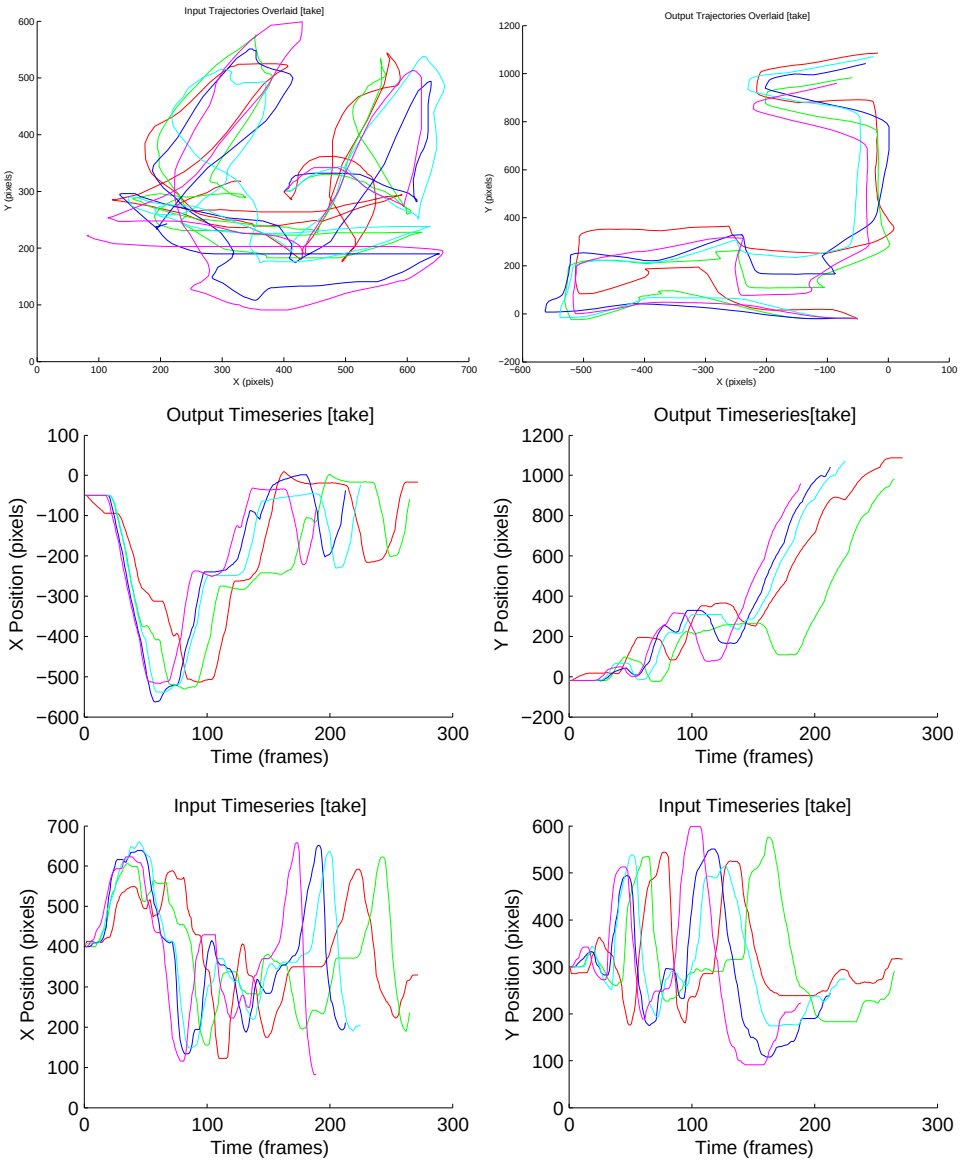


FIGURE V.24: Input and output trajectories for the word "take". Overlaid spatial plots are shown in the top two images; the lower quadrants show the time series. Compare with the ideal form shown in Figure V.18

V.4.8.1 A PRIORI MODEL: GEOMETRIC CONSIDERATIONS

The hexagonal model provides a clean way of dividing the space up into sequences of gestures which can be concatenated and coarticulated into a smooth trajectory (see the example words in Figure V.18). However, it is apparent that some areas of the space are more meaningful than others. Consider a transition through a vertex, as shown in Figure V.25. This provides no information as to which of the two alternatives was desired.

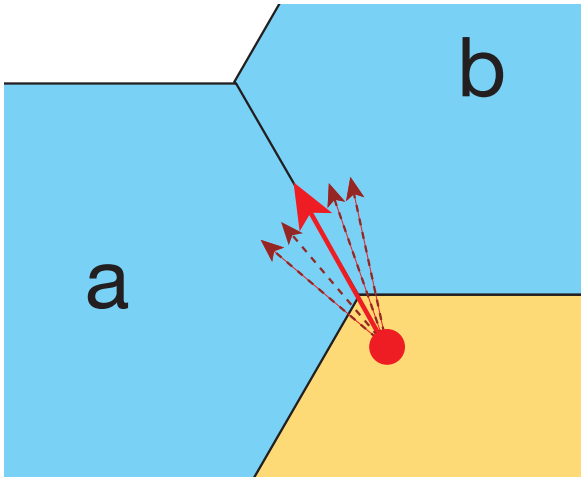


FIGURE V.25: Transitioning *through* a vertex is ambiguous. Very slight changes in initial position or heading lead to distinct jumps in state. Therefore, the physical model prohibits transitions in this region.

One way of dealing with this is to keep the full distribution of potential values, and decode the most likely sequence at some meaningful end point (the end of a word, for example, as in the Shark (Kristensson and Zhai [2004]) system). Section V.4.13.2 discusses this in more detail. In the Hex model, however, a transition pair always changes the state of the system by concatenating a new character. To mitigate the ambiguous decision problem, the dynamics are altered to prohibit moving through the vertex. Virtual "forces" are applied the cursor, repelling it from the vertices. It should be noted that these forces, unlike a true surface, always oppose the direction of the travel of the cursor; in this sense they are more like a resistive friction model.

V.4.8.2 LANGUAGE MODEL

The implemented language model is a modified partial-predictive-match (PPM) model (Bell et al. [1984] Cleary et al. [1995]), which comes close to the maximum possible compression for English (Teahan and Cleary [1996]). A tree of probabilities is stored, giving $p(\text{char}|\text{prefix})$. In the Hex implementation the prefix is variable length, and runs from the start of the word, terminating at the first non-letter character, as opposed to the fixed length model in the original PPM algorithm. A section of an example tree is given in Figure V.26. For testing, the model was trained on the Project Gutenberg corpus.[5]

The model is stored as a tree, which is pruned so that nodes with probability below a certain (very low) threshold are cut off. This reduces the size of the tree sufficiently that

[5] *http://www.gutenberg.org/*, archive as of March 2003.

it can be used without excessive memory consumption. For the PocketPC version, an even more compacted structure is used which codes the entire corpus tree into 1.4Mb along with precomputed autocompletes for every prefix. This structure has much reduced probability resolution (4 bits with logarithmic spacing), but this is quite sufficient for the interaction as it stands.

For the case when the transitions are into groups of letters (rather than into individual letters), the probability of a transition into a group is just:

$$p(\text{group}|\text{prefix}) = \sum_{i=1}^{5} p(\text{letter}_i|\text{prefix}), \tag{V.9}$$

where letter_i is the i'th letter in the group.

Hex's language model is non-adaptive. In a production system, adaption would almost certainly be necessary for effective use (e.g. for proper names etc.). Making the language model described here adaptive is straightforward (although current prototypes pre-compute the probability tree offline). The effects on learnability must, of course, be borne in mind when adaptive language models are introduced in text entry systems – making a system sufficiently flexible to give real communicative advantage without disrupting learned patterns is a difficult research problem.

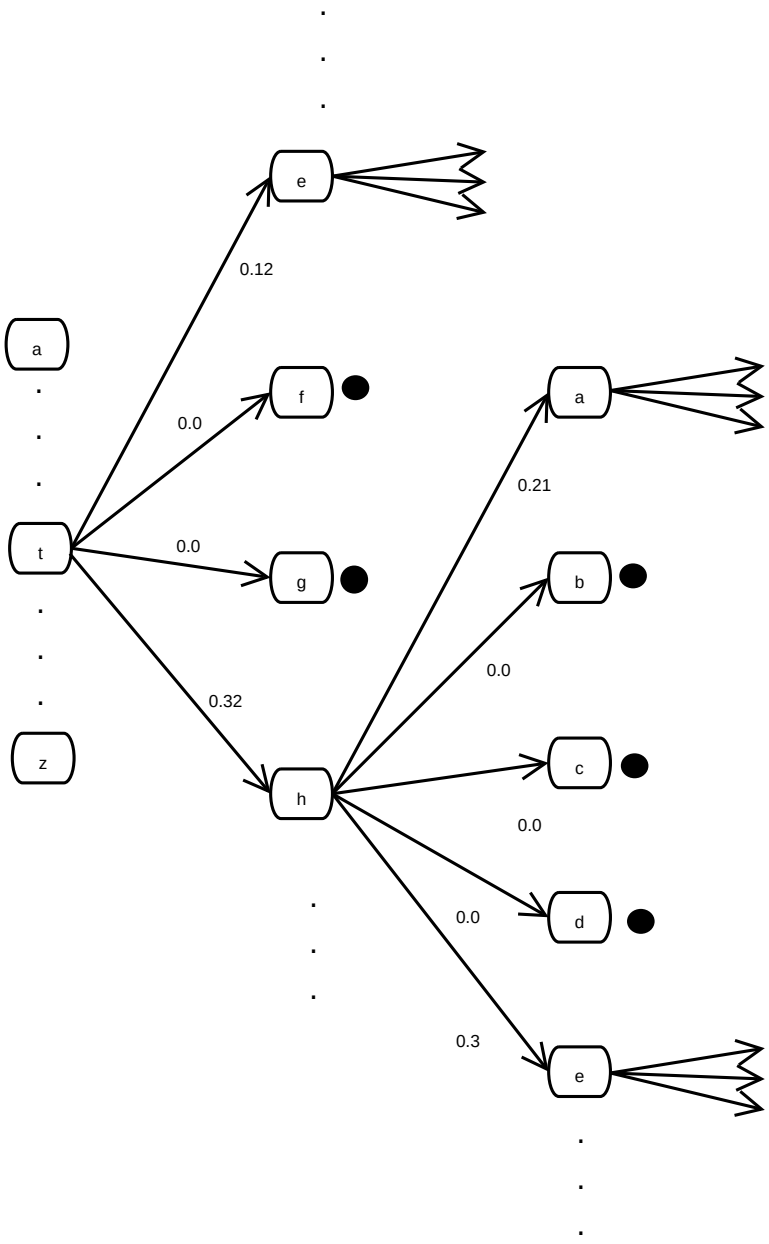


FIGURE V.26: A section of the Hex language model (probabilities are from a short test training set, not from the true corpus). The probability of jumping to a new character given a prefix is shown along the edges. Black dots indicate pruned regions of the tree.

The language model is used to alter the dynamics so as to create virtual valleys in the surface, making it easier to roll the cursor towards likely outcomes, and harder to move towards unlikely ones. A mixture of squared exponential functions used to form to the

surface, each centred at the vertices of the transition boundary. This causes the surface to be smoothly raised up towards unlikely outcomes. Some examples are shown in Figure V.27.

V.4.8.3 LANDSCAPE EQUATIONS

The surface dynamics can be summarised as follows:

$$k(x,l) = \int v(x) + m(x,l)dt, \tag{V.10}$$

where m is the surface defined by the language model, and v the vertice model.

$$v(x) = \frac{\dot{x}}{|\dot{x}|} \sum_{i=1}^{6} \gamma e^{\left(\frac{d(x,i)^2}{-\psi^2}\right)}, \tag{V.11}$$

where γ and ψ specify the strength and width of the vertex repulsion, and $d(x,i)$ is the Euclidean distance from x to the vertex i in the current hexagon. The first term forces the direction to be normal to the current velocity.

$$m(x_1,l) = \left(\frac{\partial q(x,l)}{\partial x_1}\right), \tag{V.12}$$

and similarly for x_2.

$q(x,l)$ describes the height of the total landscape surface:

$$q(x,l) = \sum_{i=1}^{6} \omega_i e^{\left(\frac{d_i^2}{-\zeta^2}\right)}, \tag{V.13}$$

where ζ is the width of the language model landscape deformation (usually with $\zeta >> \psi$), and ω_i specifies the strength of effect for one vertex:

$$\omega_i = \tau \frac{p_l + p_r}{2}, \tag{V.14}$$

where τ is a scaling factor and p_l, p_r give the probabilities of transition across the edge to the left and to the right of vertex i, respectively.

V.4.8.4 MOVEMENT DYNAMICS AS INDICATORS OF INTENTION

The interface is a dynamic system, and the trajectory by which a user arrived at a particular point obviously contains information as to their future intention. In Hex, the velocity and acceleration of the cursor are estimated, and used to alter the landscape accordingly.[6] The complete probability model in Hex for a transition into t looks like:

$$p(t) = p(t|r,l)p(t|v,a), \tag{V.15}$$

[6]This is, of course, a very simple use of the movement dynamics. More complex models might reduce sensitivity when the trajectory is highly irregular and suggests the user is ineffective at controlling the device. The information matching ideas of Section II.7.2.1 suggest that if the input bit rate greatly exceeds the potential communicable information, imperfect interaction is occurring. A reduction in sensitivity forces smoothness in the system-side, hopefully stimulating a corresponding increase in smoothness on the part of the user. In brain-computer interfaces (for which Hex has already been adapted and is being employed Blankertz et al. [2006]) there is the even more interesting possibility of using the *error potential* (Blankertz et al. [2002], Schalk et al. [2000], Ferrez and Millán [2005]) – which is an indicator of user awareness of error – to dampen the dynamics. Physiological evidence for emotional state (e.g. as discussed in Picard et al. [2001]) can also be used to dampen or enhance the movement dynamics.

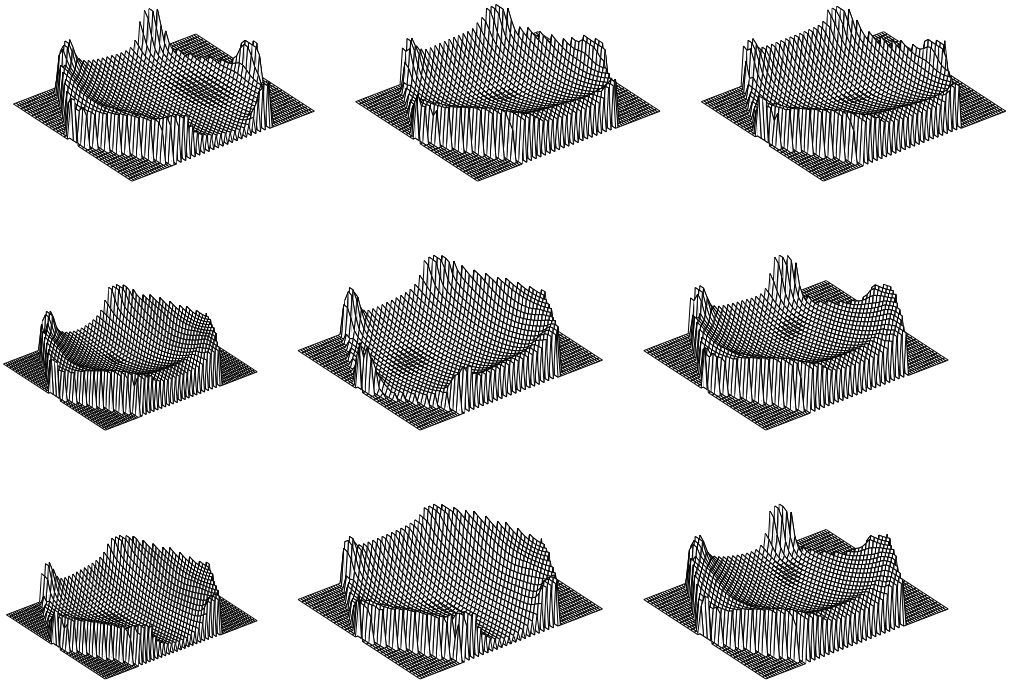


FIGURE V.27: Some example virtual landscapes in Hex, showing how the probability affects the virtual surface. These are snapshots of the landscape during an interaction.

where r is the current prefix, l the language model, v the current velocity vector and a the current acceleration vector.[7]

The probabilities $p(t|v)$ and $p(t|a)$ are given by:

$$p = s + (1-s)\frac{\cos\theta + 1}{2}, \tag{V.16}$$

where θ is the angle between the velocity/acceleration and the centre of the hexagon the transition leads to. s gives the weighting of the dynamics model; $s = 1$ has no effect on the probability, $s = 0$ has maximum effect. This implies that the probability of crossing a transition is lowest when movement is normal to it, and most likely when heading straight for it. In the Hex model, these two probabilities are simply multiplied to obtain the joint probability $p(t|v,a)$ (possibly with different s's). Although these variables are clearly not independent, this approximation suffices.

V.4.8.5 SIMPLE AUDIO FEEDBACK Hex supports simple audio feedback, producing short indicator sounds when hexagonal boundaries are crossed, a complete letter is entered and when a word is completed. Each individual hexagon boundary (i.e. each direction) has a unique sound, to give some awareness of the action taken even when the eyes are not available.

[7] This is similar idea to the technique presented in Lank and Saund [2005] for estimating the "level of intentionality" of a pen gesture by analysing its motion, taking into account the velocity and curvature of the gesture, although their formulation is not probabilistic.

V.4.8.6 AUTOCOMPLETE A minor enhancement is the inclusion of autocomplete functionality. The most likely completion for a given prefix is stored alongside the probability tree described in Section V.4.8.2. The current potential autocompletion is displayed on screen. A simple shake detection algorithm detects shake movements in the accelerometer input, and autocompletes the word. Figure V.28 illustrates the shake detection algorithm.

This is a convenient function in the current prototypes, but it is incompatible with the smoothness ideal of the modulated dynamics approach. Better language modelling – and especially *post hoc* interpretation of gestures, rather than character-at-a-time only input – would be a better approach to improving the interaction, and one that is consistent with the principles outlined in this chapter.

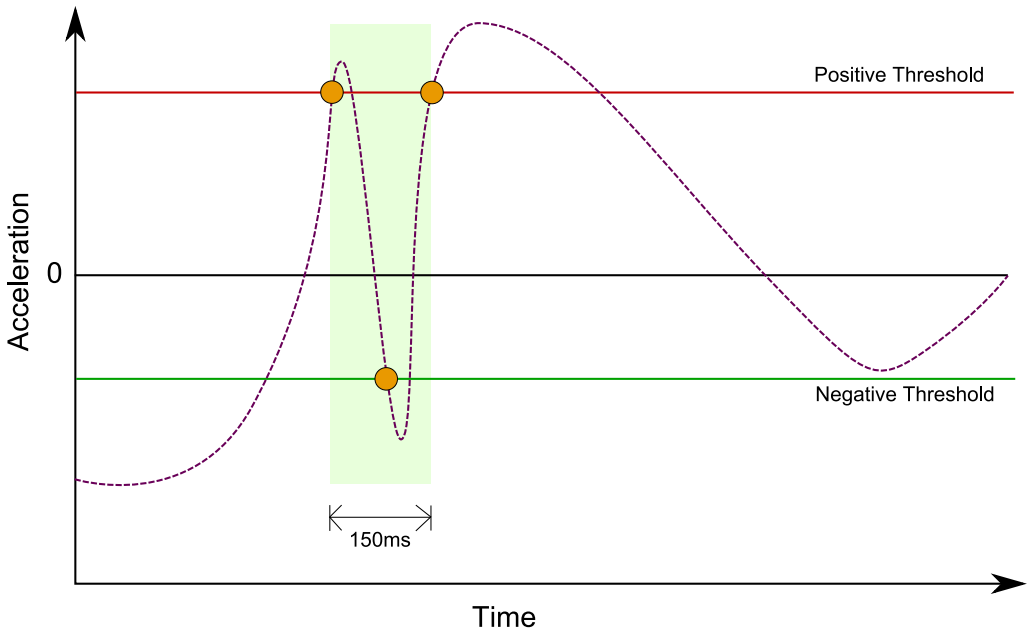


FIGURE V.28: The simple shake detection algorithm used in Accelerometer-Hex. The z-axis signal is shown at the right. The red line indicates the positive shake threshold, the green the negative. If a the series crosses the positive threshold going upwards, the negative going downwards and then the positive going upwards again within 150ms the movement is considered a shake. Crossing the first threshold locks the state of Hex until 150ms has passed, minimising the effect of unrecognised shakes .

V.4.9 MONTE CARLO SAMPLING AND PREDICTIVE DISPLAYS

Although the user perceives directly the effect of actions upon the interface from the motion of the tessellation, higher-level predictive display is absent from the system as described so far. To give the interactor an overview of the inference process (and how it is likely to affect future actions), Hex[8] implements a Monte Carlo sampled display, showing future possible completions of the current word.

[8]Only present in the desktop versions of Hex; implementation issues make this type of display difficult to realise on the current generation of PocketPC devices, particularly with regards to storing sufficiently high-

This display shows the distribution of possible completions. The sampling proceeds as follows:

1. given a current prefix string s_i;
2. randomly select a new character c_n according to the distribution $p(c|s_i)$ from the probability tree;
3. append c to s_i;
4. repeat 1–3 until the word termination character is observed;
5. place the s_i into a hash table, accumulating the observation counts for the strings;
6. reset the string to the original prefix;
7. repeat 1–4 k times;
8. normalise the each observation count by the n to get the estimated word probabilities $p(s_i)$;
9. sort table by observation count;
10. display to v observed strings.

This is done with k in the range of several hundred, with v =2–10. The final words are displayed by calculating the transitions that would be required to get to these completions, and fitting a smooth cubic spline through these edges (exactly as described below in Section V.4.10). This is drawn on screen as a series of points, whose width represents the probability of the word (normalised observation count). The top ranked word is also displayed on screen. Figure V.29 shows an example of this process in action.

resolution probability data for the language model.

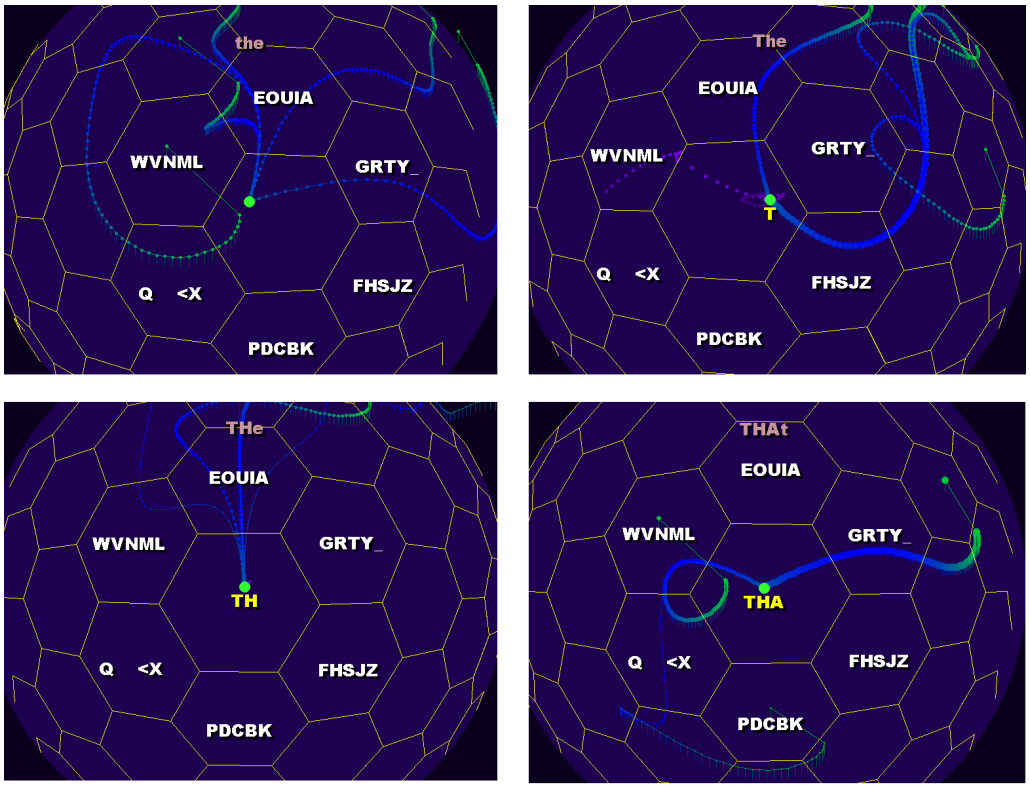


FIGURE V.29: Monte Carlo sampling display on the desktop version of Hex. Sampling is performed over the language model, given the current prefix. Spline paths for each sample word are drawn on the surface; the thickness of the line displays the probability of that path. For clarity, this example shows only two predictions, and the top-ranked predicted word is shown at the top centre.

The effect is a highly dynamic display that shows both likely directions and the variability associated with the current state. When a great number of likely potential alternatives exist (i.e. high entropy distribution) the display is visually noisy, with rapidly flickering "lightning-like" effects. When one or two completions are probable, the display stabilises, with clear paths for the user to follow. This both shows the user the optimal shape for paths, and also allows for very rapid motion towards a goal when the obvious completion is visible. In this way, the feedback can provide some of the performance benefits of an autocomplete function, but without disrupting the flow of communication or causing the interactor to lose direct control.

V.4.10 LAYOUT OPTIMISATION

The layout shown in Figure V.16 is mainly chosen for ease of learning, grouping similar sounding letters together. It is almost certain not to be the most efficient layout. A better layout can be computed by assigning a cost function to the layout and then optimising to find a minimum cost layout.

V.4.10.1 Computing the cost

The word-cost of an entire layout can be calculated by summing over an entire corpus:

$$\sum_{i=0}^{n} c(x_{wi})p(w_i)$$

where c is a cost function, x_{wi} is the trajectory for word i, and $p(w_i)$ is the probability of word i. This approximation ignores any inter-word correlations.

V.4.10.2 Cost and trajectory model

The arrangement shown in Figure V.31 was optimised via simulated annealing (see Kirpatrick et al. [1983]) according to the following model:

- **Trajectory** The optimal trajectory for a word is approximated as a cubic spline fit through the medians of the transition edges that form the word, and the centres of the hexagons that this path passes through (see Figure V.30).

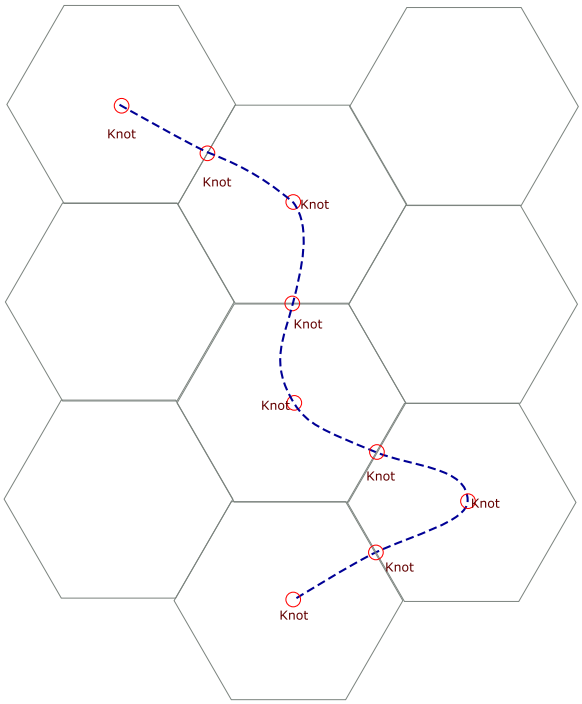


Figure V.30: Cubic spline word trajectory generation. The knots are at the centres of the hexagons and at the medians of the edges the path goes through.

- **Cost** A minimum jerk cost model is applied to these trajectories (a simple model of optimal human movements – see Flash and Hogan [1985]), that is:

$$c(\mathbf{x}) = \alpha \int_0^t \left(\left(\frac{d^3x}{dt^3} \right)^2 + \left(\frac{d^3y}{dt^3} \right)^2 \right) dt \tag{V.17}$$

This is then summed for all corpus words according to the cost equation above. This gives the cost

of the layout for this time step. Any other model of the cost of a trajectory could be substituted instead.

The complete annealing procedure then involved:

- computing the total cost of the layout;
- permutation of letters perturbed by random shuffling, with the number of shuffles determined by the current annealing temperature;
- selection of jump state;
- decay of temperature (according to an exponential schedule).

The results of this approach are not particularly stable, as multiple runs often generate quite different layouts, but with similar final costs. The cost was reduced significantly by the optimisation process, but the effect of these layouts on real human performance is hard to ascertain due to the time taken to learn a new letter layout sufficiently well to produce fluid interaction; the true effects of an optimal layout are only likely to be perceived with sufficient control skill that the minimum jerk model is a reasonable model of expected action. Obtaining a suitable cost function for novice users is a much more difficult proposition; the behaviour of such interactors is normally both more complex and more diverse. The lack of experience leads to almost random variation in some areas, and what control is present is heavily coloured by the short-term previous experiences of that specific user. It is also perhaps beneficial that the optimisation targets the experienced user; a small gain in terminal bit rate will be much more useful to interactor using the system regularly than a similar rate gain at the beginning of the learning curve.

Stabilisation of the optimisation process could be achieved by introducing some prior on the layout of the characters (e.g. one that is close to one familiar to users, such as alphabetic order or one which keeps similar sounding letters together, etc.). This would have a better chance of producing layouts which are easier to compare, and easier for users to learn.

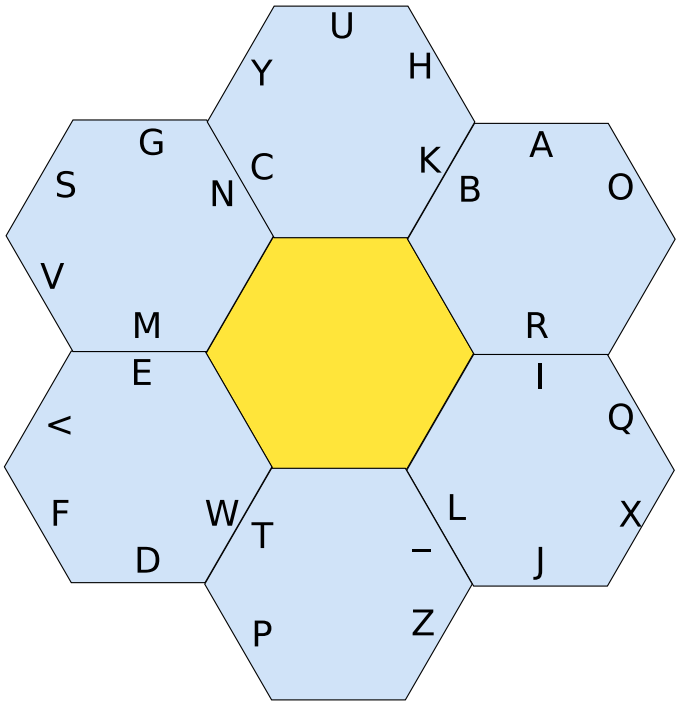


FIGURE V.31: An optimised layout of letters in Hex. This minimises the jerk of word trajectories for the training corpus.

V.4.11 PERFORMANCE AND ANALYSIS

V.4.11.1 HEX AS HIERARCHICAL CONTROL LOOPS

In Section II.10, the user and interface were decomposed into a hierarchy of control loops. Figure V.32 shows how Hex maps onto this structure. The world is sensed via the accelerometers; the landscape model forms the short-time dynamics; and the language model is the inference module.

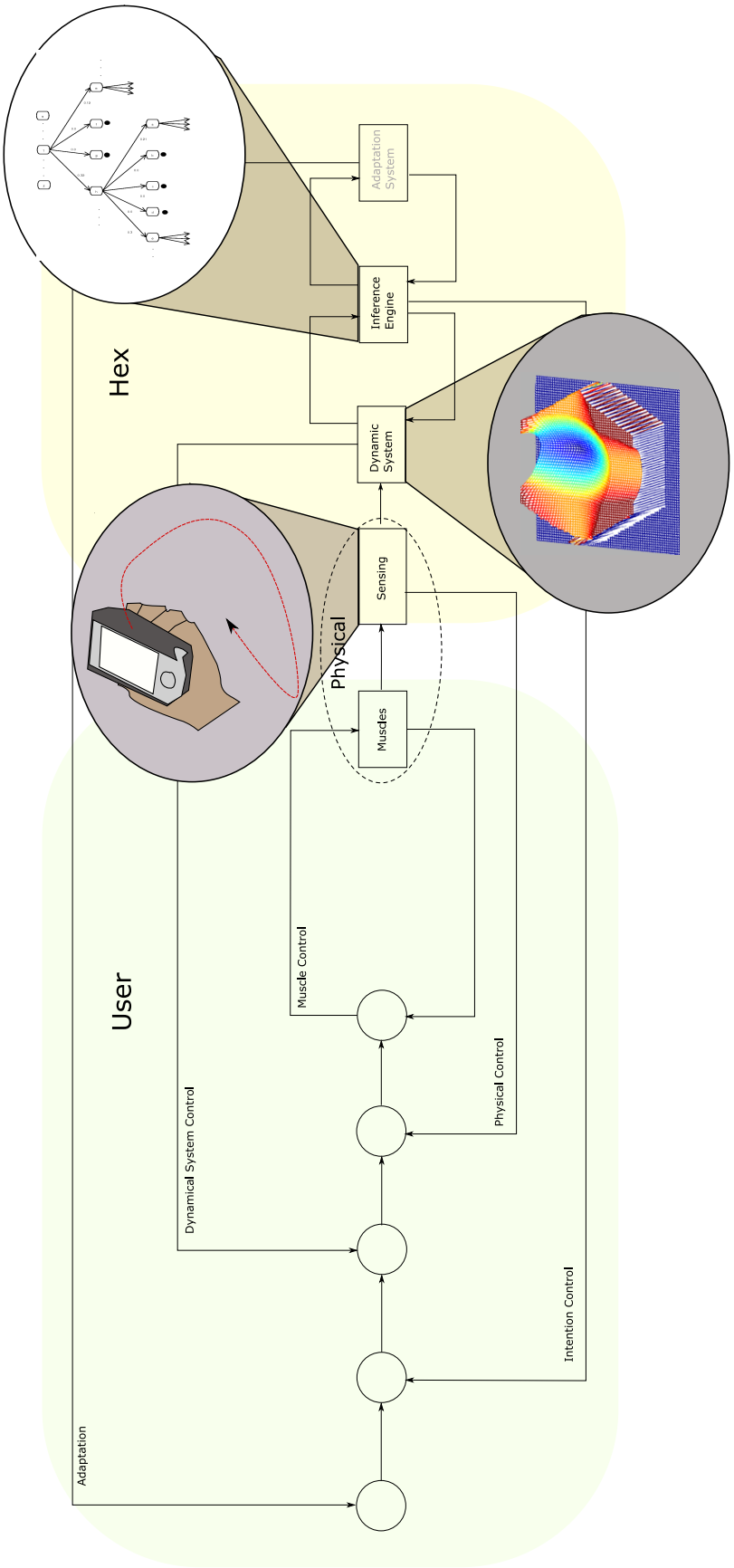


FIGURE V.32: Hex in the form of hierarchical control loops. The world is sensed via accelerometers, which control the landscape dynamic system. This is modulated by the language model, which infers users goals. Hex has no adaptation, but if it did, the adaptation process would modulate the inferential mechanism.

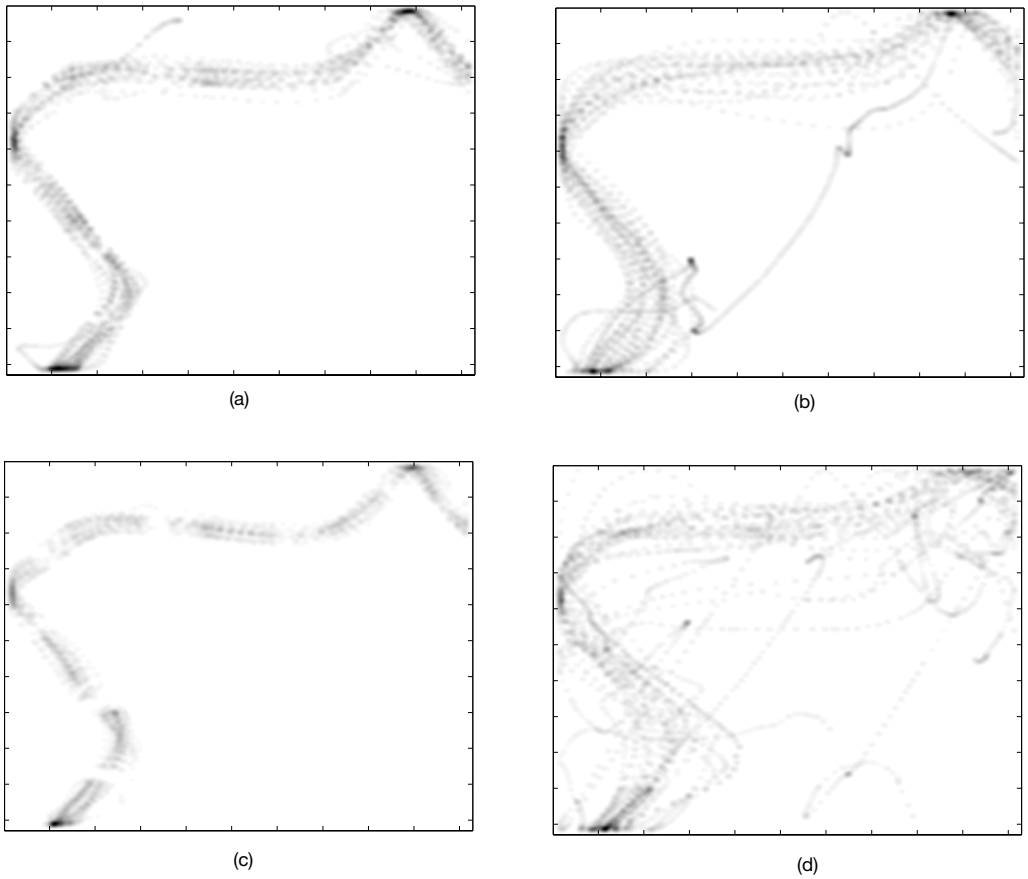


FIGURE V.33: Cloud plots of trajectories in Hex, to produce the word "the". (a) shows ten trajectories with the normal physical model, (b) shows the effect of removing the dynamics model (variance is increased here), (c) shows the effect of doubling the forces strength, creating a tighter but jumpier result, and (d) shows the effect of inverting the language model probabilities, significantly impairing the ability of the user to control the system.

V.4.11.2 THE EFFECT OF DYNAMICS

To show the effect of the dynamics on user performance, Figure V.33 illustrates a series of repeated word trajectories. The sample points from the resulting trajectories are summed onto a regular grid after being convolved with a narrow Gaussian window (the images are effectively rasterized Parzen window estimates of the trajectory density). This gives an impression of the density of the trajectory at each point in space, showing regions where movement was slower, and where different runs were closer together. These plots are shown for four conditions: no forces; normal forces; double strength forces; and inverted forces (i.e. with each letter probability p replaced with $1 - p$).

The forces have a clear constraining effect, producing a tighter performance in the normal case than in the case where the language model is deactivated. Inverting the forces makes the system even worse, making it very challenging to produce probable se-

quences. Doubling the force strengths produces a tighter plot, but the unnatural "stepping" caused by ridges at the transitions between states is visible; this is significantly reduces the controllability of the system.

V.4.11.3 WORD RATES The author achieves rates of between 12 and 17 words[9] per minute with the Hex system (PocketPC version with accelerometer), for freeform text entry. In terms of bit-rate, this is approximately 2.1 bits/s (assuming 1.5 bits per character, five characters per word). This is competitive with techniques such as the T9 system (but without requiring fiddly and expensive buttons), though it is not quite as efficient as some other gestural systems, such as Shark (Kristensson and Zhai [2004]) or Dasher (Ward et al. [2000]). However, it demonstrates that building a system according to the principles laid out here is both feasible, and leads to good results even without significant *ad hoc* alterations.

V.4.12 HEX IN BCI

There is an ongoing project to adapt the ideas in Hex for text entry in EEG brain computer interfaces, in collaboration with the IDA group at Fraunhofer First (see Blankertz et al. [2006]). The BCI environment is well-suited to the techniques described in this thesis. Interaction is asymmetric, with very limited communication rate (4-40 bits/*minute*), but high display bandwidth. Long delays between action and response (of the order of 500ms) are present, and signals are very noisy. Users have difficulty maintaining reliable control under these conditions; unfortunately frustration tends to lead to rapid decreases in performance.

The user can be supported in such conditions with predictive displays, appropriate treatment of uncertainty and dynamic systems which are designed to operate with the frequency, noise and delay properties of the EEG signals. Without this, users have to learn and compensate for the difficulties of EEG interaction, making a challenging task even more difficult. Although the details of the work are outside the scope of this thesis, current results suggest that even the early adapted Hex model is efficient for EEG text entry. Rates of up to 7.6 characters per minute (for German text entry) have been observed, making it one of the fastest EEG BCI text entry systems in the world.

[9] Actual words, not five character groups.

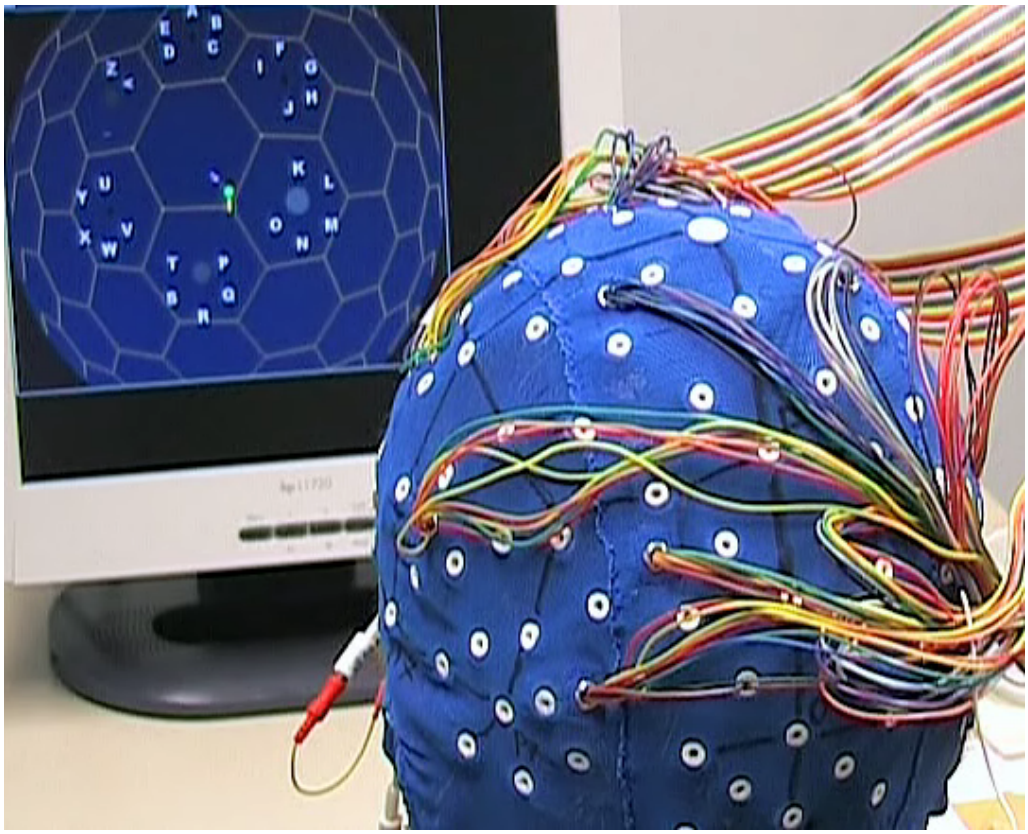


FIGURE V.34: Hex used with EEG brain computer interface control. In collaboration with the BCI group at Fraunhofer First IDA.

V.4.13 ENHANCEMENTS

The current implementations of Hex, as described in this chapter, are complete text entry systems (although lacking some basic editing features). The system could however be extended in a number of ways which could both improve the basic performance of Hex, and make it suitable in a wider array of contexts.

V.4.13.1 IMPROVING THE LANGUAGE MODEL

Introducing longer term partial sentence modelling would be one effective way of improving Hex's performance; simply using more suitable corpora would be another. Some form of grammar modelling running on top of the word prediction would be relatively simple to implement, and would improve the smoothness *between* words. Encoding such a model efficiently (especially for mobile devices) is, however, quite challenging. Text messaging corpora would give a much more realistic set of words to train from, but there is a lack of high-quality corpora of this type; one exception is the Singapore SMS corpus (How and Kan [2005]), which is unfortunately too small for training on its own, with only ∼10,000 messages, and is heavily biased towards Singapore dialect. A weighted combination of this and a another standard English model to smooth out the distribution might be a useful model for mobile text entry models.

V.4.13.2 Word length interpretation

The Hex model could be extended so that the complete distribution over possible words is retained until the completion of a word. This could be computed by Viterbi decoding, treating the interface as a hidden Markov model, where the transition and emission densities are directly observable from the construction of the interface, or using a particle filter to track the distribution of word states. The latter approach offers a clear way of combining the predictive, uncertain displays which were the described in the previous chapter, with sophisticated inference mechanisms.

Foe example, the cursor can be replaced by a "cloud" of cursors sampled from some prior distribution (e.g. Gaussian centred on "true" cursor position). These samples can be subjected to the same dynamics as the original cursor (plus some regular diffusion to avoid a collapse of the distribution). This gives a continuous stream of transition *probabilities*, rather than events. These probabilities can be fed to the decoder to track the distribution of word states. This demonstrates how probabilistic inference can be brought into an interactive system in an straightforward way, without *ad hoc* hacks to introduce "intelligence".

Hex could also be modified to work as a T6-style system, with only a single transition for each character rather than a pair. Given that PPM models can compress English to approximately 2 bits per character and choosing one from six transitions gives ~ 2.585 bits, this should be quite practical.[10]

For this to be truly effective a method for interactively reweighting the postulated alternatives must be provided. Better inference is of limited use if it forces the user to accept only the alternative with highest posterior likelihood, since the cost to the user of correcting the phrase manually is out of proportion to the difference in likelihood of the possible outcomes.

V.4.13.3 Better Visual Feedback

Hex, at the moment, does not directly display the probabilities of different options; only the handling qualities change. Directly displaying this, for example by showing thickened boundaries where probability of transition is low, or even directly display the changing probability surface, might improve the interaction by alerting the user to the likely responses of the system in sufficient time to plan accordingly.

V.4.13.4 Multimodal Feedback

There are numerous avenues for exploring enhanced feedback: friction-type physically-inspired audio or vibrotactile feedback to display the state of the dynamically system; feedback about the state of the language model using the granular synthesis techniques described in the previous chapters (e.g. linking to the Monte Carlo visual feedback); and even simple spoken indicators of change of state (as in Rinott [2004]). Such feedback enhancements could provide an eyes-free text entry system, in which users can gently transition from the visually oriented display to the more unfamiliar audio and vibrotactile displays as experience increases.

Representing the response of the surface via friction sounds (such as in the work presented in Essl and O'Modhrain [2005]) or via realistically modulated rolling sounds (e.g. as in Rath and Rocchesso [2005]) would give the user a sense of the handling qualities in a physically-inspired – and thus familiar – way. These metaphors also translate well into the vibrotactile domain.

[10] Dunlop [2004], for example, describes a "watch-top" text entry system which operates with only five buttons.

V.5 HEX AS AN EXAMPLE OF A GENERAL AUGMENTED DYNAMICS SYSTEM

Hex is intended as an example of the augmented dynamics approach in a real-life problem. The aim of this chapter is to demonstrate that controlling the parameters of the dynamic system with the results of longer term probabilistic inference is a usable technique for the design of interfaces. Combining a standard probabilistic language model with a simple dynamic system leads to a design which is both theoretically elegant and potentially useful. Making use of unusual sensor inputs can be made intuitive with a familiar physical metaphor (here, tilting a surface); this basic metaphor is a reasonable but fairly inefficient way of interacting. However, by taking this model and enriching the dynamics in a smooth fashion, the power of any language model can be brought into the system without losing the original character of the interaction. Adjustments based on basic observations of operator properties can be brought into the dynamic system without disturbing the basic operation of the system; the dead-zone and order-switching features are examples of this. If more detailed operator models are be developed for human control with a particular input device in a specific context, features which take advantage of these properties can be integrated into a Hex-like system in a natural manner.

V.6 CONCLUSIONS

This chapter has demonstrated how long term inference about the goals of a user can be brought into an interaction by changing the way the low-level properties of dynamic systems that make up the interface. The approach separates the design of a fluid, comfortable interface and the design of suitable intelligence to maximise the communication rate, and unifies a number of existing interface design tools. It provides an alternative to the traditional pattern recognition approach to gestural interaction, while retaining the power of the recognition algorithms. The application of the ideas of straightforward; Hex illustrates how the modulated dynamics approach can be applied to a very concrete real-world problem. There is great scope for building interfaces which have both pleasant handling qualities and sufficient intelligence to support interaction in bandwidth limited environments.

CHAPTER VI

ACTIVE SELECTION

Selecting by testing behaviour in a closed-loop.

> *Purposeful behaviour involves the production of consistent results in a world where unpredictable disturbances make such consistency highly unlikely.*
>
> – Marken [2002]

VI.1 SUMMARY

IN this chapter, new methods for selection are described. These operate by identifying control behaviour, and are applicable with many sensing and display technologies. This approach is developed into an agent-based structure, where the agents seek to obtain evidence for intention by provoking responses from the user. This leads to probabilistic interfaces based on damping, resonance and imitation, incorporating operator models of the selection process. Algorithms for such processes are presented, along with implementations and analysis of their behaviour. The selection methods are extended first to continuous spaces, and then to the interactive exploration of high-dimensional spaces, using proxy agents to focus display on aspects of interest.

VI.2 SELECTION PROBLEM

One of the most basic task a user interface can offer is the selection of one or more items from a set. A profusion of methods for doing selection in this most general sense have been developed over the past fifty years. Some of the more common of these are listed in Figure VI.1.

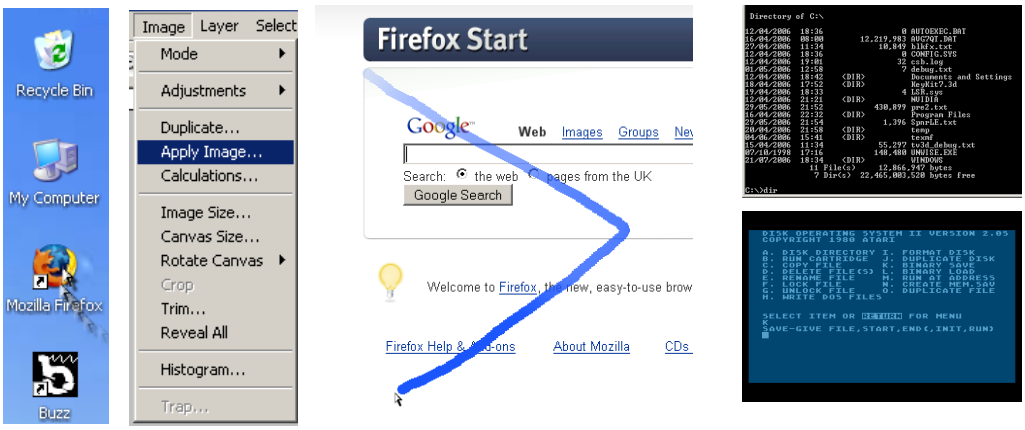


FIGURE VI.1: Some traditional selection mechanisms.

There are various theoretical and practical problems with many of the selection techniques that have been devised. One issue is a lack of a solid framework for defining selection interactions, general enough to be extendable to new modalities or sensing technologies. Many selection techniques on conventional GUI's are based around *ad hoc* physical analogies, such as targeting and pushing a virtual button or dragging a virtual object to be "dropped", or the quite different interactions seen in keyboard-based command-line interfaces, where all of the intentional inference has been performed in hardware. The GUI has brought some of the intention control out of the physical world and into the software world. The design issue is how to best make use of the flexibility that the software interpretation of sensor inputs provides.

The primary limitation on the techniques available is that a user must be able to comprehend the manner in which their actions are translated into system responses. One approach is to base interactions on analogies to physical devices, as in the targeting in a GUI. The spatial-only targeting analogy is, however, a very small subset of the potential interactions which could be designed. Gesture recognition systems attempt to extend the space of possible interaction by taking into account the time history of sensed inputs, and so massively increasing the space of options. The promise of this approach is apparent, but many attempts to build gesture systems have been unsuccessful and with the exception of some handwriting-like recognition interfaces (e.g. Graffiti on Palm devices) and simple mouse gestures, almost none have made it into products.

Taking the view outlined in Chapter II, and viewing the human interaction problem as a control process suggests that the interface should be designed to be controllable. Control-based interactions can offer the flexibility of arbitrary gesture systems with the learnability and intuitiveness of more familiar systems. The control can be quite general in timescale; from semi-open-loop interactions (such as control with concatenated ballistic manoeuvres) to tightly-closed loop interaction. Obtaining the best balance between the memory-dependent but information rich open-loop interactions and the memoryless tightly-coupled interaction is one of the important issues in designing such interfaces

The aim of this chapter is to present a set of selection methods, based around the per-

ceptual control theory view (outlined in Section II.4.2), which are widely applicable and open up a huge range of possible interactions that the flexibility of software interpretation allows. These techniques are applicable to many different sensory inputs (with their corresponding signal properties) and allow for the trade-off between open and closed loop interaction to be manipulated as a design parameter. The techniques follow directly from the assumptions set out in Chapter II.

VI.2.1 SENSING

Traditional approaches are not easily extended to new input mechanisms. It is not clear how a targeting metaphor, say, would work with a pair of foot pedals, or a vibration detector, or an array of pressure sensors. Such sensors are widely available: accelerometers and even MEMS gyroscopes are now found in many mobile devices (e.g. the Nokia 5500 and Samsung SCH-S310 phones); deformable sensors with many degrees of freedom are available for interaction (for example the free curve ShapeTape input device (Grossman et al. [2003]), or smart textile devices (Lorussi et al. [2005], Mazzoldi et al. [2002]); microphones and cameras are also widely deployed in existing products, but their utility for interaction beyond voice recognition or simple static gesture recognition is rarely exploited. They are often particularly useful outside of the desktop environment, in mobile or specialised applications. Inertial sensing platforms, such as the one described in V.4.6.1 or the MESH (Hughes et al. [2004]) are powerful tools for interaction and can be built into mobile devices; these can be used in a vaguely similar way to a mouse (as Hex demonstrates), but this requires careful design of dynamics.

VI.2.2 PATTERN RECOGNITION

One common way of dealing with these sensing modalities is to use some form of pattern recognition, as in gesture, handwriting or speech recognition. But even very sophisticated static methods have disadvantages, as discussed in Section II.11.4, and have often fallen short of the mark in quality of interaction. Creating more advanced recognition algorithms rarely makes huge improvements in the usability of systems, and implementation is often infeasible with the limited computational resources of the devices in which they would be most valuable. The position held in this work is that improvements in algorithms are of limited power unless they are appropriately coupled with more effective methods of interaction. It is postulated that novel interaction strategies – involving the user as a critical part of the system loop – can bring much larger benefits to interactions than incremental adjustments to already heavily optimised algorithms.

VI.2.3 INCORPORATING PROBABILITIES

There are many good reasons for desiring an interaction model which employs probabilistic models to represent uncertainty; Chapter II discusses this in detail. Probability theory gives a solid and well-tested mathematical basis to the inference of state given observable evidence, and a vast array of existing techniques can be brought to bear in probabilistic systems.

A probabilistic model infers the likelihood of items a user may select. The more sophisticated the model, the less information the user needs to input to make selections. This can range from simple, static *a priori* probabilities of particular actions to more complex, dynamic inferences. The design problem is to take a probabilistic model and build a usable interface with it.

Chapter III gave ways of displaying the state of a probabilistic model; Chapter IV extended this to predictive displays. Chapter V described methods for altering the dynam-

ics of the control loop based on statistical inference, improving the quality of interaction by making likely things easy. This chapter describes a completely probabilistic selection mechanism, directly incorporating the results of the inference into the process.

VI.2.4 Continuous Selection; Smooth Goal Space Paths

As described in Section II.7, an interface with constant bandwidth should have a correspondingly smooth path through the space of intentions. If a user is to stay in control of the interaction during a selection process, this smoothness must be preserved (otherwise a step-like action/correction behaviour will occur). An important aspect of the selection techniques described in this chapter is the smooth change in entropy as selection proceeds – interactors remain in control as evidence as gathered and can alter their actions in sufficient time to alter the outcome in a meaningful way.

VI.2.5 Feedback Based Active Selection; Sidestepping The Recognition Problem

Feedback is essential to the control-based interaction approach. By dealing with the conditional probabilities of actions given stimuli, the recognition problem can be neatly decomposed into shorter time segments. The basic principle of a feedback-oriented interaction mechanism is to reduce the possible meaningful actions the user could be performing, limiting them to those related to feedback previously presented in some bounded history. This greatly simplifies the recognition problem and partially forces the interaction to some particular pace (which may not necessarily be constant). Instead of static recognition models, where the aim is to infer $p(g|x_{T-k}...x_T)$, the quantity $p(g|x_{T-k}...x_T, y_{T-k}...y_T)$ is instead computed, where the y's are the previous system outputs.

VI.2.5.1 Identification of Controlled Variables – Disturbances as Tests for Intention

Perceptual control theory, as noted in Section II.4.2, described behaviours of organisms as control of perception. As a consequence of this perspective, identification of variables of interest can be performed by introducing independent disturbances to an ensemble of variables which an interactor may perceive and control, and picking out those which are compatible with the interactor attempting to cancel out the introduced disturbances. Powers and later Marken used this approach to identify which specific attributes of their perception humans were truly controlling.

Marken built a number of computer based demonstrations showing the identification of control of particular elements in a visual display. The original ratio of variance algorithm (Section VI.4.3.1) was presented in Powers [1979] and in Marken [1995] for this identification process. This chapter explores how the identification of control behaviour (and analogous behaviours, such as excitation) can be used as a general selection technique.

VI.2.6 Generalised Identification of Interaction

Feedback is the source of discriminative power in this world-view. An interaction model based around feedback-oriented interaction should be designed so that the probability of an observation being made if the previous feedback was interpreted by an interactor is much higher than the probability of that observation being made were the feedback not present. It is obvious that no test can ever guarantee the interest of an intelligent agent in a variable, but the evidence for the hypothesis that such interest is being shown

can be quantified. The interface robustness can be determined by the quantity of evidence required to assume the belief of a particular intention – which as described in Chapter II should be a function of the utility of the action associated with that intention.

VI.2.7 Asymmetric Channels

These methods rely on the existence of asymmetric communication channels, where the display (feedback) channel has greater bandwidth than the control signal. Such a setup is natural for human communication; the sensory channels of a human have vastly higher bandwidth than those of the muscular control channels (see Figure VI.2). Langolf et al. [1976] estimate the capacities of fingers, wrists and arms at ~ 38 bits/s, ~ 23 bits/s and ~ 10 bits/s, respectively (though Balakrishnan and MacKenzie [1997] suggest that these figures are highly optimistic).

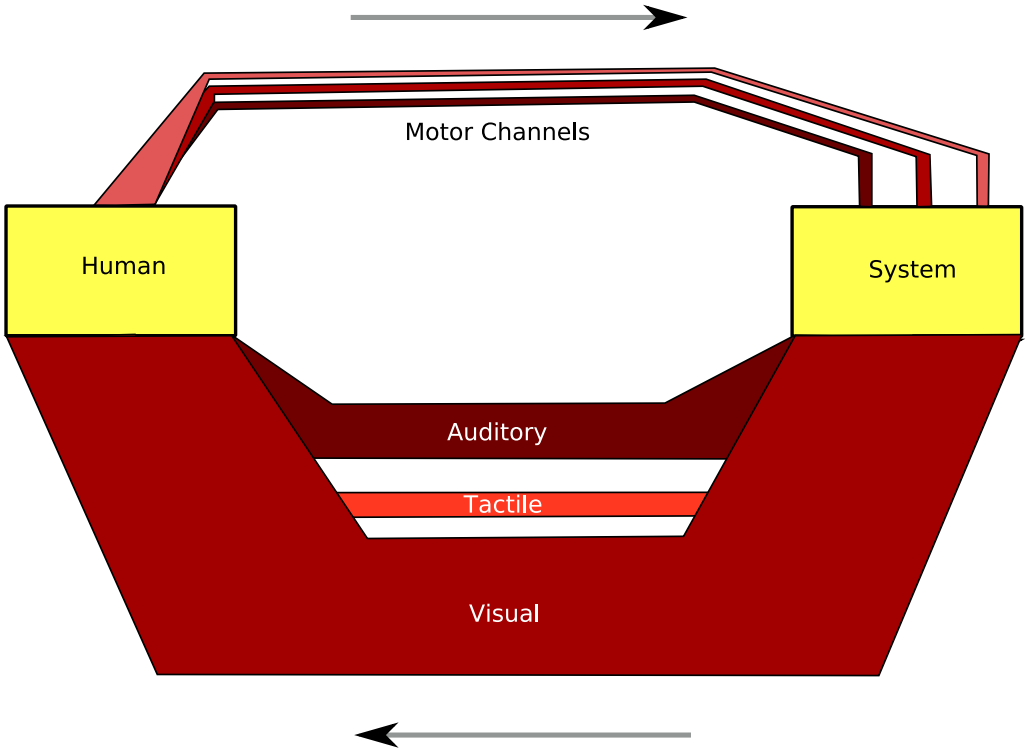


Figure VI.2: Asymmetry in the communication channels between an interactor and a computer. Human sensory channels have much higher bandwidth than the actuation channels.

Exact measurements of the capacities of the visual and auditory channels are difficult; but total bandwidth certainly lies in the hundreds of bits per second. The capacities of the eye and the ear have been experimentally bounded. Asakawa et al. [2003] suggest that experienced blind users can understand speech signals (which occupy a narrow spectral band) at rates of up to 500 words per minute (estimated from 1,300 Japanese morae per minute). This corresponds to about $1.5 \times 6 \times \frac{500}{60} = 75$ bits/s for signals in the speech band alone (and this ignores any information about speaker identity, prosody, timing and so on). Corliss [1971], for example, quotes the information capacity of the

ear as about 800-1600 bits/second, depending on the intensity of the stimulus. The visual system has even higher capacity; but determining its exact capacity is extremely difficult. Kelly [1962] suggests an (extreme) upper bound of 10^9 bits/s. The eye can also selectively track areas of interest, and so can cope when total display bandwidth exceeds the attentional bandwidth of the visual channel by observing only local variation.

The haptic communication channel is also significantly asymmetric. It is notable, for example, that the input frequencies (those generated by the limbs) are bounded at around 12–15Hz. Above this frequency, almost no intentional control can be effected over the position of the limbs. However, the tactile sensitivity of the skin has a much higher bandwidth, with spatial arrays of sensors having responses of up to 1000Hz (see e.g. Boff and Lincoln [1988]). Kuchenbecker et al. [2006] suggest that taking into account such bandwidth asymmetries is critical in the presentation of realistically stiff surfaces in force-feedback applications.

VI.2.7.1 MUTUAL INFORMATION

In particular, the evidence of interest can be characterised by $I(X;Y)$, the mutual information (MacKay [2003], Kantz and Schreiber [2003]) between a displayed time series Y and the control input series X – how much predictive information the display signal gives about the user actions. This quantifies the interaction directly in bits. In practice, direct implementation of mutual information approaches can be very computationally expensive as input dimension increases. Simpler approximate techniques, such as those described in Section VI.4.3.1, are often more practical for computation.

VI.3 A GENERAL AGENT BASED SELECTION FRAMEWORK; BRINGING THE GOALS TO LIFE

A complete interface can be viewed as an ensemble of systems for extracting intention from users.[1] These can be considered to be independent agents, one for each possible goal, whose purpose or intention is to identify whether a user's actions (which represent some user intention) are compatible with their assigned action. Figure VI.3 illustrates this. Each agent performs "experiments" on the user to test whether the user exhibits behaviour which increases the likelihood of the goal associated with the agent.

VI.3.1 AN AGENT VIEW OF INTERFACE COMPONENTS

Conventional interfaces generally involve static agents, fixed once and for all at design time. Each component in an interface (a button, a menu option, etc.) can be considered an independent agent. Obviously, not all of the information about the state of agents can be displayed simultaneously due to the limited display bandwidth of any device. Some process (e.g. hierarchical structure) must be used to display only the agents salient to the current decision at any specific time, to cope with limited display capacity.

The "experiment" such a static agent performs can be considered its *affordance*. A component is designed so as to appear useful for action. A well designed interface stimulates processes in the brain which directly relate to control of that object; the design reveals control parameters of the object.

[1] In a keyboard, for example, each key is a system for identifying the intention to communicate the label it bears.

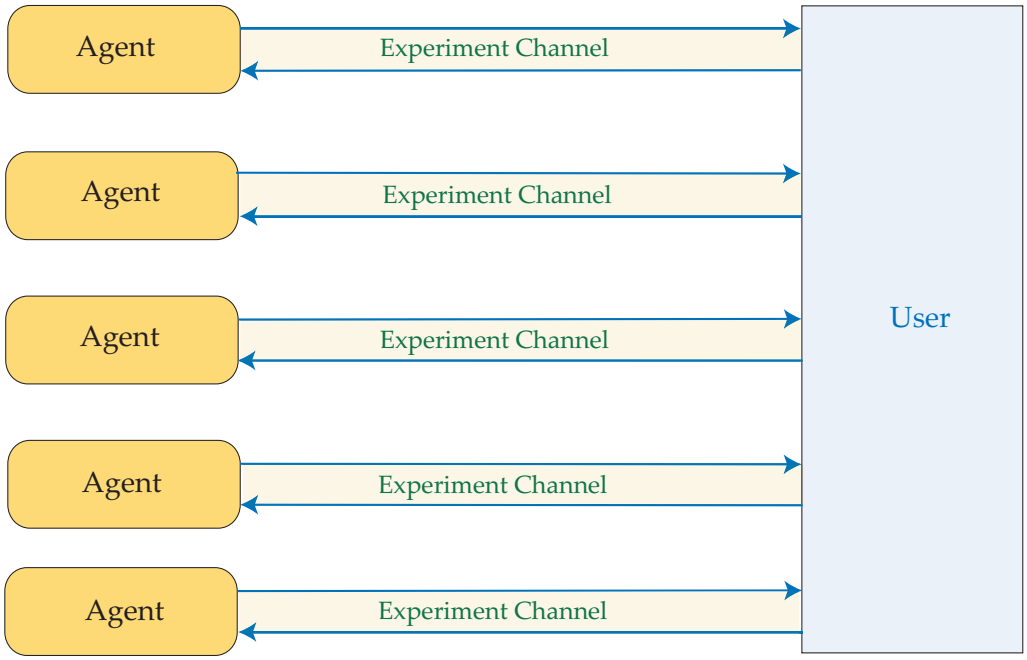


FIGURE VI.3: The agents as independent interactors. Each user interface goal (a vertex in the goal space) is now an active agent.

VI.3.2 ACTIVE AFFORDANCES – CONTROL BASED AGENTS

In contrast to existing approaches, dynamic agents can also be constructed. These create and display time-varying "experiments" – revealing ways in which they can be controlled – so as to optimally extract information from a user. The agent should be designed such that it changes state in a way which a user can control, and is independent from control actions that would affect other agents in the interface, thus maximising the potential discriminative power of the control signal.

The next sections discuss how such agents can be designed and built, and the advantages they can bring to an interactive system.

VI.4 THE STRUCTURE OF AN AGENT LOOP

A dynamic agent can be broken down into four major parts: the *model*, the *disturbance*, the *detector*, and the *feedback* unit (see Figure VI.4).

The **disturbance** generator generates the feedback patterns which the user attempts to control/imitate. These define the throughput of the system (since the maximum input entropy cannot exceed the output entropy in such a system) and a major component of the ease of use – some trajectories (in agent state space) are more difficult to control than others for the same total objective bandwidth.

The **feedback** unit displays the output of the disturbance generator to the user; this is another major influence on usability. Features such as preview display can strongly

affect user behaviour. The feedback unit also displays the state of the selection process; the position in goal space.

The **model** is a representation of the response of a user to the output from the feedback unit – it incorporates models of the limb dynamics, cognitive and perceptual delays and other transformations of the agent state to which the user responds.

The **detector** compares sensor input with the output of the model. The basic function is to identify information flowing from the feedback the user perceives back into the system sensors, and thus compute the probability[2] of interest for this agent.

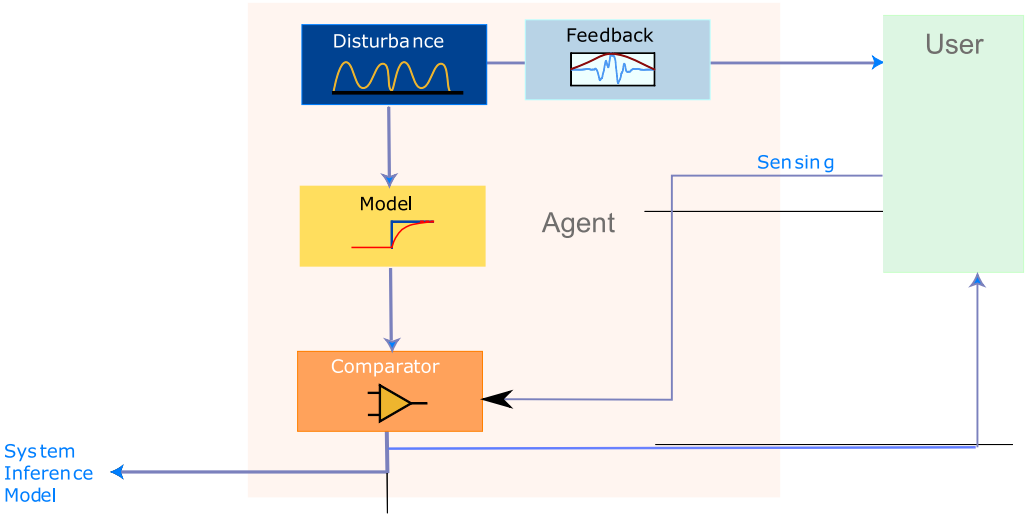


FIGURE VI.4: The structure of an agent loop.

VI.4.1 THE HUMAN/SYSTEM MODEL

On of the major advantages of this selection technique is that models of interactor behaviour can be directly incorporated into the selection mechanism. This can be done by comparing the perceptual variable presented to the *optimal behaviour a human would have performed* to control that variable. Delays, lags, transfer functions and any other model of how a user will respond can be built into this process. Since these operator models will be partially unknown, the detection algorithm can also integrate over parameters of the models to obtain a more robust estimate of the likelihood of interaction; it is also possible to adaptively optimise the parameter distributions during the interaction (e.g. to fit the behaviour different users, or the learning process of a single user).

VI.4.1.1 HUMAN OPERATOR MODELLING

Any of the models developed in manual control applications can be slotted into the framework. If they match the behaviour of a particular user well, the communication rate will increase. Transfer rate can be improved by better *modelling*, without having to restructure the entire interface.

[2]Most algorithms generate some value or score for interest, which must be normalised over all agents, and combined with a suitable prior, to obtain the probability of each agent.

There is extensive work in the field of manual control on modelling of operator behaviours in control tasks. Some postulated models of user behaviour are shown in Figures VI.6 and VI.5. The basic components of any operator model include an input reaction time delay, a controller with some level of predictive power, an output delay time and some output limb dynamics filter (normally modelled as a lag). This section briefly reviews some elementary models of human control which can be applied in detection of control behaviour. Much more sophisticated operator models can be built, but for the purpose of detection of control, simplified models suffice.

The approach also makes it easy to determine bounds on the efficiency of the interface by simulation. The interface can be automatically tested with an operator model and the communication rate attainable can be measured. Insofar as the operator model is accurate, this will provide an upper bound on the communication rate of the interface.

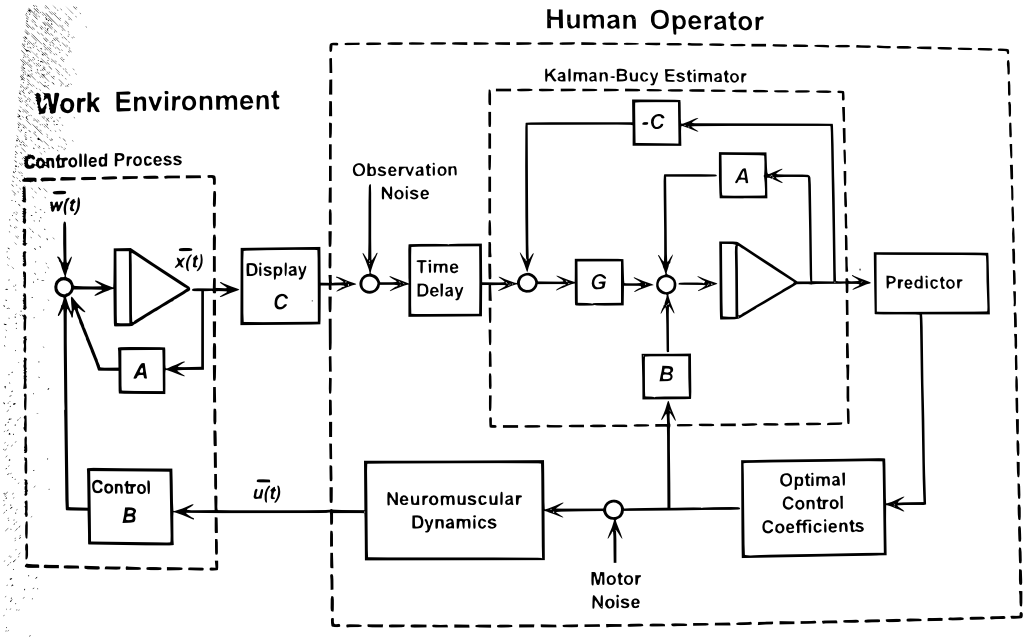


FIGURE VI.5: One postulated user model, from Jagacinski and Flach [2003] (p.205), in turn adapted from Sheridan and Ferrell [1974] (p.254)

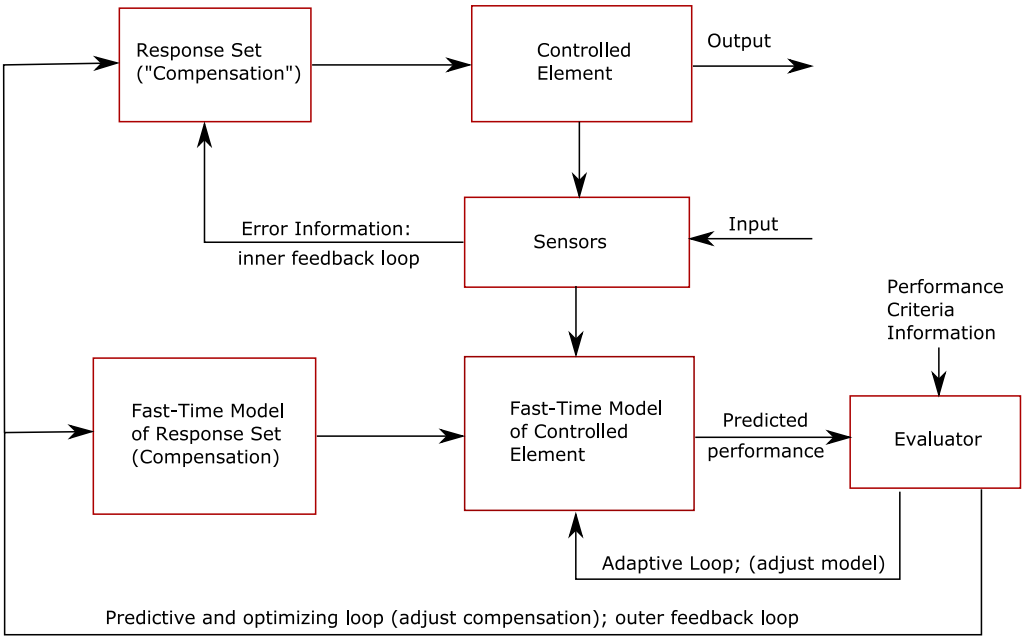


FIGURE VI.6: Another postulated user model, with a general predictive controller. Adapted from Kelley [1968] (p.140).

VI.4.1.2 CONTROLLERS In some cases human control behaviour can be approximated as a proportional linear response; this is often sufficient. Sheridan and Ferrell [1974] note that human tracking behaviour is normally approximately linear except when:

- the control process itself has significant nonlinearities;
- stiction, dead-zones or hysteresis are present;
- effort approaches the maximum the operator is capable of;
- very steep responses in the signal to be controlled are present;
- signals to control are reliably predictable;
- control inputs are discrete.

Gaines [1969], however, suggests that a bang-bang[3] model of control is more realistic for human control behaviour, especially with high-order control or heavily-delayed feedback. He notes:

> Conversely, because of the strong quantitative and qualitative resemblances between the output of sample data and "bang bang" models and that of the human operator, these seem to offer a firmer foundation on which to build extensive models of human skilled behaviour.

[3] A bang-bang controller exerts maximum control effort in the opposite direction to the sign of the error. No intermediate levels are used.

However, in situations where a system is subjectively "easy to control", (systems with low-order dynamics without significant nonlinearities) approximately-linear behaviour is a reasonable model of operator behaviour.

One example of a nonlinear operator model featuring both linear and nonlinear components is Costello's surge controller (Costello [1968]) (see Figure VI.7); this switches between bang-bang control when error or change in error is high to proportional control for small e and $\dot{e}$. This leads to rapid approximate acquisition followed by fine-tuning behaviour (the type of behaviour Phillips and Repperger [1997] suggested as an explanation of Fitts' law).

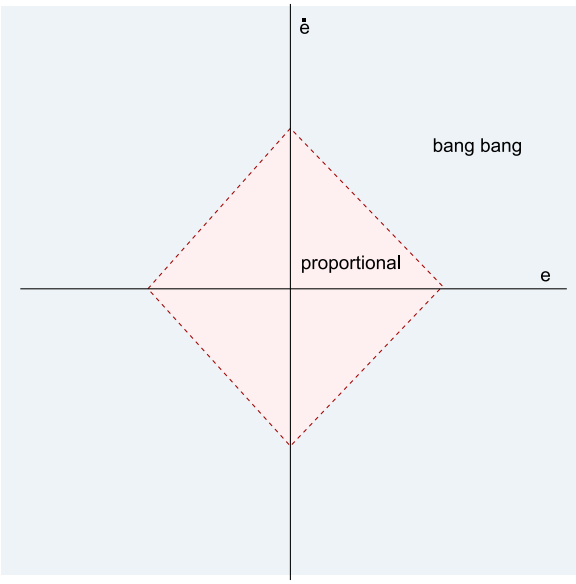


FIGURE VI.7: Costello's (Costello [1968]) surge controller. The error phase plane is shown (error against derivative of error). When the current state lies outside the diamond, bang-bang (full on or full off) control is applied, until the signal enters the central region, where proportional linear control takes over.

VI.4.1.3 LAG MODELS AND TIME DELAYS

One of the most common models of a human response in tracking tasks is a delay followed by a lag. Fixed perceptual delays are simple to model, and can be assumed to be relatively similar for all users. Typical delays are discussed in Section II.8.4.

Figure VI.8 shows typical step response for variously parameterised first-order and second-order lags. Operator response in smooth tracking tasks often has the characteristic behaviour of a first-order lag, following slightly behind the target, with a damped response. Sharply curved trajectories often show second-order properties, with overshoot and ringing near turning points and sharp changes.

Figure VI.9 shows tracking behaviour (with a target trajectory given by a sum of twelve sine waves), and corresponding first-order and second-order lag fits. The match between the trajectories is clearly improved (the lag parameter was optimised over the whole series in this example). The lag and delay models significiantly reduce the error between the disturbance and the observed response.

Where preview information is displayed, or the frequency of signals is close to the limit of performance, or where an interactor is unfamiliar with the properties of a control system, the behaviour may become much more complex than a simple lag. However, a lag is likely to remain as some component of the signal, purely because of the physical properties of the limbs.

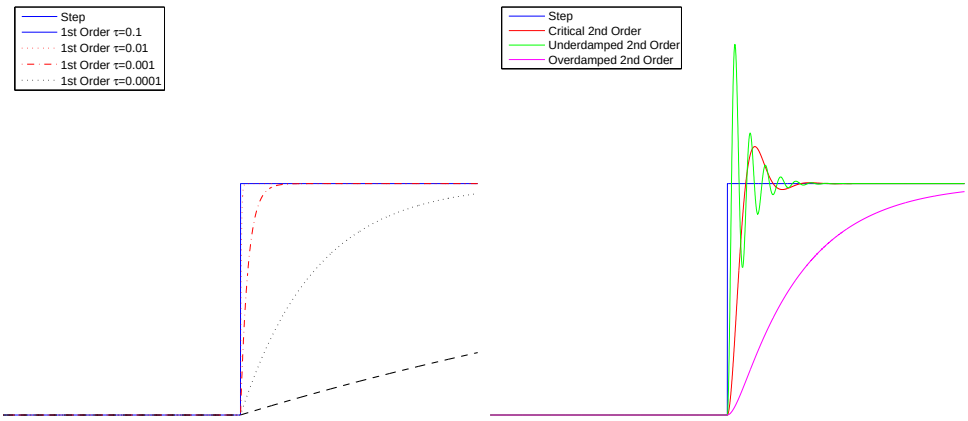


FIGURE VI.8: Step responses for first and second order lags with various parameters. Such responses (combined with delays) are often good fits to human tracking behaviour.

VI.4.1.4 SATURATION

One other common feature of human models is saturation. The range, force and velocity of movements a human can perform is restricted. Arm movements are restricted to a sphere of approximately one meter (although obviously part of this is blocked by the body). The total upward force an finger can exert saturates at some level. Input devices create constraints of their own; the range of a joystick is physically limited. Frictional and resistive forces restrict the maximum velocity of muscle movements.

VI.4.1.5 SPECTRAL COMPONENTS

Certain other characteristics of human behaviour (or *sensed* human behaviour) can be modelled in the frequency domain (see Figure VI.10). Muscle tremor, for example, is present in the 8-12Hz band (Beuter et al. [2003]) and is modulated by motions in the muscles. Although this can be used as a source of useful information (as Strachan and Murray-Smith [2004] demonstrate), in tracking or control behaviour it is effectively noise. It should be noted, however, that the presence and structure of tremor can indicate distinguish intentional control behaviour from other movements (see e.g. Morrison and Keogh [2001]). Including a tremor model can remove some of the variability associated with this phenomenon. Similarly, in tracking tasks which are performed during ambulation, narrow-band frequency components in the 1Hz range occur due to leg motion. If the sensing platform is sensitive to such disturbances, models of gait disturbances can be introduced to eliminate the motion from consideration as a part of the control behaviour.[4]

[4]Of course, a system could be built in which the control variable was some function of gait pattern. However, most interfaces sense motion from the upper limbs or head and lower limbs are not involved in control for the purposes of communicating intention to the system.

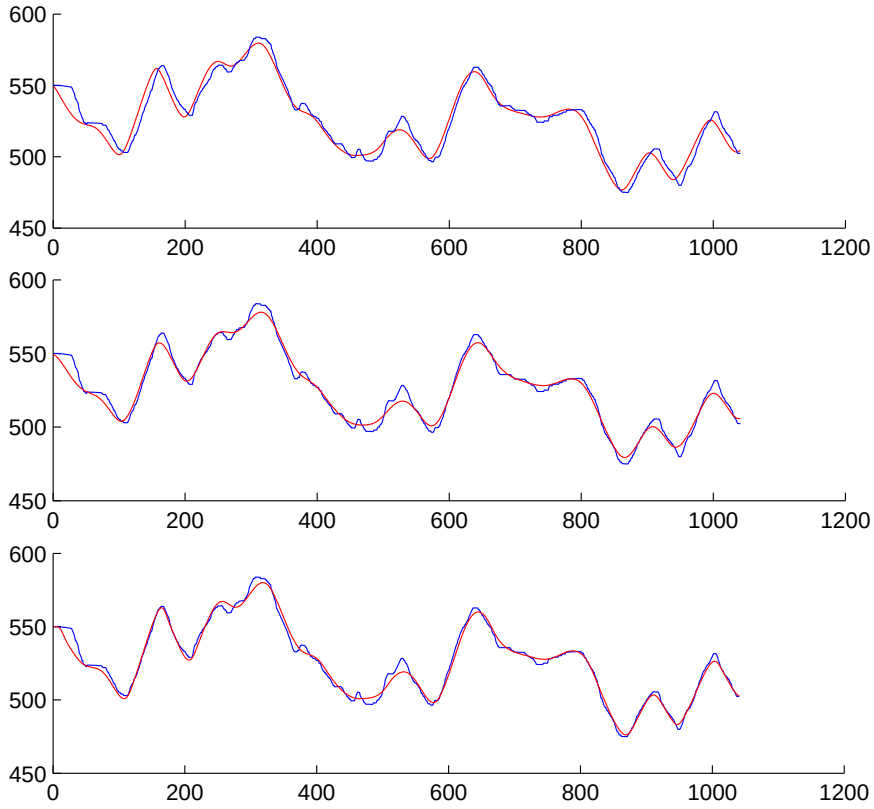


FIGURE VI.9: Compensation with first-order and second-order lags, for a single dimensional mouse tracking task. The first plot shows a generated disturbance signal (red), and the observed control input (blue). The sum-of-squared error between these signals is 3.19×10^4. The middle plot shows the effect of compensation with a 100ms delay and a first-order lag (with an optimised lag parameter). The fit is much closer with an SSE of 1.58×10^4. The bottom figure shows the application of a second-order lag and delay. The fit is extremely close with an SSE of 1.00×10^4. The second-order system (with an appropriate perceptual delay) models the dynamics of the operator well.

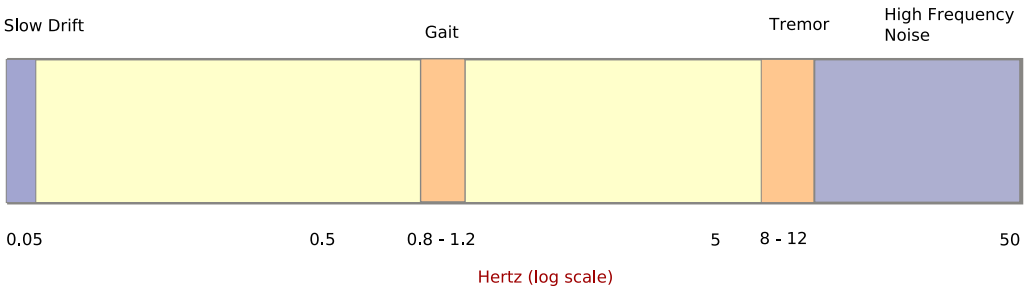


FIGURE VI.10: Some frequency components in human motion. Several bands are dominated by signals which do not relate to intentional control activity (with respect to the system).

VI.4.1.6 SENSOR MODELLING

Many existing sensors are well adapted for use in measuring human motion and are well-behaved when measuring human activity – the distortions they introduce are small compared to the distortion created by the operator when controlling some variable. However, sensors always introduce artifacts. Frequency components or phase distortion may be introduced or modified by the sensors themselves. The response of a sensor may need to be considered if the frequency response is not approximately flat over the range of interest. Delays will always be introduced (although these are usually shorter than those created by a human). Saturation effects are also omnipresent and can have a significant effect. Examples include the physical limitations of joysticks, mouse maximum velocities which are a result of limited communication bandwidth over serial protocols and accelerometer saturation effects, which generally limit acceleration sensing to a few G's.

VI.4.1.7 INTEGRATION, PARTICLE FILTERING AND ONLINE ADAPTATION

Measurements of the world are uncertain; noise is added between the generation of signals and their observation. Models of human behaviour are never accurate. This can be taken into account by integrating over possible activity given the signals measured, and over parameters of the operator model. This integral must generally be evaluated approximately (e.g. with Monte Carlo sampling). This is generally computationally challenging, and has not been implemented in any of the example systems described in this thesis. However, it may offer significant benefits in the identification of control behaviour.

Online adaptation of parameters (as opposed to integration) may also be beneficial (e.g. tracking the lag parameter as a user becomes more experienced with the system). Particle filtering techniques would be ideal for combining such tracking with parameter integration. The resulting system would incorporate fully probabilistic models of operator behaviour, more accurately reflecting the evidence for intention.

VI.4.1.8 MODELLING

In summary, there are a variety of models of human behaviour which can be applied to estimate the signals which the interactor is attempting to control. Simple linear models (lags and delays) are often sufficient; however nonlinear models such as the surge controller may offer better performance. Modelling of context-specific attributes such as saturation and spectral characteristics may be improve recognition quality in some cases. The introduction of such models directly into the interface process is one of the major advantages of the active selection approach over conventional interaction design methods.

VI.4.2 THE FORM OF EXPERIMENTS; DISTURBANCE DESIGN

The design of the stimuli presented by the system is of crucial importance in determining the level of information that can be extracted from the user. The discriminative power of the feedback depends on the disturbances being independent from one another, and from any background noise that may be present (such as tremor).[5] Excessively complex stimuli will be impossible for a user to control or imitate or will demand high levels of attention; smooth simple stimuli have a necessarily limited information rate. The stimuli presented must be as independent as possible (i.e. distinguishable

[5]Filtering in the operator/sensor model can remove the effects of tremor, but if the display has components only in this range, no information can be communicated.

over as short a time frame as possible), given the operator model of an interactor which will transform the signal.

Filtered white noise (from the output of a linear feedback shift register, for example) is one useful class of stimuli generator; the filter properties can be adjusted to fit the task at hand. Specific spectral regions which are known to be non-communicative (e.g. gait regions) can be omitted from the filter frequency response. Consequently, no display energy will be generated in this region. A similar filter can be applied in the detection system as part of the operator model. Other potential generation methods include Perlin noise (see Section VI.5.3) and sums of sinusoids (with distributions over phase, frequency and amplitude). All of these methods permit control over the spectra of the signals generated.

VI.4.2.1 Optimal Disturbances

The purpose of a disturbance is to provide a time-varying signal which a user can transform to indicate intention. Disturbances should be compatible with the possible signals generated an interactor, given the available sensors. They should partition the signal space so that intention signals are as far apart as possible; the risk of confusion should be minimised. Operator models can be used as an initial optimisation stage to maximise the potential bandwidth of the disturbances, and obtain bounds on the performance of the interface.

VI.4.2.2 Dynamic Optimisation Of Experiments

Such experiments can be adjusted in real-time, based on evidence about the controllability of that stimuli given the current conditions, and about the current need for evidence of intention. For example, as selection proceeds a number of items may become likely while others become very unlikely. At this point the most salient information is to distinguish between the likely information (or to indicate that none are relevant). Thus the disturbances produced by the agents associated with the likely goals should be as independent as possible; confusion with the already-unlikely agents is irrelevant as there is already strong evidence they are not of interest. For example, the interface can be designed so that the two most likely agents have precisely inverse disturbances; controlling one of them provides strong evidence for it and against the other. In general, choosing disturbances whose orthogonality depends of the ranking of outcomes is a difficult problem; simple heuristics like the preceding are easy solutions but are not necessarily optimal. Purely random disturbances are suboptimal (since the stimuli they generate do not depend on the current likelihood of the action with which they are associated), but are easy to generate and work reasonably well (see the implementations in Section VI.5.1).

VI.4.2.3 Multimodal Load Balancing

The optimisation of disturbances can also be used to distribute display energy across modalities depending on the interest displayed in each of the input channels. For example, a system can produce (independent) disturbances on both the visual and auditory channels; if evidence of interest is identified on one of these channels only, display energy can be concentrated in that channel, and attenuated in the other. This optimisation could be used to identify the most salient communication channels as interaction contexts change. Since the quality of selection depends on the quality of display – and the quality of selection can be quantified by the current bit-rate – the relative power of a display channel can easily be ascertained.

VI.4.3 Detection of Interaction: Sensing Intentional Behaviour

The detection of intention proceeds by identifying signals which are compatible with the disturbances generated, based upon the model of the operator. Each individual stimulus is transformed by the human/sensor model and compared with the signals sensed from the input devices. The detection unit computes the similarity of the signals, and estimates a distribution over goals.

The detector should discriminate between behaviour that is likely to be intention and that which is accidental, and between the various intended goals the system presents. Many algorithms for estimating the likelihood of interest are possible. One of the simplest is the ratio of variance technique, described in the following section. Direct implementation of mutual information methods is also possible.

VI.4.3.1 Ratio of Variance

The technique described by Marken (Marken [1995]) for detection of control behaviour is simple and effective. Given some disturbed variable, postulated to be under control, a comparison is made between the variability of the variable with and without the control signal applied. If control is being (successfully) applied, the variance of the variable should be reduced by the control action. By computing the variance σ_c of the display signal y_a and the sum of control and display signal $y_a + x$, for one agent a, over a running time window $T - k...T$ to the expected variance given no control σ_e (which is just the variance of y_a over this period), the ratio $r = \frac{\sigma_c}{\sigma_e}$ can be computed for each agent. When r is approximately 1, no control is being applied; when it is very small there is significant likelihood of interaction.[6] This process is illustrated in Figure VI.11.

[6] The actual likelihood is determined by the accuracy of the user model, and the time window over which the comparison is made.

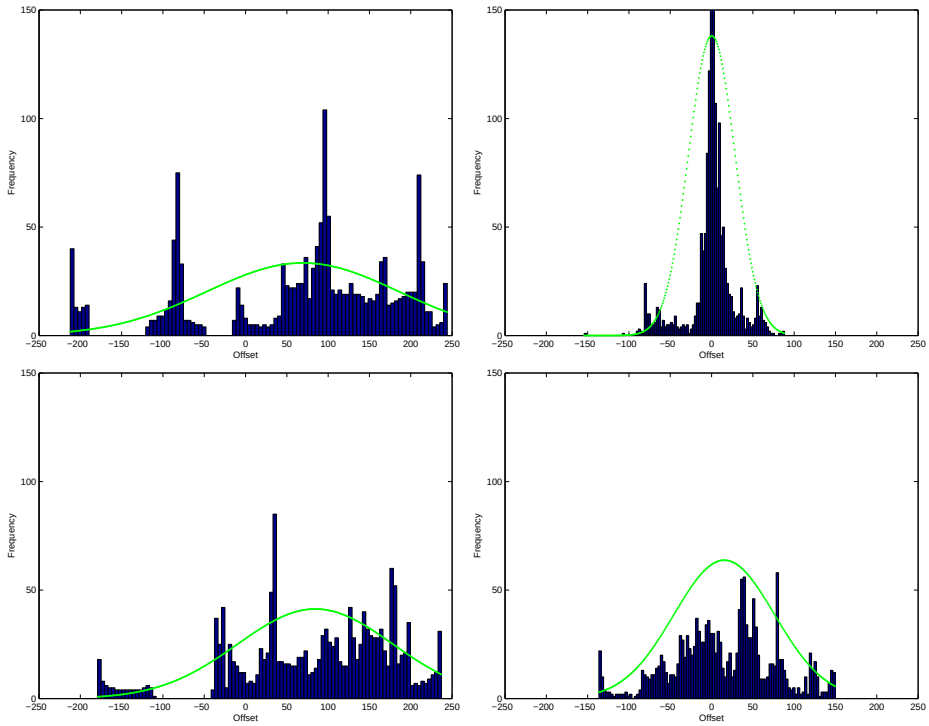


FIGURE VI.11: Controlled and uncontrolled agents and a computed Gaussian fit, for a single dimensional tracking task using mouse motion and a Brownian noise disturbance. Top-left shows the histogram (blue) and Gaussian fit (green) of an agent (the one the user is trying to select) without applying the control inputs. The effect of applying the control inputs to the agent is shown to the right, where the variance is markedly reduced. The lower pair of figures show the same process for another agent present during the selection task. Exactly the same control inputs were applied to this agent; however since the disturbances are independent there is relatively little change in the distribution of offset.

The output from this process is generally somewhat variable in its value. This can be improved by choosing a non-rectangular comparison window, down-weighting the oldest samples (for example with an exponential window). The values can also be smoothed by applying some filter to the sequence. One suitable process is a linear increase, exponential decay function applied to the output, based on a thresholded value of r. This involves computing:

$$o_t = \begin{cases} o_t + \alpha & r < \gamma(\alpha > 0, \gamma > 0) \\ o_t \beta & r > \theta(0 < \beta < 1, \theta > 0, \gamma < \theta) \end{cases} \tag{VI.1}$$

where γ and θ define the increase and decay regions respectively, and α and β give the increase and decay rates, respectively. o_t is computed for each agent, and the result normalised, giving an estimate $p(a)$. This produces a time series which looks like Figure VI.12 (based on the motion example of Section VI.5.1). The estimates of agent likelihood vary smoothly over time and the output from the estimation process can be fed back to the user during the interaction.

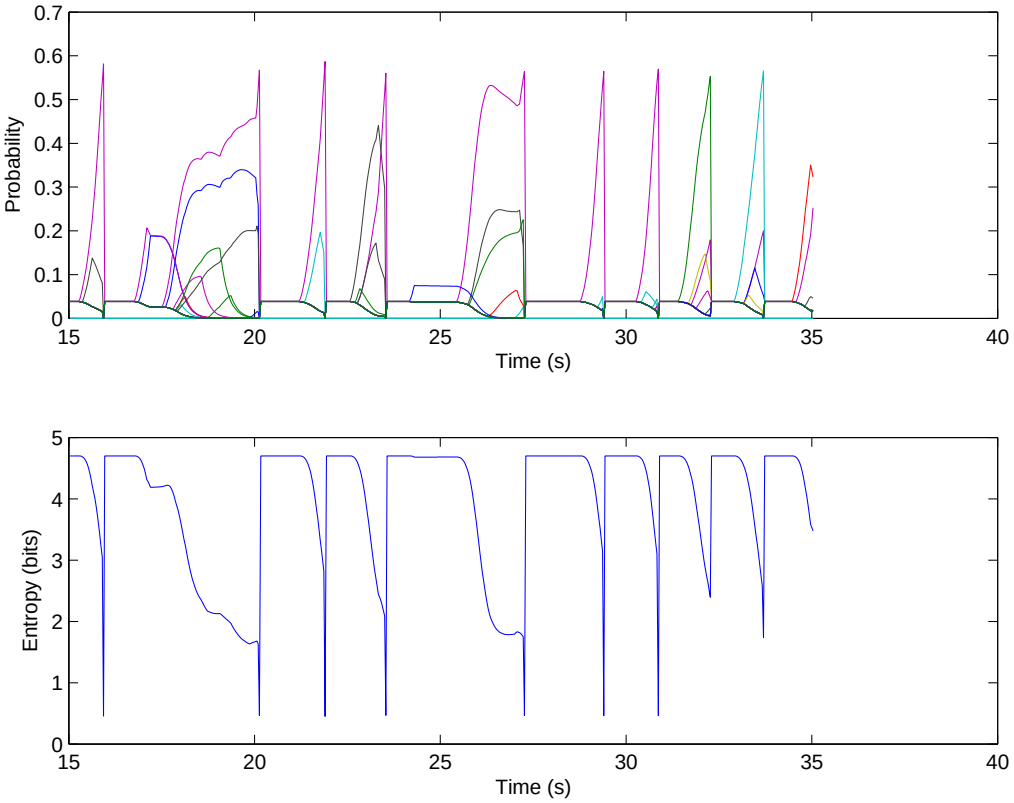


FIGURE VI.12: Probability and entropy time series for a selection task. The top panel shows the probability of each agent, and the entropy ($\sum_{i=0}^{n} p_i \log_2 p_i$) is shown below in blue. The gradually changing likelihoods of the hypotheses are visible. Selection events are indicated by the rapid increases in entropy (the vertical jumps in the lower diagram).

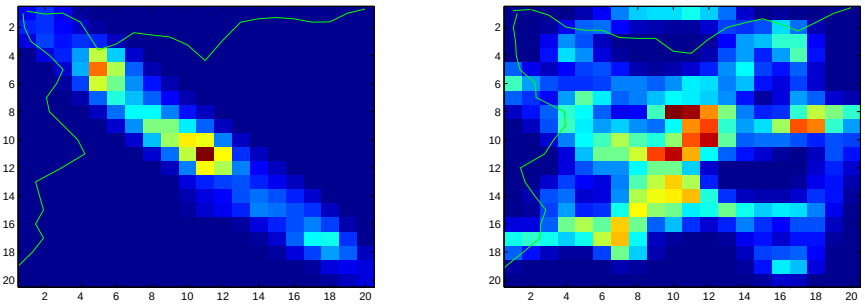


FIGURE VI.13: Simple computation of the mutual information between a one-dimensional display (position on screen) and control variable (mouse input) during a tracking task using simple histograms. The joint probability $p(x,y)$ is shown as colour level. The marginals $p(x)$ and $p(y)$ are shown in green. The mutual information is $\sum_x \sum_y \log \frac{p(x,y)}{p(x)p(y)}$. The left figure shows a controlled agent; the right shows a second agent which was not being controlled. The mutual information between the disturbance and sensed signals in the controlled case is 2.17 bits, while it is 1.11 bits in the uncontrolled case (although the bit measure is strongly dependent on the estimation procedure). This technique becomes less effective in higher dimensions (longer time windows, or high-DoF inputs).

VI.4.3.2 ALTERNATIVE: MUTUAL INFORMATION

A non-parametric estimate of the mutual information[7] could be implemented directly, for example via a histogram or other non-parametric fitting approach (e.g. kernel density estimation, see Figure VI.13 for an histogram-based example). The value

$$I(X;Y_a) = \sum_x \sum_y \log \frac{p(x,y)}{p(x)p(y)} \tag{VI.2}$$

is computed directly for each agent a, and the result normalized to produce the intention likelihoods. This direct implementation is afflicted with the curse-of-dimensionality – especially as it is necessary to compare the time series over some time window rather than at a single point, and it can consequently become very computationally expensive. However, it is a more general technique than the ratio of variance approach, and may produce better results where it is computationally tractable.

VI.4.4 PRIOR STRUCTURE AND INCORPORATION OF LONG-TERM PROBABILISTIC MODELS

Any prior structure over agents can be incorporated into the model, reweighting the intention estimates. Long-term inference (for example giving the conditional probability of sequences of goals) can be applied in a straightforward way. Nothing in the design of the interface needs change; only the level of evidence required for each agent is altered. If the likelihoods of each agent are being displayed to the user, the effect of the priors will be transparent to the interactor.

[7] This absolute measure is not the most useful measure for an interaction, where the relative information gain by assuming perception of feedback is more important. This measure can be approximated via data surrogacy techniques (see Palus and Hoyer [1998], Theiler et al. [1992]), where the two series are subjected to identical random transformations which destroy the relationship of interest but retain the other statistical properties (this is a common technique for detection of synchronisation of oscillators).

VI.4.5 FEEDBACK UNIT

The final unit of an interaction agent is the feedback component. This is responsible for displaying the state of the agent (and its disturbances) in the optimal manner to extract intention from the user. Good feedback should display the optimal behaviour that will communicate intention *in a way that a user can relate to*. Display of disturbance should relate to the control inputs a user can apply. The change of state should be displayed in a form which directly suggests manipulation with the control sensors an interactor has available. Two dimensional trajectories are ideal for mouse control, for example. A face expression control system could be visualised as an array of faces with flexible poses, which the user can manipulate by performing corresponding poses.

VI.4.5.1 MEMORY AND ITS EFFECTS

Systems which do not rely upon feedback for classification instead rely upon user models of (invisible) classes. This requires less feedback information, but requires significant mental effort and training. Memory based interaction is not strongly affected by delays in the control loop; motions can be communicated as bursts of internal-loop control activity. Display enhancements, such as prediction and preview increase the modelling power of an interactor, even without extensive training. These can bring some of the speed benefits of "open-loop" interaction into an otherwise closed-loop interface. Cues encoding specific motion patterns can be displayed in advance.

VI.4.5.2 PREVIEW DISPLAYS

In general, human tracking behaviour can be improved if the goal-related trajectories of a system can be displayed in advance (preview display). Most of the possible motions in any interface will not be information bearing; they will be equivalent to some other motion, or will have no meaning whatsoever. Showing the optimal (best separating, or easiest to perform) trajectories that communicate intention makes it easier to plan actions to communicate intention. The asymmetry of the communication channels means that it is quite possible to display detailed information about the future state of the disturbances, and thus simplify the control problem for the interactor. Doherty and Wickens [2001], for example, demonstrate increased control performance in (simulated) aircraft manoeuvres with "pathway in the sky" preview displays shown on the aircraft's HUD. Such displays can also be combined with predictive display (displaying how the system will respond). Many control-based selection interfaces can support such predictive displays, transforming the problem from continuous attention control to partially intermittent control. The pace of the interaction can be relaxed so that the timing of movements is not strictly synchronous with their display. This requires that the operator model be adjusted so that estimates of delays can be sufficiently flexible to deal with this relaxation (e.g. coping with leading behaviour as opposed to lag behaviour).

VI.4.5.3 PROBABILISTIC FEEDBACK UNIT

The agents must provide two distinct types of feedback. They must display their current disturbance in a form that an interactor can control; and they must also represent the current distribution over intentions (i.e. goal space position). It is natural to produce this feedback on a per-agent basis, since each agent represents a single intention. The feedback about goals can either be represented visually (as in the examples in the following sections), or can be sonified with the techniques of Chapter III. This display makes the evidence accumulation process clear to the user, and (at least for longer-term interactions) presents the system's beliefs in sufficient time for the interactor to intervene if the intention estimation is deviating from the true intention.

VI.4.6 SUMMARY: THE POWER OF THE CONTROL-BASED INTENTION AGENTS

The active agent takes advantage of the asymmetry of the human-computer interface to provide a powerful framework for selection mechanisms. This approach to the selection problem has several advantages over conventional approaches. The problem is cleanly split into the four major components described above, each of which can be dealt with separately and independently. Models of control behaviour can be used directly; they form a key part of the interface. Entropy changes gradually; the path through goal space is smooth and continuous, and the interactor can *control* the system's belief about user intention. The agent approach is general and is not tied to specific modalities. The characteristics of the available modalities can be explicitly accounted for, and the distribution of display energy over modalities can even be dynamically optimised during interaction depending upon their current communicative effectiveness.

VI.5 IMPLEMENTATION EXAMPLES

The following sections illustrate the implementation of active selection interfaces, with various display and sensing methods. These proof-of-concept interfaces demonstrate the active selection technique can be workable, but are not suggested as practical interfaces in their current form. Usable interfaces would require features such as hierarchical structure or focus and context displays. The models and detection algorithms are of the very simplest order in all of these examples; more sophisticated behavioural models would be necessary for high-effeciency interaction.

VI.5.1 EXAMPLE: BROWNIAN MOTION SELECTION

Figure VI.14 shows an example selection mechanism based on motion cancellation with a mouse input. The interface consists of a number of agents (in the image, twenty six agents – one for each letter of the alphabet – are shown), each of which follows a smooth, unpredictable trajectory across the screen. The trajectories are generated from integrated, lowpass filtered white noise:

$$y \sim \mathcal{N}(0,\sigma), \tag{VI.3}$$

$$z(t) = y^s\alpha + (1-\alpha)z(t-1), \tag{VI.4}$$

$$x(t) = x(t-1) + z(t), \tag{VI.5}$$

where α changes the filtering level and σ adjusts the scale of the motion.

Mouse input simultaneously affects the translation of all of the agents. An agent can be selected by controlling out its motion, stabilising its position. The probability of interest in an agent is indicated visually by an expanding circle around the agent's position, with radius proportional to the current probability of interest. The display also shows the moving average of the agents position, as a guide when stabilising the position. The agent trajectories are computed in advance, and a prediction of the future path of the agent is displayed extending from the agent's position. This, as described in Section VI.4.5.2, allows time to plan motions accurately. The ratio of variance algorithm (with the integrate/decay modification) is used to estimate the agent of interest. First-order lag compensation is applied with a fixed lag parameter (no integration over parameters is applied). The following table summarises the decomposition of this interface in terms of the components above:

Component	**Implementation**
Operator Model	First-order lag (fixed)
Disturbance	Low-pass filtered noise
Detector	Ratio of variance with smoothing
Feedback	Preview trajectory display

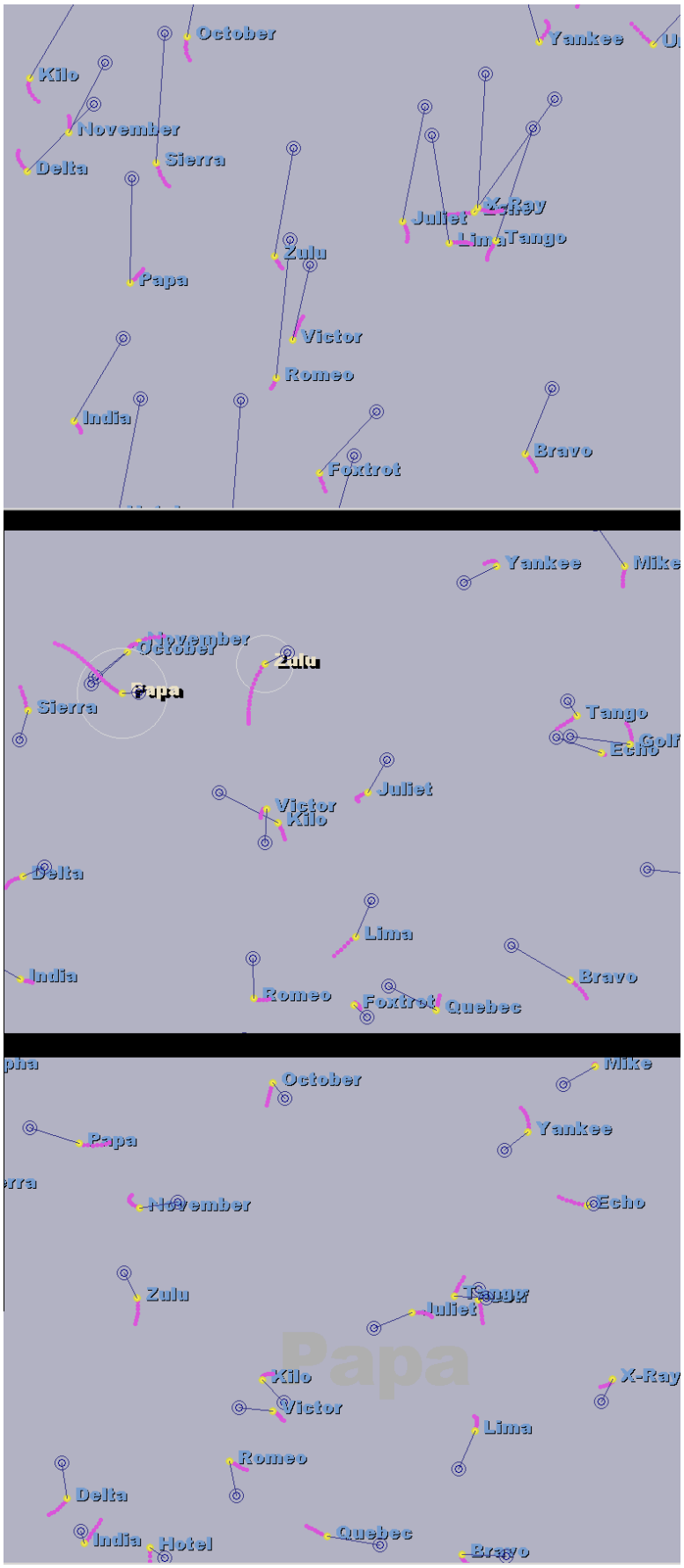


FIGURE VI.14: A sequence of images from Brownian motion selection example, from the initial state, through partial selection, to the completion of the selection process. Here, there are a number of objects (labelled with names from the phonetic alphabet) with smooth random disturbances. Correlating the mouse motion with the object motion results in selection.

Figure VI.15 shows a probability time series for the 26 agent case; Figure VI.16 shows the same for 100 agents; while Figure VI.17 shows the same for 1000 simultaneous agents. It is apparent that the selection mechanism scales well with increasing numbers of agents; the bit-rate remains around ∼1.5–2 bits per second across the cases. Testing beyond approximately 1000 agents is constrained by computational power.

VI.5.2 Example: Orientation Disturbances

Figure VI.22 illustrates a different configuration, which is significantly more compact and less visually noisy than the translation based display. It features a rectangular array of "egghead" objects – spheroids with cross-hairs and eye-like features superimposed on their surfaces – which are subject to random changes in orientation. The random orientation trajectories are designed to operate in a manner similar to the motion of human eyes, with rapid (but smooth) saccades rather than slow continuous trajectories. Selection is performed by orienting the desired agent so that it appears to face forwards. The accuracy of humans in determining the direction of gaze makes this an effective display method. High-quality antialiased drawing is used to obtain maximum accuracy in the orientation display, avoiding jumping artifacts when small changes to the orientation are made. Any image can easily be image-mapped onto the spheriods to distinguish them, as would be required for a practical selection interface.

The disturbance motion is a nonlinear transformation of a sum of sinusoids, with frequencies and phases drawn from Gaussian distributions so that each agent has a different signal generator. One such signal generator is assigned to each of the two axes about which the visual representation of the agent can rotate. The signal is passed through a nonlinear function to produce the saccadic effect. The generation for each axis is as follows:

$$x(t) = \frac{1}{2\left(1 + e^{-v\left(\sum_{k=1}^{n} \sin(f_k \phi + \theta_k)^r\right)} - \frac{1}{2}\right)}, \tag{VI.6}$$

followed by a saturation such that:

$$x(t) = \begin{cases} \beta_{min} & x(t) <= \beta_{min} \\ x(t) & \beta_{min} < x < \beta_{max} \\ \beta_{max} & x(t) >= \beta_{max} \end{cases}, \tag{VI.7}$$

where n is the number of sinusoidal components, v is a scaling factor, r gives the "jumpiness" of the signal, β_{min} and β_{max} give the saturation points, and ϕ_k and f_k give the phase and frequency of each component respectively. A time series from this function is shown in Figure VI.18 (for twelve sinusoidal components), showing the irregular changes in orientation with smooth transitions between the levels.

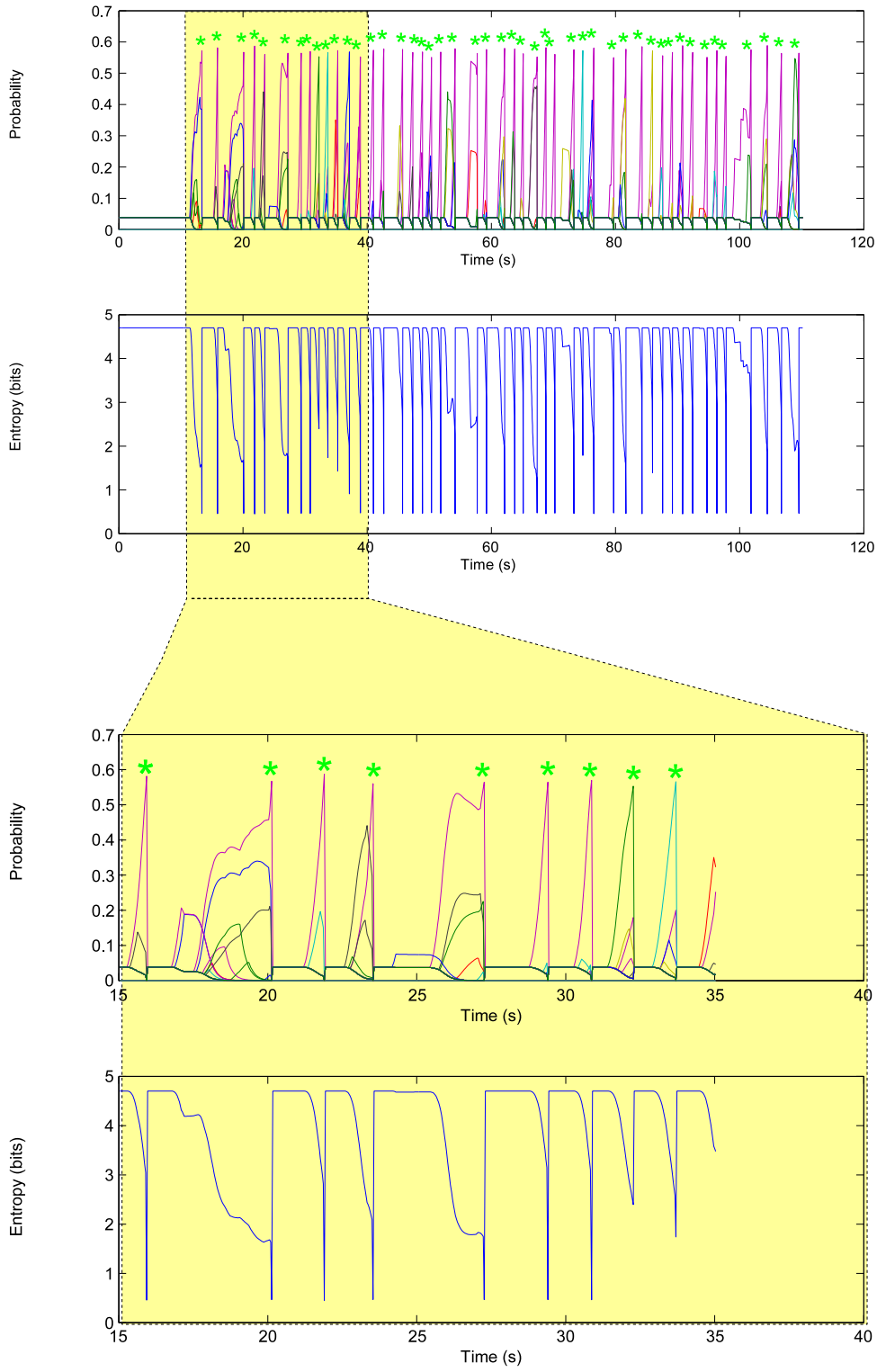


FIGURE VI.15: Probability and entropy time series for the Brownian motion selection example. The probability of each agent is shown, with the overall entropy shown below in blue. The lower figure shows a close up of the time series, where the continuous change of probabilities is visible. Green '*' symbols mark succesful selection events. The sharp drops in entropy happen when selection has occured and the state is reset. Each selection was separated by approximately 0.5–1 second; the visible onset of the increase in probability occurs about half a second after selection behaviour begins.

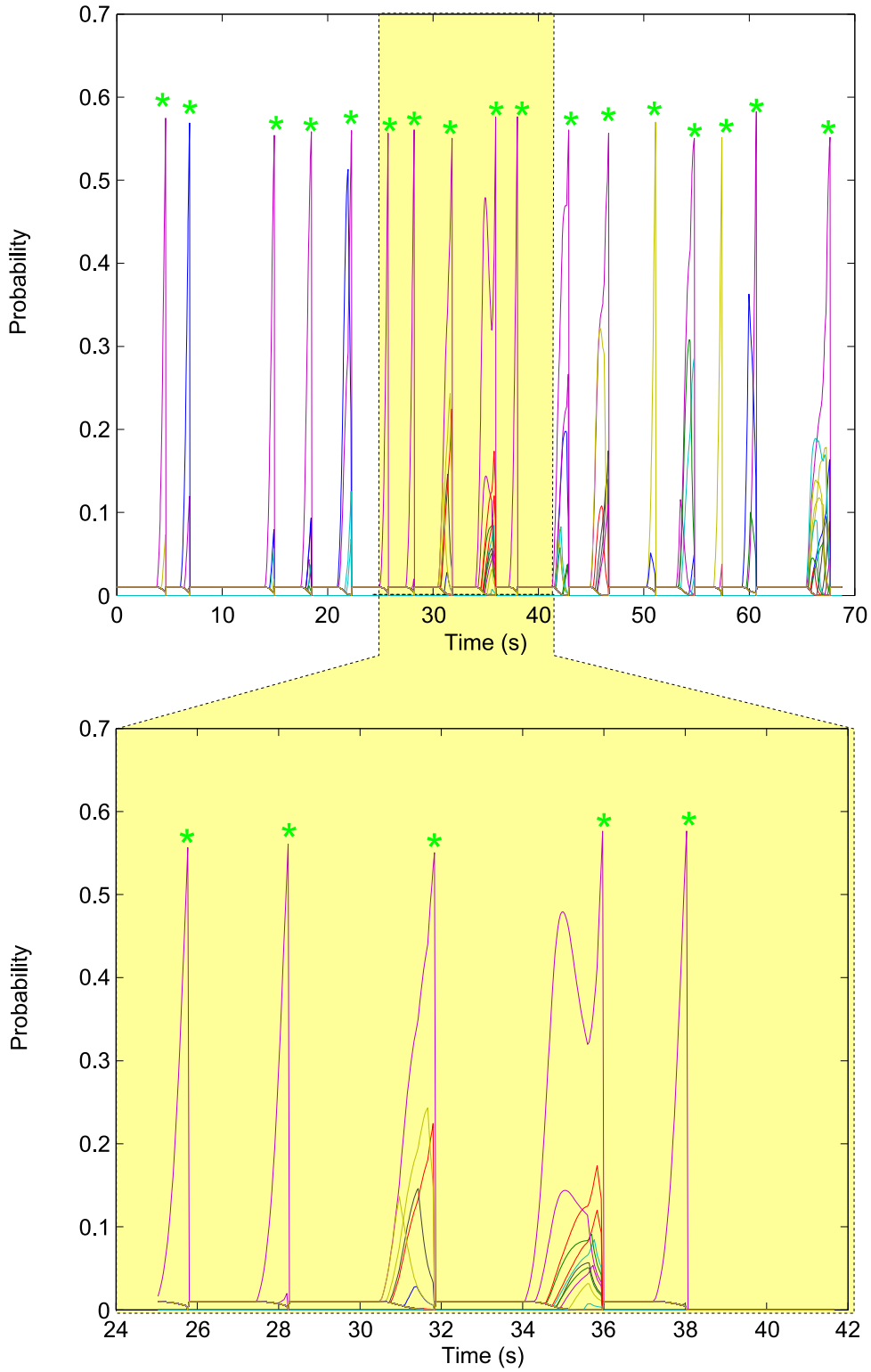


FIGURE VI.16: Probability time series for the 100 agent Brownian motion selection example. The lower figure shows a close up of the time series. Green '*' symbols mark succesful selection events.

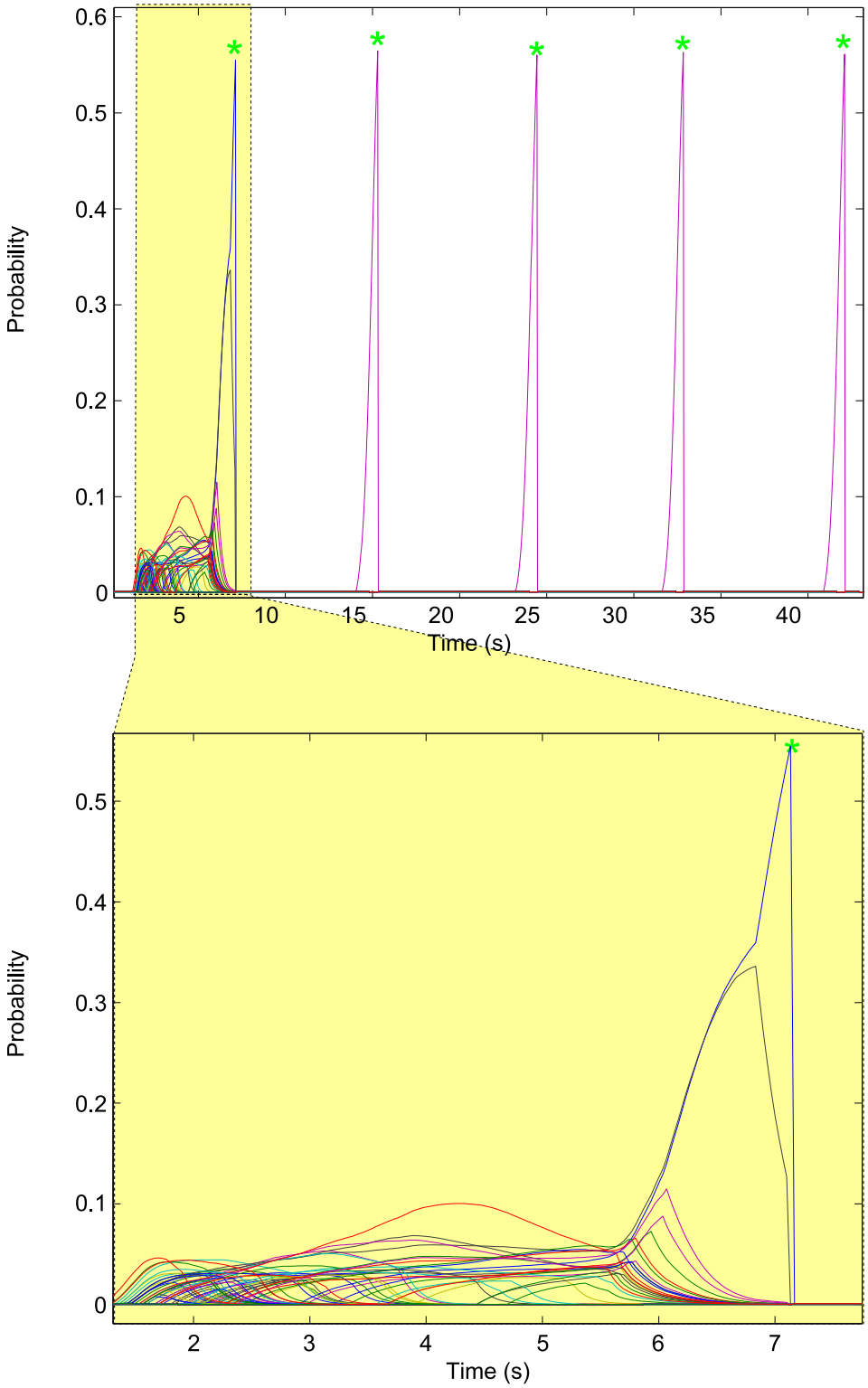


FIGURE VI.17: Probability time series for the 1000 agent Brownian motion selection example. The lower figure shows a close up of the time series, highlighting one single selection where many hypotheses become likely before fading away as new evidence arrives. Green '*' symbols mark succesful selection events.

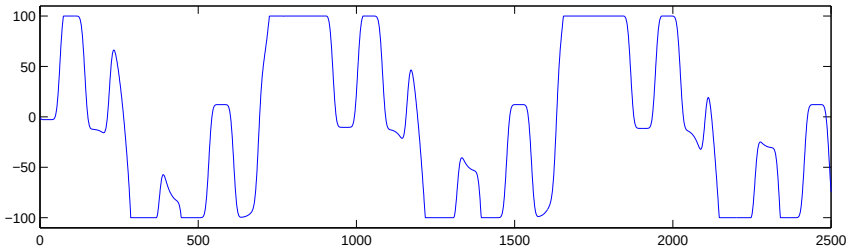


FIGURE VI.18: Time series from the "saccadic" disturbance function, for one dimension in orientation space. The function features plateaux with smooth, rapid transitions between the levels.

Figure VI.19 shows the probability time series for fifty objects, while VI.20 shows the series for two hundred objects. The bit-rates here are ∼1.9–2.2 bits per second.Figure VI.21 shows the fifty object case, using a joystick rather than the mouse for input. As the controller model was not adjusted to reflect this change, the rate of evidence acquisition is somewhat impaired, with bit-rates of ∼1.5 bits per second. However, selection is still quite possible.

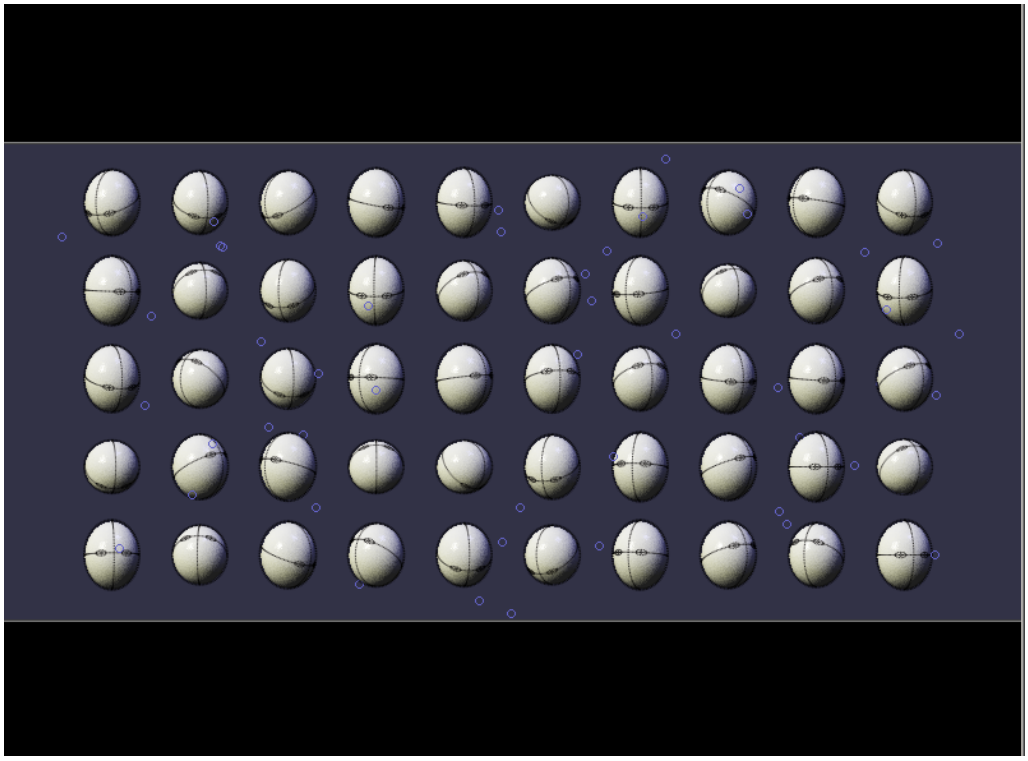


FIGURE VI.22: Here, there are again a number of objects (the "eggheads"), with relatively rapid but infrequent perturbations to their state. The orientation offset of all heads can be controlled with the input device. Maintaining a head in the "looking straightforward" position causes selection.

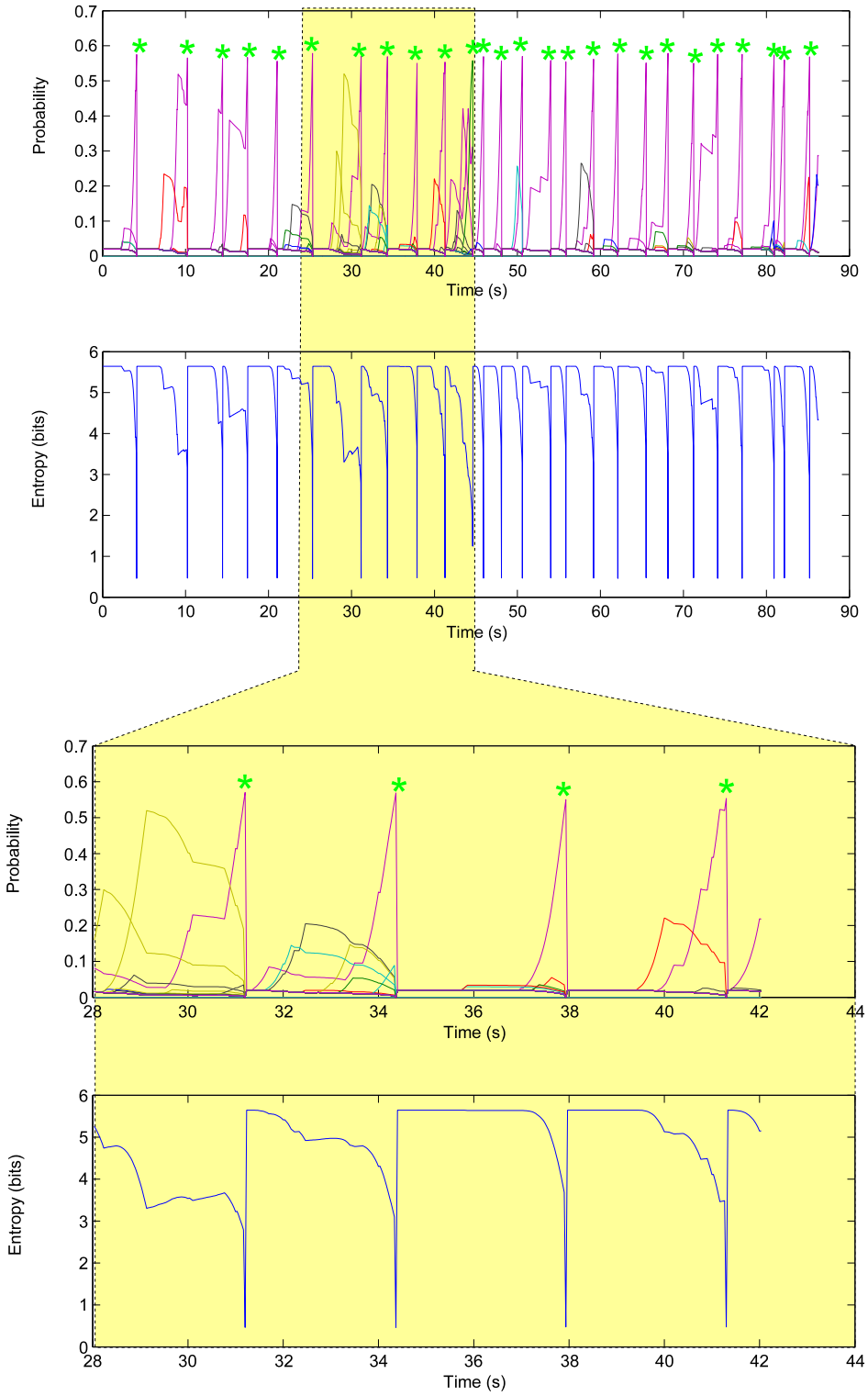


FIGURE VI.19: Probability time series for the "eggheads" orientation selection example. The probability of each agent is shown. The lower figure shows a close up of the time series, where the continuous change of probabilities is visible. Green '*' symbols mark succesful selection events.

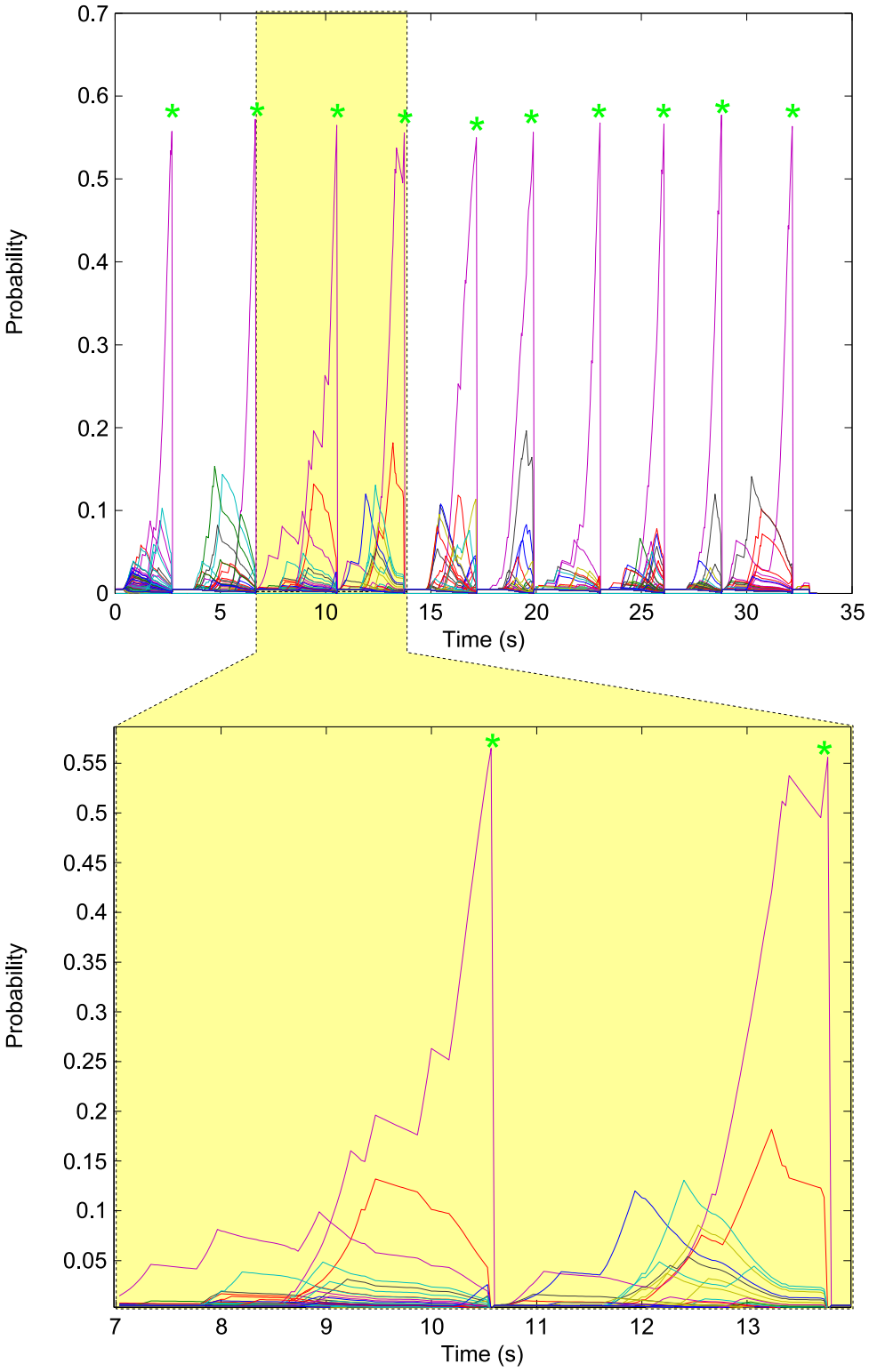


FIGURE VI.20: Probability time series for the "eggheads" orientation selection example. with 200 objects. The lower figure shows a close up of the time series. Green '*' symbols mark succesful selection events.

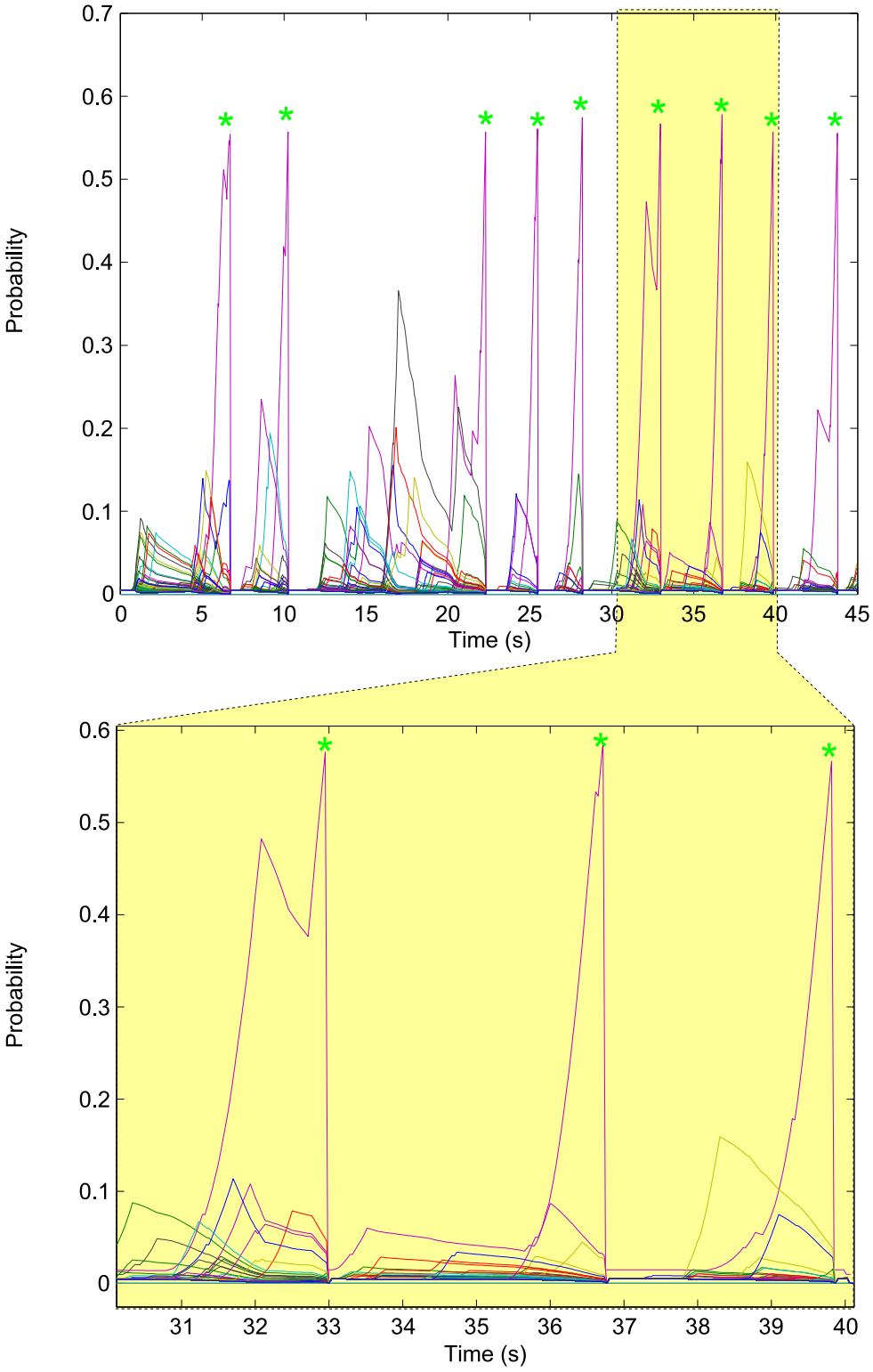


FIGURE VI.21: Probability time series for the "eggheads" orientation selection example. with 200 objects using a joystick for input . The lower figure shows a close up of the time series. Green '*' symbols mark succesful selection events.

VI.5.3 Example: Selection (Zooming) on Continuous Spaces

Thus far, the selection problem has been restricted to the selection of a goal or goals from a set of discrete alternatives. This is general enough to encompass many common user interface tasks. However, the technique can be further extended to continuous spaces. Rather than a set of individual agents, each looking for evidence of intention, the interaction can take place with a function which smoothly varies in time and space. This could, for example, be used for interaction on spaces such as maps, possibly in combination with a generalised fisheye lens (Furnas [1986]).

The principle is identical to the previous systems, except that the agents are replaced with a single (possibly multi-dimensional) function. This function is sampled, regularly or otherwise, and compared with the sensor inputs. The spatial covariance of the function can be used to specify the localisation of the selection; rapidly spatially varying functions will have greater discriminative power than smoothly-varying ones.

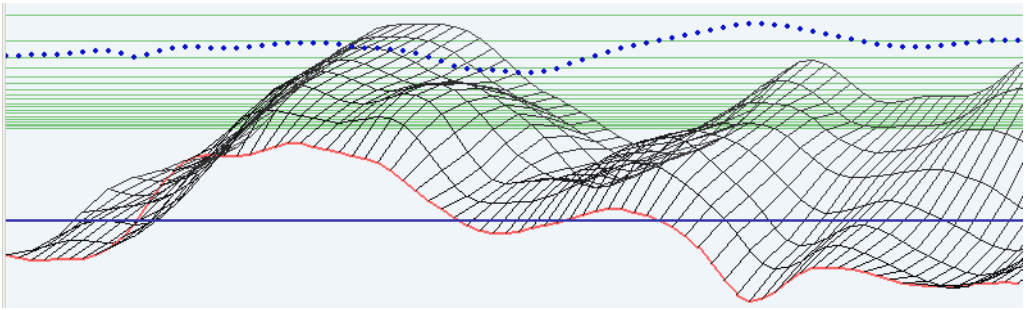


Figure VI.23: An image of the continuous selection interface. The red line shows the current state of the agent function. The black gridded surface extending backwards is a prediction of the future state of the selection line. Vertical mouse movement moves the entire surface up and down. Stabilising one part of the display increases the likelihood of interest in that neighbourhood; this is displayed via the circles visible at the top of the image, which represent the log likelihood at each sample point.

An implementation of such a selection interface is shown in Figure VI.23. The disturbance function in this case is a two-dimensional cosine-interpolated Perlin noise function (Perlin [1985]), which generates smooth random functions with configurable frequency content.[8] The function $z(x,y)$ consists of a sum of "octaves" of functions generated by interpolating between regularly spaced points with heights drawn from some

[8]The interpolator is based on the one described by Hugo Elias, at `http://freespace.virgin.net/hugo.elias/models/m_perlin.htm`, retreived 12th June 2006.

random distribution:

$$\begin{bmatrix} u \\ v \end{bmatrix} = A \begin{bmatrix} x \\ y \end{bmatrix}, \tag{VI.8}$$

$$j(a,b,\tau) = a\left(1 - \frac{1-\cos((\tau - \lfloor\tau\rfloor)\pi)}{2}\right) + b\left(1 - \frac{1-\cos((\tau - \lfloor\tau\rfloor)\pi)}{2}\right), \tag{VI.9}$$

$$l(u,v,i) = j(r(\lfloor\alpha^i u\rfloor + \lfloor q\alpha^i v\rfloor), r(\lfloor\alpha^i(u+1)\rfloor + \lfloor q\alpha^i v\rfloor), \alpha^i u), \tag{VI.10}$$

$$z(u,v) = \sum_{i=0}^{n} \gamma^i j(l(u,v,i), l(u,v+1,i), \alpha^i v), \tag{VI.11}$$

where n is the number of octaves, τ is an interpolation parameter, α is the octave size, A is some transformation matrix specifying the covariance of the function, γ is the octave weighting, $r(x)$ is a random number generator seeded by x,[9] and q is an arbitrary integer constant used to hash the co-ordinates. The result is a function with approximately $\frac{1}{f}$ frequency distribution (becoming more accurate as n increases). Increasing γ increases the proportion of high-frequency energy. This function is very flexible with regards to its spectral properties, and can be evaluated at any point without the evaluation of any other points. Some example Perlin noise surfaces (surface plots of $z(x,y)$) are shown in Figure VI.24.

In the implemenation, successive one-dimensional slices of the two-dimensional function are used as the disturbances (i.e. one dimension is used to represent time). Preview display is implemented, showing future slices of the function as a landscape moving towards the lower edge of the screen. The test for control is evaluated at regularly sampled points along the "current" slice, and the likelihoods computed for that slice. Figure VI.25 shows a time series of these probabilities for a noise function with a long length-scale; Figure VI.26 shows a time series for a function with a shorter length-scale (rapidly spatially-varying function). Although in these examples the function is regularly sampled (and so there is effectively a finite collection of agents), the function can be evaluated everywhere and a value for the likelihood of interest can be computed at any point; the probabilities are spatially as well as temporally smooth. The temporal smoothness is a result of the selection mechanism; the spatial smoothness is a result of the correlated structure of the disturbances.

[9] This could for example be implemented with a hash table and a standard random number generator, storing newly generated numbers only if the value at that point is not already in the hash table.

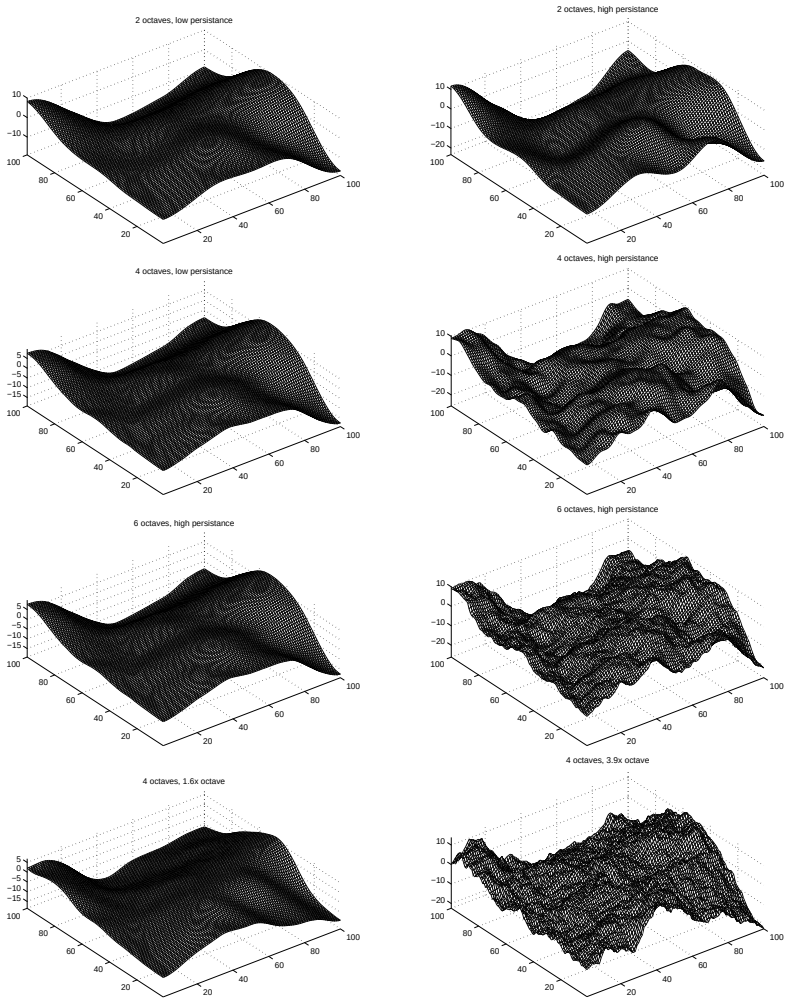


FIGURE VI.24: Perlin noise examples, showing the flexibility of the function. Left to right, top to bottom: 2 octave function with low persistance; 2 octave with high persistance; 4 octave, low persistance; 4 octave high persistance; 6 octave low persistance; 6 octave high persistance; 4 octave, medium persistance with an "octave" size of 1.6 instead of 2 (i.e. with dense frequency bands); 4 octave, medium persistance, octave size of 3.9 (widely spaced frequency bands).

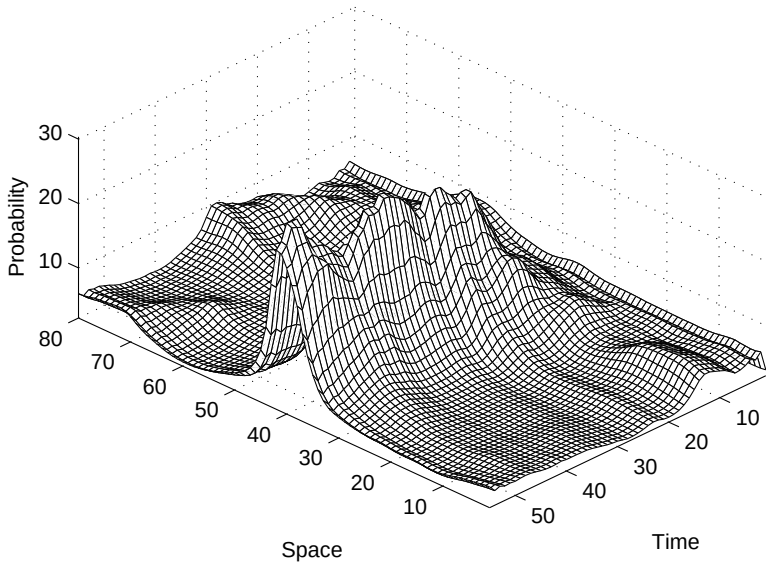


FIGURE VI.25: Probability time series for the continuous selection mechanism with a long length-scale. The rising ridge near the centre indicates the increasing probability of interest in that region. In this example, the interactor is attempting to select a region near $x = 40$. Probabilities are smooth in both space and time.

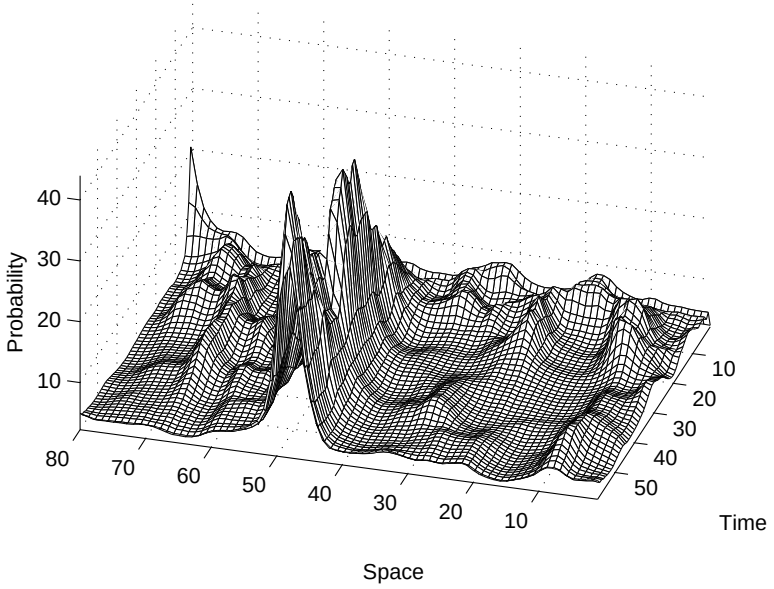


FIGURE VI.26: Probability time series for the continuous selection mechanism with a short length-scale. Here, the interactor is selecting a region near $x = 50$. The localisation is tighter than that of Figure VI.25.

VI.6 PERSPECTIVES

The active selection process can be viewed in a number of ways. So far, it has been considered as a compensatory control problem. The interactor applies control signals to damp the changes of state of an agent; controlling out the disturbances. Heavy damping increases the probability that the interactor is interested in the goal the agent is associated with. If the task is reversed so that the interactor is to move in *sympathy* with the agents, the problem is transformed from damping to *imitation*. The process is rather like some form of gesture recognition, with dynamically optimised gestures. The agents display a signal; the user attempts to imitate this signal. The *compensatory* display becomes a *pursuit* display. This involves changing only the display the user perceives – the underlying process remains entirely unchanged.

Alternatively, excitatory interfaces can be produced. In such an interface, interactors stimulate modes of agents. Agents have no activity until they are stimulated. Once they have some initial level of energy, they display ways in which their activation can be increased (e.g. by resonating). Such interfaces use little display bandwidth when nothing relevant is occurring, but display progressively more feedback as interaction

levels increase. The next sections describe how such interfaces can be implemented in practice.

VI.6.1 EXAMPLE: RESONANT INTERFACE

Excitatory interfaces involve the stimulation of particular modes of an agent. Figure VI.27 shows a very simple physical example; weights supsended on different lengths of cord can be *independently* stimulated by the holder.

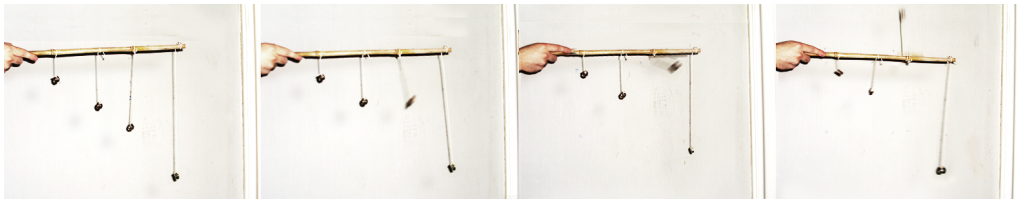


FIGURE VI.27: A physical resonant system. Weights are suspended from a bar by cord of varying length. It is easy to excite modes of the system such that one weight vigorously oscillates while the others stay relatively steady.

An excitatory interface initially displays no activity. The interaction is again self-revealing; the resonant modes of the object can be ascertained by stimulating the system with an impulse and observing the resulting motion. The desired agent can be selected by producing sympathetic motions. These modes can range from one-dimensional simple harmonic motion to more complex resonances (e.g. multiple resonant frequencies) or multi-dimensional resonances (for example, agents having resonances with specific inter-dimensional phase-relations).

Figure VI.28 shows how a simple resonant selection interface can be implemented in software. Selection proceeds by inducing sinusoidal oscillation in one of the agents, each of which has a specific resonant frequency. This frequency gradually drifts, with the rate of drift being a function of the current activation. This helps discriminate between two agents with (initially) closed spaced frequencies.

One-dimensional movements are transformed by a bank of high-Q (i.e. narrow resonant peak) bandpass filters with initially equally spaced centre frequencies (see Figure VI.28). The centre frequency slowly drifts randomly, at a rate determined by current filter energy. This interface has three basic stages of varying dependence on feedback. Users first inject an impulse to observe the natural frequency of oscillation and then phase lock with the oscillator. They then stimulate the resonant frequency, increasing the likelihood of interest in the agent. As the energy increases, the frequency drifts more rapidly and an observer must rely upon the feedback to retune movements. The total energy of each agent is normalised to estimate the likelihood of each agent; once this crosses some threshold, selection occurs.

Figure VI.29 shows the a time series of agent probability generated from an implementation of the above algorithm. The interface is one-dimensional, with mouse input and visual (position display) and auditory (chime sounds generated at the turning points of the agent trajectories, with volume proportional to the amplitude of oscillation) feedback. Figure VI.30 shows the position of the agents during the selection process, where the resonant characterestics of the agents are apparent.

This interface is highly inefficent in the form described, compared to other mouse input techniques (although it is suitable for other sensing devices which have naturally limited bandwidth, as the next section describes). The slow harmonic resonances have very little information content. More complex resonances (e.g. with patterns generated according to the disturbance generation algorithms described previously) could of course be introduced to expand the range of possible motions and consequently the bandwidth of the interface. Hybrid systems can also be constructed where agents have an excitatory phase followed by a damping phase. This minimises the "noiseiness" of the display when no interaction is required, while still permitting high-bandwidth selection.

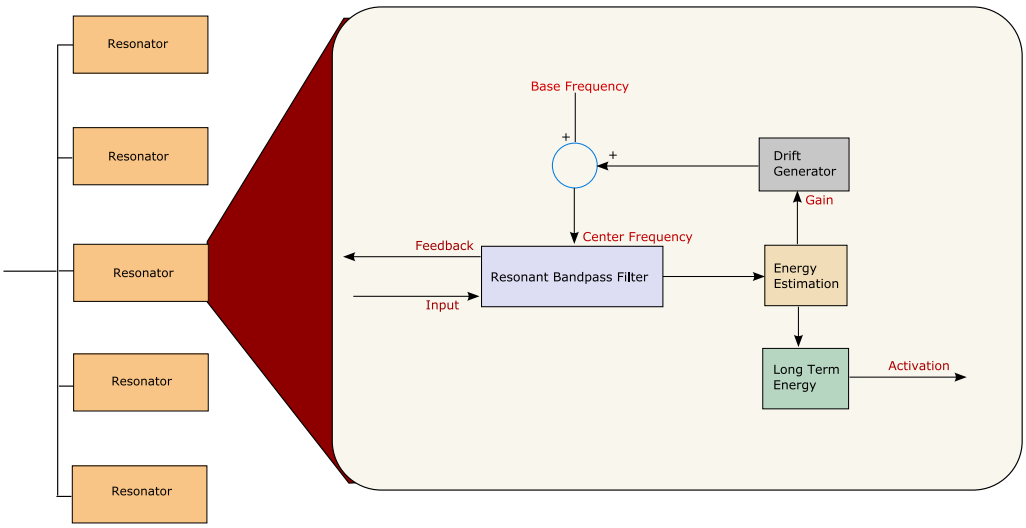


FIGURE VI.28: A resonant interface setup for selection. Input signals are passed through a bank of high-Q bandpass filters whose centre frequency drifts randomly, at a rate determined by their current energy. This type of interface exhibits time-varying dependence on feedback.

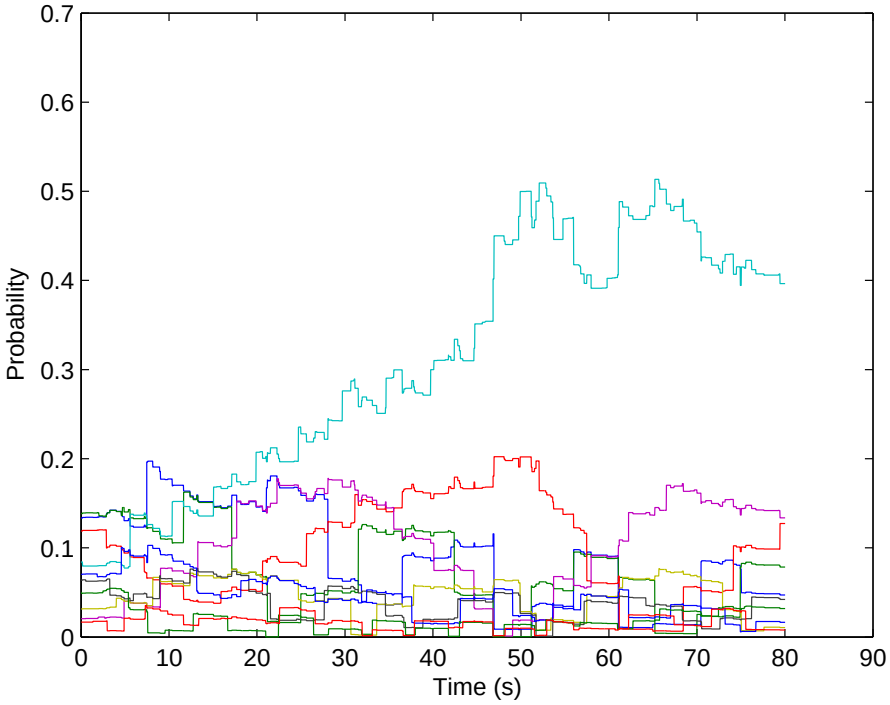


FIGURE VI.29: Probability time series from the one-dimensional resonant interface, where the fourth agent (light blue) is being selected. In this case selection would have been declared around $t = 30$, however the process was allowed to continue for the purposes of illustration.

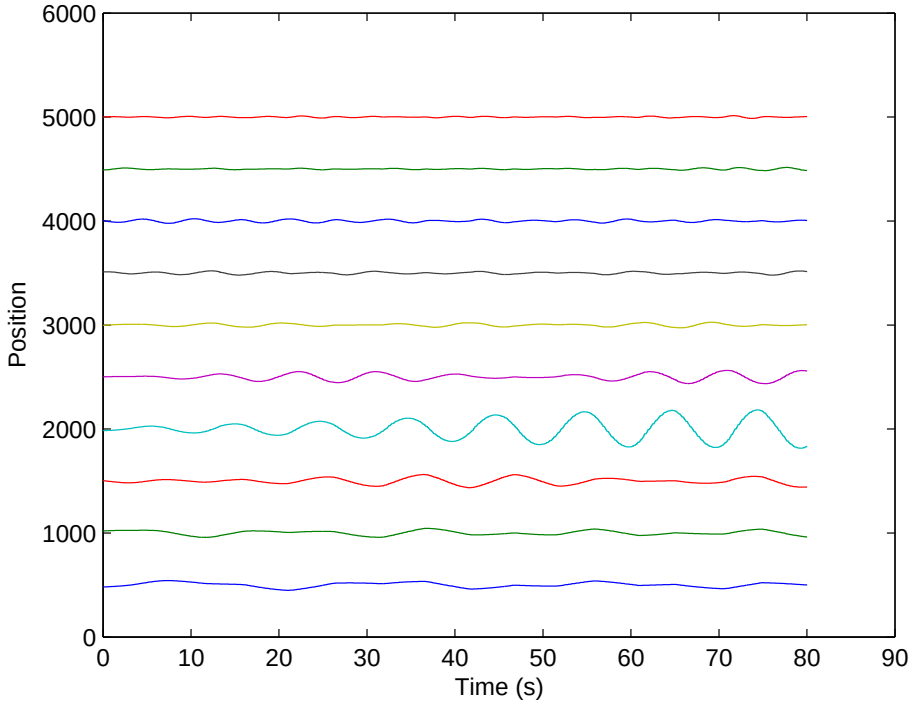


FIGURE VI.30: Position time series from the example one-dimensional resonant interface during the selection of the fourth agent from ten (light blue). The sinusoidal resonance of the agents is apparent.

VI.6.2 EXAMPLE: FOOT PEDAL INTERACTION

The selection interface illustrated in Figure VI.31 uses a footpedal (Figure VI.32) for control. The interface is identical to that described above for the mouse motion; the footpedals are a natural canditate for this style of interaction, being limited in control bandwidth, and easy to rhythmically stimulate.

The result is an interface with which selection can be performed via footpedals alone. Communication rates are approximately 0.35 bits per second, which is unsuprisingly significantly less than that attainable with mouse inputs. The limited accuracy of foot motions, and the low information content of the resonances restrict the achievable communication rates. However, it demonstrates that the technique can be adapted to new sensing modalities with little effort (although better modelling can of course increase the quality of the interaction).

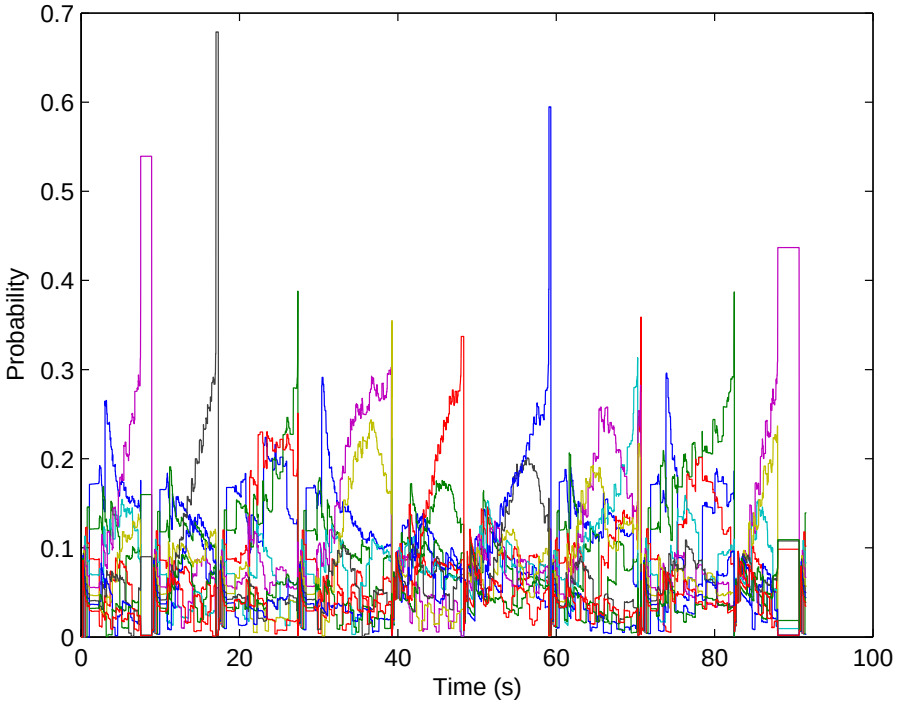


FIGURE VI.31: Probability time series for the footpedal selection example. As in the Brownian motion time series, each sharp drop indicates a selection event. Bit-rates of approximately 0.35 bits per second are obtained.

FIGURE VI.32: The CH Products footpedals used in this example.

VI.7 MONTE CARLO MARKOV CHAIN EXPLORATION

The following section[10] briefly outlines how the active selection approach could be used to explore and zoom displays and sonification in high-dimensional spaces. The idea clearly illustrates the power of the active selection approach.

VI.7.1 FOCUSING IN HIGH DIMENSIONAL SPACES

The problem of navigation in high-dimensional (but often sparse) data spaces is common in data visualisation and sonification. Many techniques have been developed for doing such exploration, such as parallel co-ordinates (Inselberg [1985]), worlds within worlds (Feiner and Beshers [1990]), or exploration along principal curves (Hermann et al. [2000]). Specialised input devices have been suggested for high-dimensional navigation (e.g. the data glove based n-Vision system (Feiner and Beshers [1990])). Particularly in sonification contexts, where the data must be presented in a primarily temporal form, and spatial cues are very limited, navigation can be a very challenging problem. The approach outlined in this chapter leads to a novel way of performing dynamic navigation, zooming and focusing on regions of interest – that correspond to postulated user goals – during exploration.

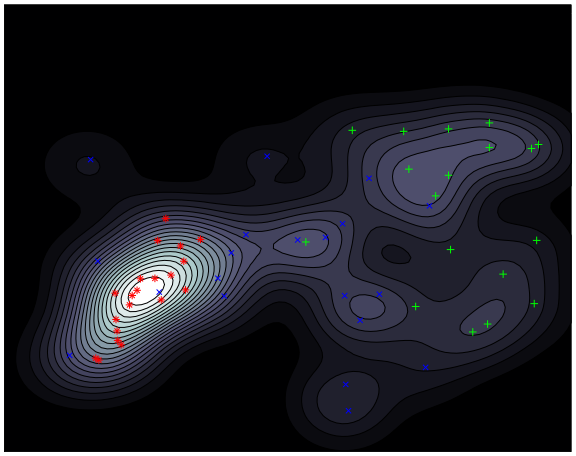
Navigating directly in the dimensions of the original space is challenging: the dimensions are often too numerous to control satisfactorily; the space is often extremely sparse, with only a tiny fraction of the total explorable volume containing relevant data; and the original dimensions may be meaningless to the interactor. Dimensional reduction techniques like PCA, force-directed models (Morrison et al. [2003]), ISOMAP (Tenenbaum et al. [2000]), or local linear embedding (Roweis and Saul [2000]) can identify "better" manifolds for which to display data, or on which to perform navigation. These static representations of the data space make it possible to navigate with low-dimensional control, but they are computationally expensive (they cannot easily be updated dynamically) and are based upon geometrical considerations rather than the hypothesised interests of a user.

As an alternative, it is proposed that the navigation be performed interactively by independent *proxy agents*, who explore the space under the control of the user and display relevant information about their local neighbourhood.[11] The user's control is expressed directly in terms of postulated goals. The agents, in effect, move along manifolds whose structures deform as the expressed user interest changes. The interaction becomes an exploration process, rather than static visualisation.

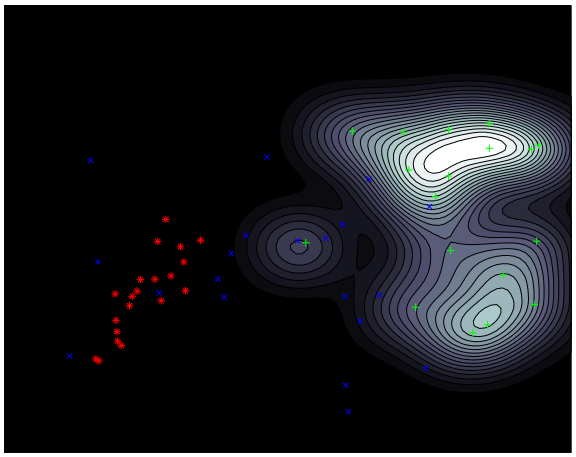
This is done by defining continuous *salience densities* over the high-dimensional data space. Such maps represent the predicted level of interest in regions of the space, given various user goals, i.e. $P(X|g)$. These are formed from combinations of basic, computable, properties of the data: density, local dimensionality, class boundaries (for labelled data), density isosurfaces, or more specific, task or dataset-related attributes. Figure VI.33 shows some simple example salience maps. These maps reflect particular aspects of the data an interactor may be interested in at some point in time. The quality of the interaction depends on how well these salience maps correspond to the user's true goals; poor identification of areas of salience will lead to explorations which are difficult to control and display irrelevant information.

[10] Some of the ideas in this section were developed in discussions with Thomas Hermann at the University of Bielefeld.

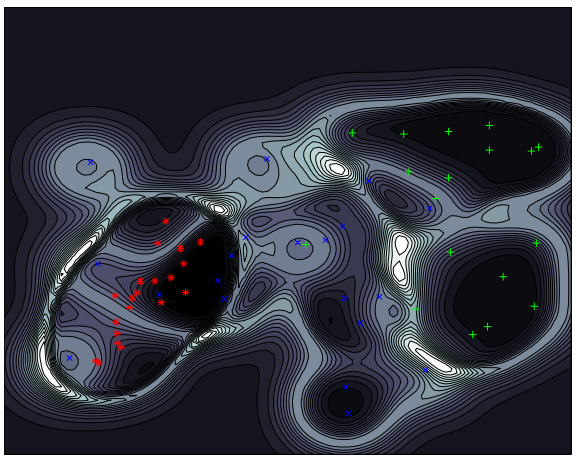
[11] Dimensional reduction is still useful in this context, to introduce additional dimensions to the space which represent "better" distance metrics between data points for the purposes of exploration.



(a) Data point density (from a Parzen window estimate).



(b) Class density for class 2 (green '+' symbols).



(c) Class boundary estimates.

FIGURE VI.33: Some example two dimensional salience maps, for a three class synthetic data example. Each class is drawn from a Gaussian distribution (with different covariance). Data points are shown as '+', '*', '.' symbols, one for each class; contours give the salience density. Labelled data is used for illustrative purposes; it is not a requirement.

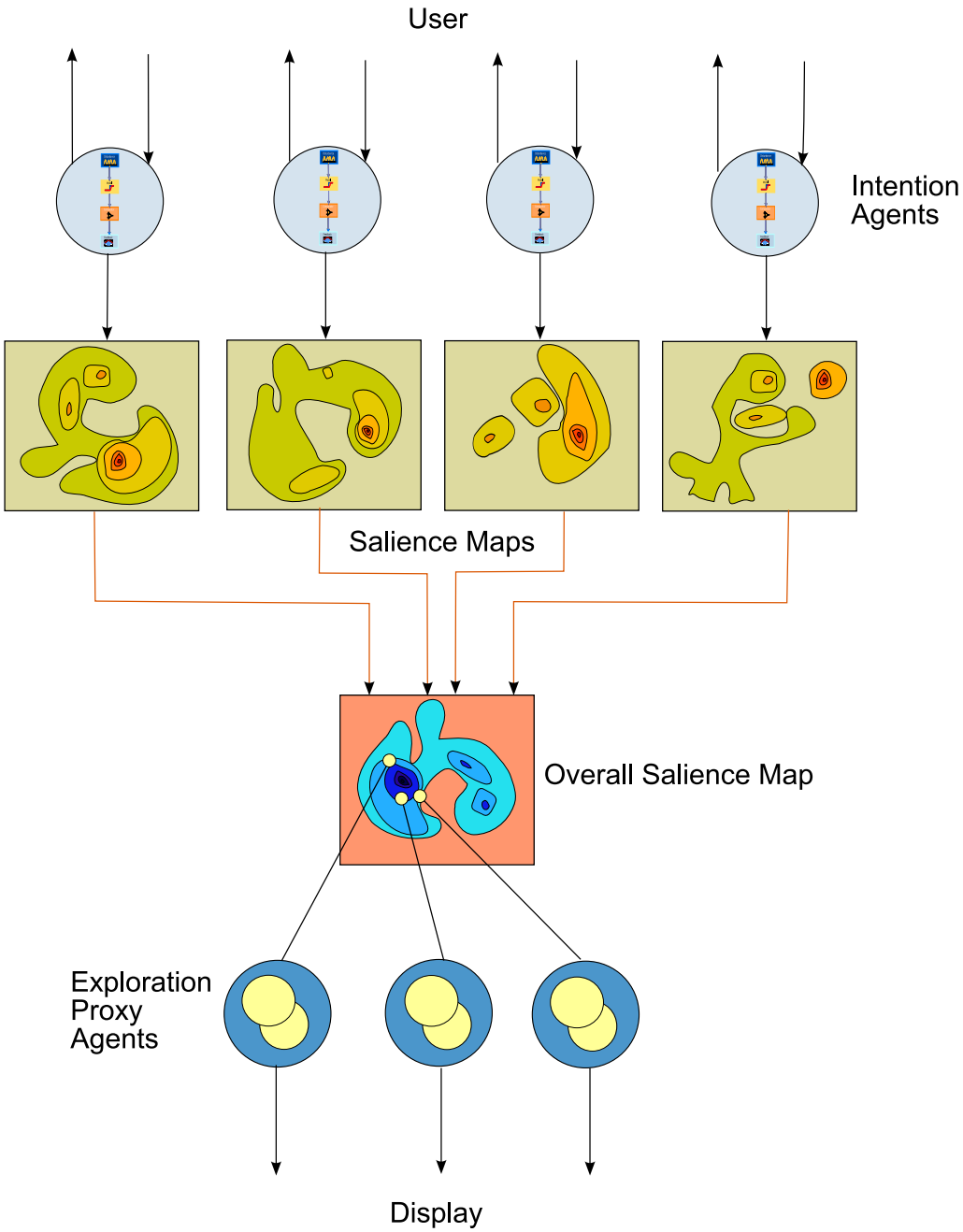


FIGURE VI.34: Proxy agents and intention agents in exploration. The intention agents identify aspects of the data the user is interested in; this is used to weight the combination of salience maps into a single map, which the proxy agents explore and visualise/sonify.

The proxy agents navigate the space, seeking to explore the regions estimated to be most relevant at the current instant. They display, sonify or otherwise reveal attributes of data in the vicinity of the proxy agent.

Along with the exploration agents, there are also intention agents of type described earlier in this chapter. Each intention agent is associated with one salience density. They produce disturbances, and attempt to ascertain which of these aspects the user is currently interested, exactly as in the interactions described earlier in the chapter. Figure VI.34 shows this structure. The exploration agents then focus on these aspects, adjusting their behaviour to traverse these newly relevant areas of the space. In other terms, the user "rewards" the system if the agent begins to reveal interesting structure, shaping the behaviour of the system towards salient regions of the space. If a user ceases to provide evidence for interest in any particular goal, the traversal relaxes to a global prior density (e.g. a salience density favouring data point density). As the user provides stronger and stronger evidence for a goal, the agent behaviour becomes more tightly focused.

VI.7.2 METROPOLIS FOCUSING

A proposed proxy agent exploration algorithm is a Markov Chain Monte Carlo process, which wanders the space, jumping around regions of high probability (i.e. high salience). This section explains how Metropolis-Hastings sampling can be used to implement proxy agents.

Metropolis-Hastings sampling (MacKay [2003]) is an efficient way of obtaining Monte Carlo samples from distributions, which extends well to large numbers of dimensions. It forms a continuous space, discrete time, Markov chain, selecting new samples from some local proposal density, and accepting them according to some accept/reject rule. The process moves randomly through the space, but roughly follows the density from which it is sampling. The proposal density defines the behaviour of the sampler; for many proposal densities the process will eventually converge. However, in the exploration task, it is not important to obtain independent samples from the original distribution; what is important is that the process wanders "interesting" parts of the space.[12] Figure VI.35 shows an MCMC process exploring a space with various salience densities.

A global salience density can be formed from linear weightings of individual density maps. The total probability of salience can be computed as $\sum_{i=1}^{n} p(g)P(X|g)$ for n goals $g_0 \ldots g_n$. This density is continuously updated as interaction proceeds. The MCMC process walks this (dynamic) density.

VI.7.2.1 THE ENTROPY LENS

When little interest in any goal is being shown by the user, it is likely that the system is failing to display any aspect of interest. Thus it should seek to change its behaviour so as to reveal more of the space. Conversely, if a user shows very strong evidence for interest in an aspect of the data, the system should endeavour to focus more tightly. This can be achieved by changing the rate of the random walk – adjusting the width of the proposal density to widen when goal entropy is high and to shrink as goal entropy decreases.[13] This "entropy lens" zooms in with strong interest, and relaxes back to a rapid summary of the data as overall interest decreases.

[12] Many other exploration algorithms are of course possible. Hybrid (Hamiltonian) Monte Carlo processes (MacKay [2003]), which include derivative information when making jumps, are one possible alternative. These can be significantly better at exploring the space.

[13] Salience densities themselves do not specify locality – only regions of interest. The proposal density defines the locality of the jumps.

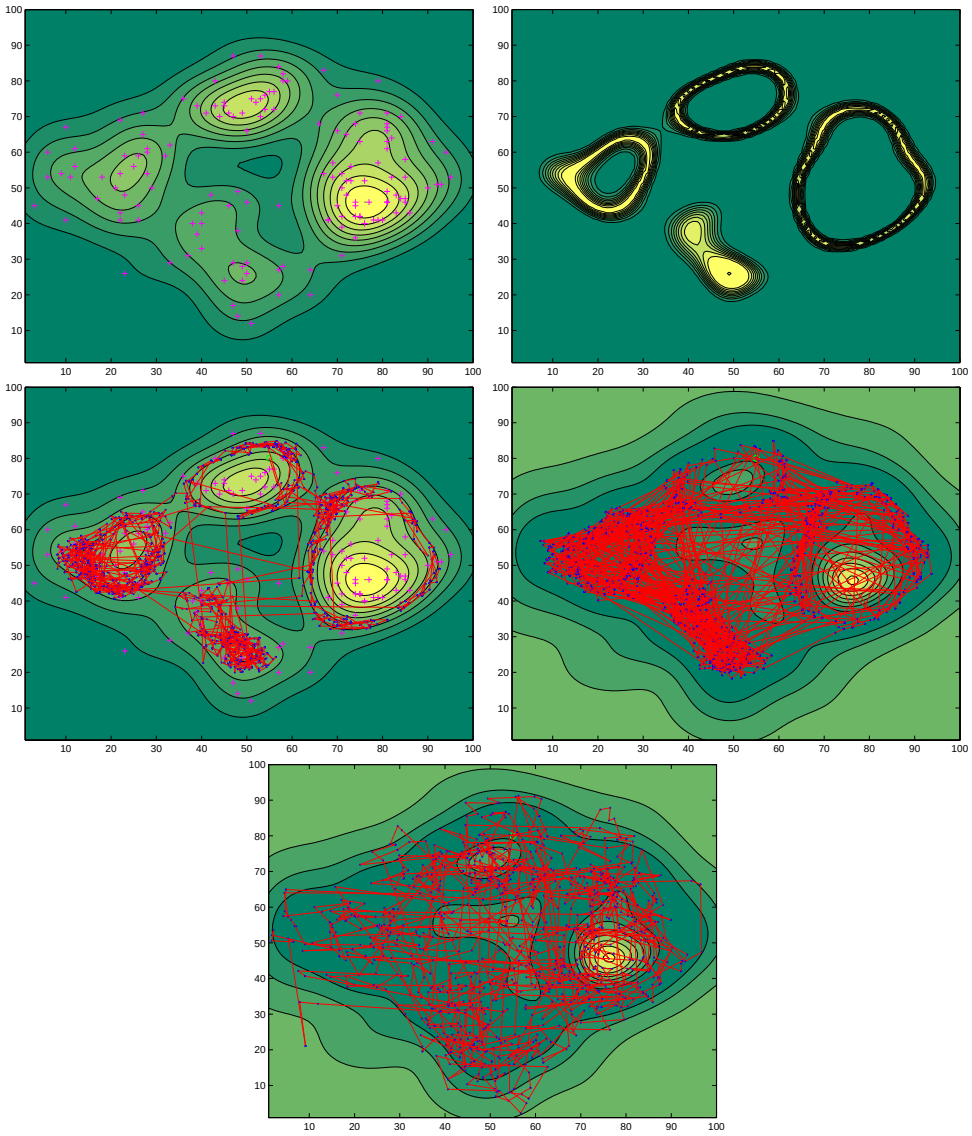


FIGURE VI.35: Markov Chain Monte Carlo exploration example. This is a 2-dimensional example for ease of display. (a) shows data points in a space, with a Parzen window kernel density estimate overlaid as contours. (b) shows a (Gaussian) density around the isocontour $p = 0.2$ of the density in (a). (c) shows the MCMC process (with Cauchy proposal density) where the estimated goal is the density shown in (b), overlaid on the original data. The process walks the boundaries of the data density. (d) shows the MCMC process with increased proposal width (this would correspond to higher goal entropy). (e) shows the MCMC process relaxing back to a exploration of the data point density.

VI.7.2.2 OUTLINE OF THE ALGORITHM

In the following, the proposal density will be assumed to be isotropic. The Metropolis agent algorithm operates as follows:

- Starting from a point s;
- for each time step t:
  - The proposal density is adjusted so that the $\mathrm{var}(Q_t(x)) = g(\gamma)$, where γ is the current selection entropy and g is some function (for distributions without variances, some appropriate width scaling parameter can be used instead);
  - a new sample s' is drawn from the proposal density $Q_t(x)$;
  - the total salience density is computed as

$$P_t(x) = \sum_{k=1}^{n} \lambda_k f_k(x); \tag{VI.12}$$

  - accept jump to s' with probability

$$p_j = \begin{cases} 1 & P_t(s') >= P(s) \\ \frac{P_t(s')}{P_t(s)} & P_t(s') < P(s) \end{cases} \tag{VI.13}$$

  - feedback from the local neighbourhood of s is produced;
  - the intention agents update their disturbances;
  - evidence for each goal is computed by the intention agents, updating $\lambda_0 \ldots \lambda_k$;

The display or sonification can display relevant attributes of the data in the locality of each agent (for example displaying specific properties of the k-nearest neighbours to the agent, or every data point within some radius). Sonification techniques like crystallisation (Hermann and Ritter [2005]), which perform local neighbourhood sonification would be quite suitable for such an interface. These sonifications or displays can combine display of the specific attributes of the data with display about the salience densities which they represent; the feedback shows both the local structure and the "directions" in which the agents can explore.

The structure laid out above demonstrates how the active selection approach can be applied to general exploration problems. The approach can provide opportunities for control expressed in terms of smoothly changing intentions, where the system reveals the motions which the interactor may use to express such intentions. The continuous, smoothly changing estimates of likelihoods of possible intention make this style of interaction possible. The flexibility of the technique potentiates the integrate the expression of the available control motions with feedback about the objective structure of a space.

VI.8 CONCLUSIONS

This chapter has outlined a coherent framework for continuous selection interfaces, where the spatio-temporal aspects of interaction can be dealt with in a way that utilises existing generative models of human behaviour. The interactions operate by injecting signals into the control loop and identifying selective filtering activity on the part of

the interactor. The methods are probabilistic by nature; the interface is viewed as a collection of agents who accumulate and evaluate evidence for the intention to effect the specific action with which they are associated. Prior model structures can be built into the interface directly. Communication rate is bounded by the properties of the channels that communicate intention, the quality of modelling and the quality of display. Spatial-metaphor-specific bounds (Fitts' law) do not apply in such contexts. Implementations have shown the that technique is workable, and is adaptable to different sensor contexts. The framework encompasses metaphors based upon control, imitation and excitation and hybrid combinations of these. The selection interfaces of the type described here can be improved by better operator modelling. The techniques presented here are extremely general and can be applied in a wide range of contexts.

CHAPTER VII

CONCLUSIONS

Summaries and the path ahead.

VII.1 SUMMARY

This chapter summarises the main points from each of the preceding chapters, drawing them together into the structure laid out in Chapter I.

VII.2 FRAMEWORKS FOR INTERACTION

This thesis has laid out a theoretical basis for the design of interactive systems. This framework encompasses conventional HCI while revealing a number of relatively unexplored areas. Concrete techniques for applying the concepts have been discussed, and working interfaces which embody these novel interaction concepts have been created using these techniques.

VII.2.1 THEORETICAL ASPECTS

The previous chapters have presented the continuous uncertain interaction framework, and details the major components of this framework: evidence display and evidence acquisition. Under this model, interaction is viewed as a continuous, hierarchical control problem, where hidden variables must be estimated across the interface boundary. Sensing provides *evidence* for goals, and is accumulated over time. Evidence is never certain, and models mapping sensing to intention are always inaccurate. The accumulation of evidence can be delayed and can happen at different rates on different channels. Uncertainty and delay are treated as key elements of interaction design in this framework. Feedback is a critical element, informing the user and reducing the complexity of the inference algorithms needed. The communication of intention is mediated by lower-level control loops with which the user engages via the physical embodiment of the interface.

The four aspects that have been tackled in this work are: the development of display techniques for uncertain information which inform the user about the true state of the system's beliefs; the extension of such displays to have predictive power; the design of changing dynamic systems which facilitate the flow of information by incorporating long term inference; and the construction of selection mechanisms which are probabilistic and directly include models of interactor behaviour. These form the major compo-

nents of the interface framework. Some aspects have not been considered, in particular the role of learning and adaptation – the outermost loops of the interaction process.

VII.2.2 Feedback, Uncertainty and Prediction

Chapters III and IV emphasised the importance of feedback with appropriate levels of uncertainty. Without this, decision making *cannot be rational*. Appropriately uncertain feedback regularises the behaviour of the user, and reduces the control effort they must apply when sensing is uncertain. Systems should display their beliefs about the intention of the user, and permit the user to *control* those states.

The sonification methods of Chapter III reveal the system's beliefs about an interactor as the interactor controls the system. Such feedback directly displays goal space trajectories as the user manipulates the interface. The inferential processes going on inside the system are revealed, along with the effect an interactor's actions have upon the beliefs of the system. Granular synthesis makes it easy to transform probabilistic models into rich audio feedback, with timbral, spatial and temporal flexibility. Point cloud displays or convolutions can be used in visual contexts. However, the abstractness of the auditory feedback makes it particularly suitable for display of uncertain belief. The flexible nature of the granular synthesis approach, and the ease with which it can be integrated with existing probabilistic models, makes it an attractive option for the augmentation of interfaces.

Delay has a critical effect on control, and human capacity to compensate for delay is limited. Incorporating prediction in displays can reduce the negative effect of delay upon interaction. The Monte Carlo prediction approaches of Chapter IV can be applied to many interaction contexts. Such displays can relate both the likely future states of the system, and the likely future states of the system's belief about user goals. Monte Carlo filtering processes are equally amenable to sonification and probabilistic visual display. These techniques facilitate displays that remain appropriately uncertain; where the evidence or model is poor the display can signal this clearly.

The displays can be applied in many contexts, from complex, unstable dynamic systems such as the helicopter control problem – where there is uncertainty in the evolution of the model but accurate sensing – to the slower-paced but noisy GPS navigation problem, where there is very significant uncertainty in the evidence acquired by sensors but reliable models of behaviour. Uncertain displays have been experimentally demonstrated to have a beneficial effect where noisy measurement subsists.

VII.2.3 Dynamics and Control

Chapters V and VI discussed the acquisition of evidence for intention, and in particular the design of dynamic systems which mediate the flow of information. These techniques seek to optimise the ability of a user to provide evidence for goals by enhancing the ability to control the intermediate dynamic system. Evidence from multiple timescales can be brought into the immediate interaction.

The augmented dynamics approach of Chapter V unifies enhanced pointing techniques, but it also motivates a large class of alternative approaches. Long-term inference can be brought into short-term control loops, bridging the gaps in the control loop hierarchy. Doing so can regularise the behaviour of an interactor, requiring less input energy when prior information is available. Control is not wrested from the hands of the user; it is augmented by the inferential engine. Local, tightly-coupled control can remain stable while slowly changing estimates of evidence improve the qualities of this loop. The

technique leads to useful systems; the Hex system is a functional text entry devices for mobile, inertial interaction.

Chapter VI introduced active selection methods. These are a viable alternative to conventional interactions. They can be directly designed to take into account the specific properties of communication channels, using manual control models of operator behaviour and including the characteristics of the sensing channels present. The methods are suitable for adaptation to unconventional sensing contexts where pointing metaphors are impractical or nonsensical. They are bounded by the quality of modelling, the power of the comparison models, and the information capacity of the input and output channels; they do not suffer the Fitts' bounds of spatial interfaces. Evidence for intention is accumulated gradually and rationally, and can be displayed to an interactor as the accumulation occurs. Interactors can thus control the system as evidence is acquired; decisions are made with appropriate opportunities for intervention, and can easily incorporate existing probabilistic models of user intention. The technique potentiates many interesting techniques for interaction where control is expressed in terms of intentions, mediated by agents who seek to gain evidence for such intention.

The overview figure of Chapter I (reproduced below, in Figure VII.1) shows how these components fit together into an architecture for the design of interaction. Techniques for displaying the changing states of the inference process have been given in the form of the granular synthesis and visual particle displays of Chapters III and IV. Chapter V showed how these changing estimates of intention can modulate the immediate world with which the user interacts. This increases the potential flow of information without disruption. The design of specific dynamic systems which can extract intention by revealing the ways in which they can be controlled was the focus of Chapter VI. This agent-based approach gives a concrete way of designing the short-term interaction. Together, these form a powerful basis for the construction of novel interfaces.

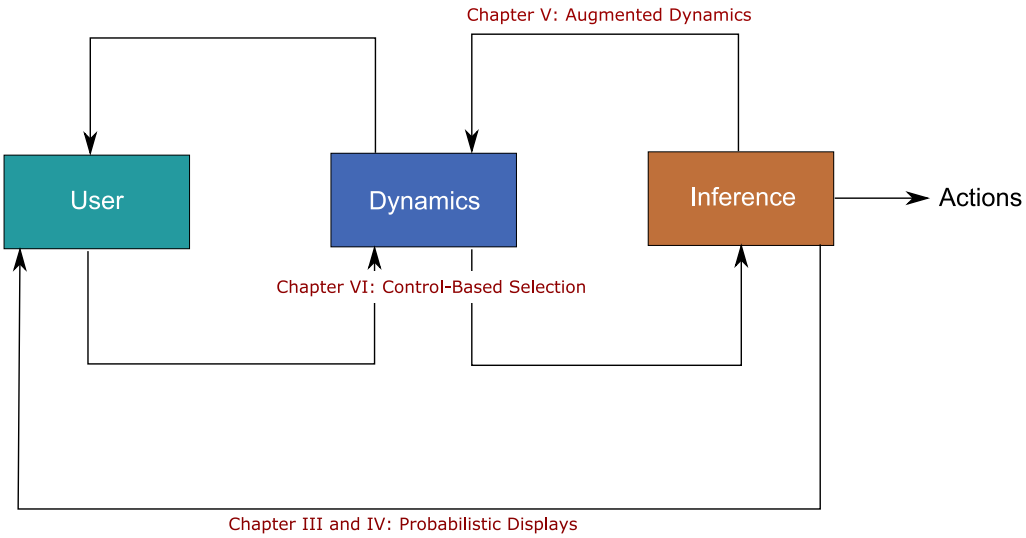


FIGURE VII.1: The overview of the interaction process, marked with the areas covered in this work.

The approach to interface design detailed in this work opens up a rich seam for fur-

ther exploration. Entirely new interactions can be built based upon the principles laid out, featuring continuous control across multiple timescales with rich, representative, multimodal feedback. New sensing channels or contexts of use can be brought in to the interactions easily. Sophisticated inferential tools can be introduced into the interaction without *ad hoc* alterations. Interactions designed according to these principles have strong theoretical underpinnings and can be analysed and designed coherently.

APPENDIX A

GSLIB – THE PROBABILISTIC GRANULAR SYNTHESIS LIBRARY

GSLIB, the granular synthesis library developed for the implementation of demonstrations in this thesis, provides a convenient way of sonifiying probability distributions. It is a pure C granular synthesis implementation, heavily optimized for speed. The accumulation functions operate in fixed point (important for mobile devices which often lack FPU's) with a packed representation to partially parellize the computations. The basic version depends on the availabilty of the Allegro[1] library, but is otherwise platform independent. Any sound library which can provide a series of buffers to fill for audio output can easily be adapted to work with the library (for example, SDL, DirectX or the Windows MMSystem routines). The library code has has been used succesfully on Windows 98/XP/2000 (with Allegro) and Windows CE (ARM), running with the standard Microsoft MMSystem audio output. Since Allegro is available on Linux, Mac OS X, QNX and Beos and many other UNIX platforms, the library should compile without issue on most machines.

A.1 PARAMETERS

The library renders audio in buffers, of a certain specified length. These buffers are filled with the grains from the synthesis process, with the appropriate transformations. More details on the granular synthesis process and its parameters are given in Chapter III. Specification of the audio is given via PDF's for the parameter distributions. These parameters are:

- **Source**: Discrete distribution over wave form sources that have been registered with the library.
- **Time**: Continuous distribution over samples inside the source wave form.
- **Pitch**: Continuous distribution over pitches of grains.
- **Pan**: Continuous distribution over stereo pan position of grains.

The system also has a global grain density, grain length, maximum number of grains, and parameters for adjusting the grain envelope. These are fixed and are not defined by distributions.

These parameters can be split into two groups: initialize time and run-time parameters. Initialize time parameters must be set before synthesis begins, and cannot be modified

[1] http://www.talula.demon.co.uk/allegro

during synthesis. Run-time parameters are updated by callbacks which are called every time the (sub)buffer is filled; these specify the time-varying properties of the synthesis.

A.1.1 INITIALIZE TIME PARAMETERS

At initialize time, the following parameters must be set:

- Grain length. Length of grains in *samples*.
- Envelope type. GSLib supports two envelope types, Gaussian and linear (see Figure III.4.2.2). For the Gaussian type, the width of the kernel can be set; for the linear type the attack/decay time as proportions of the grain length are given.
- Maximum number of grains. This fixs the maximum simulatenous number of grains.

These parameters are initialize time to provide significant optimizations to the granulation process, especially envelope precalculation.

A.1.2 RUN TIME PARAMETERS

The density of the grains (as a probability of activation at a time step) can be specified at run time, as can all of the basic parameters described in Section A.1. For discrete distributions, the parameters are specified as individual probabilities. For the continuous parameters, Gaussian, half-Gaussian ($f(x) > 0 = 0$ or $f(x) < 0 = 0$) and delta functions can be used, as well as mixtures of these.[2]

A.1.3 MONTE CARLO OR ROUND-ROBIN MODE

The library can operate in normal (Monte Carlo) mode where parameters are automatically drawn from the parameter distrubutions. Alternatively, the parameters can be set individually on each grain generation (round robin mode) for using custom distributions. This makes implementing the type of display in Chapter IV very straightforward.

[2] Although types cannot be mixed, so a Gaussian and delta mixture cannot be created, for example.

A.2 Annotated Example

```
//Compiles correctly with MINGW GCC 3.4.2 on Windows
//gcc -Wall granular-example.c -o granular-example grain_source.c \
//grain_engine.c pdf.c -lalleg
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>
#include <assert.h>
#include "grain_engine.h"

// Allegro [http://www.talula.demon.co.uk/allegro] for audio i/o
#include "allegro.h"

//Output sample rate
#define SAMPLE_RATE 44100

//Output buffer size. Shorter has less latency,
//but can lead to clicking artifacts when the buffer underruns
#define BUFSIZE 4096

//Allegro's output audio stream structure
static AUDIOSTREAM *out_stream;

//Wave files to used
static char *wave_names[] =
{"sh5.wav",
"choir1.wav",
"bass2.wav"};

static int n_waves = 3; // Number of wave files


// Load the waveforms
void load_waveforms ()
{
  SAMPLE *samp;
  //defines the properties of a grain generator
  grain_source *source;
  int i;


  for (i = 0; i < n_waves; i++)
    {
      // Read each wave and put it in a grain_source structure
      samp = load_sample (wave_names[i]); // Allegro function

      if(samp==NULL)
      {
        printf("Sample %s missing\n", wave_names[i]);
```

```
        exit(-1);
      }
      source = create_uniform_source (i, samp->data, samp->len);
      // Register the source with the engine (important!)
      add_source (source);
    }
}

//Audio generation loop
void play_audio ()
{
  unsigned char *buf, *dbl_buf;
  int copied = 0;

  // Create a double buffer (16 bit = 2 bytes for each sample)
  dbl_buf = malloc(BUFSIZE*2);

  //Start the audio stream, 16 bit mono stream at SAMPLE_RATE
  out_stream =
    play_audio_stream (BUFSIZE, 16, FALSE, SAMPLE_RATE, 255, 128);

  //Set the sub-buffer length (callback is called for each one)
  set_buffer_length (128);

  //This loop can also be implemented in a multi-threaded manner

  //Check keyboard state, exit when the user presses escape
  while (key[KEY_ESC] == 0)
    {
          //Get next audio buffer (if one's available)
          buf = get_audio_stream_buffer (out_stream);

          //If no buffer available
          if (buf == NULL)
          {

            //If new buffer not already created
            if(!copied)
             {
              //Call the synthesis algorithm
              fill_buffer (dbl_buf, BUFSIZE * 2 );
              copied = 1;
            }
            else
              rest(1); // Wait a millisecond
}
        else // output buffer ready
        {
          // Copy double buffer into output buffer
          memcpy(buf, dbl_buf, BUFSIZE * 2 );
```

```
          copied = 0;
          //Mark the buffer as ready to play
          free_audio_stream_buffer (out_stream);
        }
    }

  //Stop the audio
  stop_audio_stream(out_stream);
}

//This is the callback that updates the properties of a source

double update_source_pdf (grain_source * source)
{
  static double phase = 0.0;
  double p, pitch_width;

  // Probabilities here are simple sinusiodal functions
  //  phase determined by their ID
  p = fabs(sin(phase+source->id))*0.3;
  phase+=1e-4; // Update the phase

  // Set the probability.
  //THIS MUST BE CALLED on every update_source_pdf
  set_probability (source, p);

  //optional parameters
  //Set the pitch to a narrow gaussian about its natural rate.
  //Width of the Gaussian changes as a cyclic function
  //This produces a dissonant sound slowly resolving to clear tones
  //and then back to dissonance
  pitch_width = fabs(cos(phase*0.2+source->id)) * 1e-1;
  set_pitch_distribution(source,1.0,pitch_width, D_GAUSSIAN);

  //Set the phase distribution to start at phase*10000
  //with 500 sample half-Gaussian (negative only) distribution
  //this causes the sounds to "play" forwards very slowly.
  set_distribution(source, phase*10000, 500, D_GAUSSIAN_BACKWARD);

  //Return the probability this source was assigned
  return p;
}

void init_grains ()
{
  //Set sample rate, bit depth and channels
  set_format (SAMPLE_RATE , 16, 1);

  //Maximum grains (this must be set)
  set_max_grains(200);
```

```
  //P(generation) = 0.4
  set_grain_density(0.4);

  //Set the grain length to 4000 samples (~100ms)
  set_grain_length(4000);

  //Don't normalize the probabilities
  set_normalize(0);

  //Squared Exp. envelope, width 0.1
  set_envelope(ENV_EXP, 0.1);

  //Register the callback
  register_distribution_function (update_source_pdf);

  //Initialize the engine
  initialize_engine ();

}

int main (int argc, char *argv[])
{
  //Open  the allegro subsystem
  allegro_init ();

  //Install the sound driver (if possible)
  if (install_sound (DIGI_AUTODETECT, MIDI_NONE, NULL) != 0)
    {
      printf ("Could not initialize sound driver...\n");
      exit (-1);
    }

  //Init RNG
  srand (time (NULL));

  //Install keyboard driver.
  install_keyboard ();

  //Initialise the engine
  //must be done _BEFORE_ loading the waveforms
  init_grains ();

  //Load the wave data
  load_waveforms ();

  //Start the audio playback loop
  play_audio ();

  //Close down the allegro subsystem
  allegro_exit ();
  return EXIT_SUCCESS;
```

```
}
END_OF_MAIN ();
```

The sound output from this example is available online at the GSLib homepage (see below).

A.3 LIMITATIONS OF THE LIBRARY

There are some minor limitations of the synthesis library: the library cannot deal with run-time adjustment of grain length; envelopes can be adjusted at run-time but not on a per-grain basis; and pitch shifting is implemented with linear interpolation, and can produce aliasing and interpolation noise at large pitch shifts. These are unlikely to be of much consequence in most applications.

A.4 OBTAINING AND COMPILING THE LIBRARY

The library is available at `http://www.dcs.gla.ac.uk/~jhw/thesis/gslib.html`. To use in a project, simply include `grain_engine.c`, `grain_source.c` and `pdf.c` into the build. Register the callbacks and initialize the parameters as described above, and then pass `fill_buffer()` a new buffer every time one must be filled.

APPENDIX B

GUIDELINES FOR DESIGN

GUIDELINES for design, aimed at designers building systems according to the principles laid out in the thesis, are presented in this chapter. The major points of the thesis are condensed into these guidelines.

- Where uncertainty is present in an interactive system, the full uncertainty should be preserved as far as possible and the inference done on the complete distribution of potential values when an action must be performed. Early, irreversible, filtering of the values to sequentialise the process should be avoided.
- The utility and likelihood of the goals a user interface provides should be explicitly accounted for. These can either be used directly to formulate design, or used to make decisions about the structure of the interface (e.g. ranking interface elements by expected value).
- Uncertainty in an interface should be displayed to the user wherever it is relevant; doing so can regularise their behaviour in a natural manner. Granular synthesis is a powerful way of forming such audio displays. Point cloud displays can be used for visual display.
- Designers should take careful note of delays that will be present in interacting with a system and should employ predictive displays to mitigate the effect of such delays, wherever possible. Such displays should taken into account the certainty of the model and the sensor readings upon which they are based.
- When introducing "intelligence" into an interface, the flow of the underlying control process should not be disrupted. Creating a stable dynamic system which the user interacts with – whose parameters are influenced by the higher level inference – can be used to achieve this. The user should be able to model the response of system well at short time scales, with the ability to model dropping off as a function of the communication bandwidth of the interface.
- Designers should ensure that decisions are only made when sufficient evidence has accumulated to support them, and that the arrival of such evidence does not violate the input bandwidth limitations in the joint human-computer loop. Systems which fail to do this degrade poorly with reductions in control quality.
- Where asymmetric control loops exist (with system output bandwidth exceeding input bandwidth) selection can be performed by providing statistically independent stimuli, and testing for behaviour compatible with controlling those stimuli; this is applicable to many sensor and display types.

APPENDIX C

ONLINE MATERIALS

ONLINE materials which accompany this thesis are briefly listed below. The up-to-date index, is available at `http://www.dcs.gla.ac.uk/~jhw/thesis/`. These materials include demos (suitable for desktop Windows machines), videos, audio and pictures of the various implementation demonstrations in action. The electronic version of this thesis, with hyperlinked crossreferences, is also available. The granular synthesis library code GSLib can also be obtained; see Appendix A for details.

C.1 MATERIALS

C.1.1 CHAPTER III

- **Gaussian Spatial Example**
  Demo, sound files and video of the implementation.
- **Trajectory Following Example**
  Demo, sound files and video of the implementation.

C.1.2 CHAPTER IV

- **Ball Bearing Example with Uncertain Predictive Feedback**
  Video of the implementation.
- **GPS Navigation Demo**
  Video of system mock up on desktop machine. Video of application running on PocketPC device.

C.1.3 CHAPTER V

- **Hex**
  Video of Hex in use on both desktop and PocketPC.
- **BCI Hex**
  A video of an early version of Hex adapted for use in BCI.

C.1.4 CHAPTER VI

- **Pointing without a Pointer**
  Video and demo of the trajectory following selection example.
- **Egghead Selection**
  Video of the Egghead Selection Demo.

C.1.5 CHAPTER A

- **GSLib demo**
  Audio generated from the example source code.

INDEX

BIBLIOGRAPHY

J. Accot and S. Zhai. More than dotting the i's — foundations for crossing-based interfaces. In *CHI '02: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 73–80, New York, NY, USA, 2002. ACM Press.

J. Accot and S. Zhai. Beyond fitts' law: models for trajectory-based hci tasks. In *CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 295–302, New York, NY, USA, 1997. ACM Press.

C. Asakawa, H. Takagi, S. Ino, and T. Ifukube. Maximum listening speeds for the blind. In *International Community for Auditory Display*, pages 276–279, 2003.

T. Asano, E. Sharlin, Y. Kitamura, K. Takashima, and F. Kishino. Predictive interaction using the delphian desktop. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 133–141, New York, NY, USA, 2005. ACM Press.

W. R. Ashby. *Introduction to Cybernetics*. Methuen, 1956.

R. Balakrishnan and I. S. MacKenzie. Performance differences in the fingers, wrist, and forearm in computer input control. In *Proceedings of the CHI '97 Conference on Human Factors in Computing Systems*, pages 303–310. ACM Press, 1997.

J. Bardram and O. W. Bertelsen. Supporting the development of transparent interaction. In *EWCHI '95: Selected papers from the 5th International Conference on Human-Computer Interaction*, pages 79–90, London, UK, 1995. Springer-Verlag. ISBN 3-540-60614-9.

R. C. Barrett, E. J. Selker, J. D. Rutledge, and R. S. Olyha. Negative inertia: a dynamic pointing function. In *CHI '95: Conference companion on Human factors in computing systems*, pages 316–317, 1995.

M. Barth, T. Burkert, C. Eberst, N.O. Stffler, and G. Frber. Photo-realistic scene prediction of partially unknown environments for the compensation of time delays in presence applications. In *IEEE Int. Conf. on Robotics and Automation, ICRA*, pages 3132–31372, 2000.

D. Beamish, S. A. Bhatti, I. S. MacKenzie, and J. Wu. Fifty years later: A neurodynamic explanation of fitts' law. *Journal of the Royal Society Interface.*, April 2006.

A. K. Bejczy, S. Venema, and W. S. Kim. Role of computer graphics in space telerobotics: Preview and predictive displays. *Cooperative Intelligent Robotics in Space*, SPIE Volume 1387:365–377, 1990.

T. Bell, J. Cleary, and I. Witten. Data compression using adaptive coding and partial string matching. *IEEE Transactions on Communications*, 32(4):396–402, 1984.

A. Beuter, A. Glass, C. Mackey, and M. Titcombe. *Nonlinear Dynamics in Physiology and Medicine*. Springer, New York, 2nd edition, 2003.

H. P. Birmingham and F. V. Taylor. A design philosophy for man-machine control systems. In *Proceedings of the Institute of Radio Engineers*, volume 42, pages 1748–1758, 1954.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

M. J. Black and A. D. Jepson. A probabilistic framework for matching temporal trajectories: Condensation-based recognition of gestures and expressions. In H. Burkhardt and B. Neumann, editors, *European Conf. on Computer Vision, ECCV-98*, volume 1406 of *LNCS-Series*, pages 909–924, Freiburg, Germany, 1998. Springer-Verlag.

R. Blanch. *Architecture logicielle et outils pour les interfaces hommes-machines graphiques avances (Software architecture and tools for advanced computer-human graphic interaction)*. PhD thesis, Universit Paris XI, Orsay, 2005.

R. Blanch, Y. Guiard, and M. Beaudouin-Lafon. Semantic pointing: improving target acquisition with control-display ratio adaptation. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 519–526, New York, NY, USA, 2004. ACM Press.

B. Blankertz, C. Schfer, G. Dornhege, and C. Gabriel. Single trial detection of eeg error potentials: A tool for increasing BCI transmission rates. In *Artificial Neural Networks – ICANN 2002*, pages 1137–1143, 2002.

B. Blankertz, G. Dornhege, M. Krauledat, M. Schroder, J. Williamson, R. Murray-Smith, and K-R. Muller. The Berlin brain-computer interface presents the novel mental typewriter hex-o-spell. In *3rd International BCI Workshop and Training Course*, 2006.

K. R. Boff and J. E. Lincoln. Engineering data compendium: Human perception and performance. Technical Report 3, 1988.

P. Bowcott. Cellular automation as a means of high level compositional control of granular synthesis. In *ICMC Proceedings*, pages 55–57. ICMA, 1989.

S. Brewster. Nonspeech auditory output. In *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications*, pages 220–239. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, USA, 2003.

S. A. Brewster. Using non-speech sound to overcome information overload. *Displays*, 17:179–189, 1997.

S. A. Brewster, P.C. Wright, and A.D.N. Edwards. An evaluation of earcons for use in auditory human-computer interfaces. In S. Ashlund, K. Mullet, A. Hendersona, E. Hollnagel, and T. White, editors, *InterCHI'93*, pages 222–227, 1993.

P. F. Brown, V. J. D. Pietra, R. L. Mercer, S. A. D. Pietra, and J. C. Lai. An estimate of an upper bound for the entropy of English. *Comput. Linguist.*, 18(1):31–40, 1992.

R. Brown. Animated visual vibrations as an uncertainty visualisation technique. In *GRAPHITE '04: Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 84–89, New York, NY, USA, 2004. ACM Press.

S. T. Bryson. Effects of lag and frame rate on various tracking tasks. In *Stereoscopic Displays and Applications IV*, volume SPIE 1915, pages 155–166. Springer, 1993.

T. Burkert, J. Leupold, and G. Passig. A photorealistic predictive display. *Presence: Teleoper. Virtual Environ.*, 13(1):22–43, 2004.

F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal. *Pattern-Oriented Software Architecture*. John Wiley and Sons, 1996. ISBN 0-471-95869-7.

B. Buxton. Human input to computer systems: Theories, techniques and technology. URL `http://www.billbuxton.com/inputManuscript.html`. Draft available online, 2002.

W. Buxton. Integrating the periphery and context: A new taxonomy of telematics. In *Graphics Interface '95*, pages 239–246, 1995.

S. K. Card, W. K. English, and B. J. Burr. Evaluation of mouse, rate-controlled joystick, step keys, and text keys for text selection on a CRT. *Ergonomics*, 21:601–613, 1978.

S. K. Card, A. Newell, and T. P. Moran. *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, USA, 1983.

D. Chapman and C. Ware. Manipulating the future: Predictor based feedback for velocity control in virtual environment navigation. In D. Zeltzer, editor, *In Computer Graphics (1992 Symposium on Interactive 3D Graphics)*, volume 25, pages 63–66, 1992.

E. Childs. Achorripsis: A sonification of probability distributions. In *ICAD 2002*, 2002.

J. Cleary, W. Teahan, and I. Witten. Unbounded length contexts for PPM. In *DCC-95*, pages 52–61. IEEE Computer Society Press, 1995.

P. R. Cook. *Real Sound Synthesis*. A K Peters, 2002.

E. L. Corliss. Estimate of the inherent channel capacity of the ear. *Journal of the Acoustical Society of America*, 50(2):671–677, 1971.

R. G. Costello. The surge model of the well-trained human operator in simple manual control. *IEEE Transactions on Man–Machine Systems*, 9(1), 1968.

T. Cover and R. King. A convergent gambling estimate of the entropy of English. *IEEE Transactions on Information Theory*, 24(4):413–421, 1978.

R. T. Cox. *The algebra of probable inference*. Johns Hopkins Press, 1961.

A. Crossan and R. Murray-Smith. Variability in wrist-tilt accelerometer-based gesture interfaces. In *Mobile Human-Computer Interaction MobileHCI 2004 LNCS 3160*, pages 144–155, 2004.

A. Crossan, J. Williamson, and R. Murray-Smith. Haptic granular synthesis: Targeting, visualisation and texturing. In *International Symposium on Non-visual and Multimodal Visualization*, pages p527–532. IEEE Computer Society, 2004.

D. W. Cunningham, A. Chatziastros, M. von der Heyde, and H. H. Blthoff. Driving in the future: Temporal visuomotor adaptation and generalization. *J. Vis.*, 1(2):88–98, 10 2001. ISSN 1534-7362.

J. Decety and M. Jeannerod. Mentally simulated movemnts in virtual reality: Does fitts' law hold in motor imagery? *Behavioural Brain Research*, 72:127–134, 1996.

A. Degani. The grounding of the royal majesty. In A. Degani, editor, *Taming HAL: Designing Interfaces Beyond 2001*, pages 100–121. Palgrave Macmillan, 2004.

D. Dennett. *Elbow Room: The Varieties of Free Will Worth Wanting*. MIT Press, 1984.

A. Dix. Beyond intention: pushing boundaries with incidental interaction. In *Building Bridges: Interdisciplinary Context-Sensitive Computing*, pages 1–6, 2002.

A. Dix, J. Finlay, G. Abowd, and R. Beale. *Human-Computer Interaction*. Prentice-Hall, 1993.

G. Doherty and M. Massink. Continuous interaction and human control. In *European Conference on Human Decision Making and Manual Control*, pages 80–96, 1999.

G. Doherty, T. Anderson, W. Wilson, and G. Faconti. A control-centred approach to designing interaction with novel devices. In *In Conference on Universal Access in Human Computer Interaction*, pages 286–290. Lawrence Erlbaum Associates, Inc., 2001.

S. M. Doherty and C. D. Wickens. Effects of preview, prediction, frame of reference, and display gain in tunnel-in-the-sky displays. In *Proceedings of the 11th International Symposium on Aviation Psychology.*, 2001.

A. Doucet, N. Freitas, and N. Gordon. *Sequential Monte Carlo in Practice*. Springer-Verlag, 2001.

P. Dourish. What we talk about when we talk about context. *Personal Ubiquitous Comput.*, 8(1):19–30, 2004.

M. D. Dunlop. Watch-top text-entry: Can phone-style predictive text-entry work with only 5 buttons? In *Mobile HCI 2004*, volume 3160 of *Lecture Notes in Computer Science*, pages 342–346, 2004.

S. R. Ellis, M. J. Young, B. D. Adelstein, and S. M. Ehrlich. Discrimination of changes in latency during head movement. In *Proceedings of the HCI International '99*, pages 1129–1133. Lawrence Erlbaum Associates, Inc., 1999.

W. H. B Ellis, A. Burrows, and K. F. Jackson. Presentation of air speed while deck-landing: comparsion of visual and auditory methods. Technical Report 841, UK RAF Flying Personnel Research Committee, 1953.

P. Eslambolchilar and R. Murray-Smith. Tilt-based automatic zooming and scaling in mobile devices – a state-space implementation. In S. Brewster and M. Dunlop, editors, *Mobile HCI 2004*, volume 3160, pages 120–131. Springer LNCS, September 2004.

P. Eslambolchilar, J. Williamson, and R. Murray-Smith. Multimodal feedback for tilt controlled speed dependent automatic zooming. In *UIST 2004*, 2004.

G. Essl and S. O'Modhrain. Scrubber: An interface for friction-induced sounds. In *NIME'05*, pages 70–75, 2005.

S. K. Feiner and Clifford Beshers. Worlds within worlds: metaphors for exploring n-dimensional virtual worlds. In *UIST '90: Proceedings of the 3rd annual ACM SIGGRAPH symposium on User interface software and technology*, pages 76–83, New York, NY, USA, 1990. ACM Press.

P. W. Ferrez and J. del R. Millán. You are wrong!—automatic detection of interaction errors from brain waves. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pages 1413–1419, Edinburgh, UK, August 2005.

P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47(6):381–391, 1954.

T. Flash and N. Hogan. The coordination of arm movements: an experimentally confirmed mathematical model. *Journal of Neuroscience*, 5(7):1688–1703, 1985.

T. W. Forbes. Auditory signals for instrument flying. *Journal of Aeronautical Science*, 13: 255–258, 1946.

C. Forlines, C. Shen, and B. Buxton. Glimpse: a novel input model for multi-level devices. In *CHI '05*, pages 1375–1378. ACM Press, 2005.

G. W. Furnas. Generalized fisheye views. In *CHI '86*, pages 16–23, 1986.

D. Gabor. Acoustical quanta and the theory of hearing. *Nature*, 159(4044):591–594, 1947.

B. R. Gaines. Linear and nonlinear models of the human controller. *International Journal of Man-Machine Studies*, 1:333–360, 1969.

Gamma, Helm, and Johnsonand Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley, 1995.

W. Gaver. Auditory icons: using sound in computer interfaces. *Human Computer Interaction*, 2:167–177, 1986.

A. Girard. *Approximate MethodsforPropagation of Uncertainty with Gaussian Process Models*. PhD thesis, Dept. Computing Science, University of Glasgow, 2004.

M. Goodchild, L. Chih-Chang, and Y. Leung. Visualizing fuzzy maps. In *Visualization in Geographical Information Systems*, pages 158–167. John Wiley & Sons, 1994.

H. Griethe and H. Schumann. The visualization of uncertain data: Methods and problems. In *SimVis*, pages 143–156, 2006.

G. Grigoryan. Point-based probabilistic surfaces to show surface uncertainty. *IEEE Transactions on Visualization and Computer Graphics*, 10(5):564–573, 2004.

T. Grossman and R. Balakrishnan. The bubble cursor: Enhancing target acquisition by dynamic resizing of the cursors activation area. In *CHI 2005*, pages 281–290, 2005a.

T. Grossman and R. Balakrishnan. A probabilistic approach to modeling two-dimensional pointing. *ACM Trans. Comput.-Hum. Interact.*, 12(3):435–459, 2005b.

T. Grossman, R. Balakrishnan, and K. Singh. An interface for creating and manipulating curves using a high degree-of-freedom curve input device. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 185–192, New York, NY, USA, 2003. ACM Press.

Y. Guiard and M. Beaudouin-Lafon. Special issue: Fitts law 50 years later: Applications and contributions from human-computer interaction. *Internation Journal of Human-Computer Studies*, 61(6), 2004a.

Y. Guiard and M. Beaudouin-Lafon. Target acquisition in multiscale electronic worlds. *International Journal of Human-Computer Studies*, 61(6):875–905, 2004b.

Y. Guiard, R. Blanch, and M. Beaudouin-Lafon. Object pointing: a complement to bitmap pointing in GUIs. In *GI '04: Proceedings of the 2004 conference on Graphics interface*, 2004.

B. L. Harrison, K. P. Fishkin, A. Gujar, C. Mochon, and R. Want. Squeeze me, hold me, title me!: An exploration of manipulative user interfaces. In *CHI'98*, pages 17–24, 1998.

T. Hermann and A. Hunt. Guest editors' introduction: An introduction to interactive sonification. *IEEE MultiMedia*, 12(2):20–24, 2005.

T. Hermann and H. Ritter. Crystallization sonification of high-dimensional datasets. *ACM Transactions on Applied Perception*, 2(4):550–558, 2005.

T. Hermann and H. Ritter. Listen to your data: Model-based sonification for data analysis. In M. R. Syed, editor, *Advances in intelligent computing and mulimedia systems.*, pages 189–194. Int. Inst. for Advanced Studies in System Research and Cybernetics,, 1999.

T. Hermann, P. Meinicke, and H. Ritter. Principal curve sonification. In *International Conference on Auditory Display*, pages 81–86, 2000.

R. A. Hess and P.J. Gorder. Design and evaluation of a cockpit display for hovering flight. *J. Guidance, Control and Dynamics*, 13:450–457, 1990.

K. Hinckley. Input technologies and techniques. In *Handbook of Human-Computer Interaction*, pages 151–168. Lawerence Erlbaum Associates, 2002.

K. Hinckley, J. Pierce, M. Sinclair, and E. Horvitz. Sensing techinques for mobile interaction. In *UIST'2000*, pages 91–100, 2000.

K. Hinckley, J. Pierce, E. Horvitz, and M. Sinclair. Foreground and background interaction with sensor-enhanced mobile devices. *ACM Trans. Comput.-Hum. Interact.*, 12(1): 31–52, 2005.

E. R. Hoffmann, K. K. Tsang, and A. Mu. Data-entry keyboard geometry and keying movement times. *Ergonomics*, 38:940–950, 1995.

S. J. Hope. Decision-making under spatial uncertainty. Master's thesis, Department of Geomatics, University of Melbourne, 2005.

E. Horvitz and G. Rutledge. Time-dependent utility and action under uncertainty. In *Seventh Conference on Uncertainty in Artificial Intelligence*, pages 151–158. Morgan Kaufman, 1991.

Y. How and M.-Y. Kan. Optimizing predictive text entry for short message service on mobile phones. In *Proc. of Human Computer Interfaces International (HCII 05)*, 2005.

S. Hughes, I. Oakley, and S. O'Modhrain. Mesh: Supporting mobile multi-modal interfaces. In *UIST 2004*. ACM, 2004.

A. Inselberg. The plane with parallel coordinates. *The Visual Computer*, 1:69–91, 1985.

M. Isard and A. Blake. A smoothing filter for condensation. In *Proceedings of the 5th European Conference on Computer Vision*, volume 1, pages 767–781, 1998a.

M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998b.

P. Isokoski and R. Raisamo. Speed and accuracy of six mice. *Asian Information-Science-Life*, 2(2):131–140, 2004.

T. Itagaki. Sound compression/interpolation by granulation. In *108th Audio Engineering Society Convention*, 2000.

R. J. Jagacinski and J. M. Flach. *Control theory for humans : quantitative approaches to modeling performance*. L. Erlbaum Associates, Mahwah, N.J., 2003.

R. J. Jagacinski and D. L. Monk. Fitts' law in two dimensions with hand and head movements. *Journal of Motor Behavior*, 17:77–95, 1985.

C. L. James and K. M. Reischel. Text input for mobile devices: comparing model prediction to actual performance. In *CHI '01: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 365–371, New York, NY, USA, 2001. ACM Press.

I. Jang and W. Park. Gesture-based user interfaces for handheld devices using accelerometer. *Lecture Notes in Computer Science*, 3331:359–368, 2004.

E.T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, 1998.

G. Johannsen. Special issue on engineering and music: Supervisory control and auditory communication. *Proceedings of the IEEE*, 92(4), 2004.

C.R. Johnson and A.R. Sanderson. A next step: Visualizing errors and uncertainty. *IEEE Computer Graphics and Applications*, 23(5):6–10, 2003.

P. Kabbash and W. Buxton. The "prince" technique: Fitts' law and selection using area cursors. In *ACM CHI Conference on Human Factors in Computing Systems*, pages 273–279, 1995.

R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

H. Kantz and T. Schreiber. *Nonlinear Time Series Analysis*. Cambridge University Press, 2nd edition, 2003.

C. R. Kelley. *Manual and Automatic Control: A Theory of Manual Control and Its Applications to Manual and to Automatic Systems*. Academic Press, 1968.

D. Kelly. Information capacity of a single retinal channel. *IEEE Transactions on Information Theory*, 8(3):221–226, 1962.

A. Kim. *Development of Sensor Fusion Algorithms for MEMS-Based Strapdown Inertial Navigation Systems*. PhD thesis, University of Waterloo, 2004.

S. Kirpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

K. P. Kording and D Wolpert. Bayesian integration in sensorimotor learning. *Nature*, 427:244–247, 2004.

G. Kramer and B. Walker. *Sonification report: Status of the field and research agenda*. International Community for Auditory Display, 1999.

P.-O. Kristensson and S. Zhai. Shark2: A large vocabulary shorthand writing system for pen-based computers. In *17th Annual ACM Symposium on User Interface Software and Technology (UIST 2004)*, pages 43–52. ACM Press, 2004.

K. J. Kuchenbecker, J. Fiene, and G. Niemeyer. Improving contact realism through event-based haptic feedback. *IEEE Transactions on Visualization and Computer Graphics*, 12(2): 219–230, 2006.

G. D. Langolf, D. B. Chaffin, and J. A. Foulke. An investigation of Fitts' law using a wide range of movement amplitudes. *Journal of Motor Behaviour*, 8:113–128, 1976.

E. Lank and E. Saund. Sloppy selection: Providing an accurate interpretation of imprecise selection gestures. *Computers and Graphics*, 29:490–500, 2005.

V. Lantz and R. Murray-Smith. Rhythmic interaction with a mobile device. In *NordiCHI '04, Tampere, Finland*, pages 97–100. ACM, October 23-27 2004.

M. Latash. *Control of human movement*. Human Kinetics, 1993.

A. Lécuyer, S. Coquillart, A. Kheddar, P. Richard, and P. Coiffet. Pseudo-haptic feedback : Can isometric input devices simulate force feedback? In *IEEE International Conference on Virtual Reality*, pages 83–90, 2000.

A. Lécuyer, J.M. Burkhardt, S. Coquillart, and P. Coiffet. "boundary of illusion" : an experiment of sensory integration with a pseudo-haptic system. In *IEEE International Conference on Virtual Reality*, pages 115–122, 2001.

A. Lécuyer, J.M. Burkhardt, and L. Etienne. Feeling bumps and holes without a haptic interface: the perception of pseudo-haptic textures. In *CHI 2004*, pages 239–246, 2004.

M. Leitner and B.P. Buttenfield. Cartographic guidelines for visualizing attribute accuracy. In *Auto-Carto 13*, pages 84–193, 1997.

B. Libet. *Mind Time*. Harvard University Press, 2004.

B. Libet. The experimental evidence for subjective of a sensory experience backwards in time. *Philosophy of Science*, 48:182–197, 1981.

F. Lorussi, E. P. Scilingo, M. Tesconi, A. Tognetti, and D. De Rossi. Strain sensing fabric for hand posture and gesture monitoring. *IEEE Transactions on Information Technology in BioMedicine*, 9(3):372–381, September 2005.

H. J. Luinge. *Inertial Sensing of Human Movement*. PhD thesis, University of Twente, 2002.

A. M. MacEachren, A. Robinson, S. Hopper, S. Gardner, R. Murray, and E. Gahegan, M.and Hetzler. Visualizing geospatial information uncertainty: What we know and what we need to know. *Cartography and Geographic Information Science*, 32(3):139–160, 2005.

D. J. C. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

I. S. Mackenzie. *Fitts' Law as a Performance Model in Human-Computer Interaction*. PhD thesis, University of Toronto: Toronto, 1991.

I. S MacKenzie. Input devices and interaction techniques for advanced computing. In *Virtual environments and advanced interface design*, pages 437–470. Oxford University Press, 1995.

I. S. MacKenzie and C. Ware. Lag as a deteriminant of human performance in interactive systems. In *InterCHI'93*, pages 488–493, 1993.

S. MacKenzie and W. Buxton. Extending Fitts' law to two-dimensional tasks. In *Proceedings of CHI '92*, pages 219–226, 1992.

S. I. MacKenzie and R. W. Soukoreff. Text entry for mobile computing: Models and methods,theory and practice. *Human-Computer Interaction*, 17:147–198, 2002.

T. Maki-Patola and P. Hamalainen. Latency tolerance for gesture controlled continuous sound instrument without tactile feedback. In *Proc. International Computer Music Conference (ICMC)*, 2004.

J. Mankoff. *An architecture and interaction techniques for handling ambiguity in recognition-based input.* PhD thesis, Georgia Institute of Technology, 2001.

J. Mankoff and G. D. Abowd. Cirrin: A word-level unistroke keyboard for pen input. In *UIST '98*, pages 213–214. ACM Press, 1998.

J. Mankoff, S. E. Hudson, and G. D. Abowd. Interaction techniques for ambiguity resolution in recognition-based interfaces. In *UIST*, pages 11–20, 2000.

G. Marentakis and S. A. Brewster. Gesture interaction with spatial audio displays: Effects of target size and inter-target separation. In *ICAD2005*, pages 77–84. ICAD, 2005.

R. S. Marken. *More Mind Readings: Methods and Models in the Study of Purpose*. newview, 2002.

R. S. Marken. *Mind Readings: Experimental Studies of Purpose*. The Control Systems Group Book, 1995.

P. S. Maybeck. *Stochastic models, estimation, and control*, volume 141 of *Mathematics in Science and Engineering*. Academic Press, 1979.

A. Mazzoldi, D. De Rossi, F. Lorussi, E. P. Scilingo, and R. Paradiso. Smart textiles for wearable motion capture systems. *AUTEX Research Journal*, 2(4):199–203, 2002.

RC Miall and JK. Jackson. Adaptation to visual feedback delays in manual tracking: evidence against the smith predictor model of human visually guided action. *Exp Brain Res.*, 172(1):77–84, 2006.

N. Milnes-Walker. A study of pursuit and compensatory tracking of auditory pitch. *Ergonomics*, 14:479–486, 1971.

E. R. Miranda. The art of rendering sounds from emergent behaviour: Cellular automata granular synthesis. In *26th EUROMICRO Conference*, pages 350–355. IEEE Computer Society, 2000.

A. Morrison, G. Ross, and M. Chalmers. Fast multidimensional scaling through sampling, springs and interpolation. *Information Visualization*, 2(1):68–77, 2003.

S. Morrison and J. Keogh. Changes in the dynamics of tremor during goal-directed pointing. *Human Movement Science*, 20:675–693, 2001.

E. Moulines and F. Charpentier. Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones. *Speech Communication*, 9:453–467, 1990.

C. Müller-Tomfelde and S Steiner. Audio-enhanced collaboration at an interactive electronic whiteboard. In *ICAD'2001*, pages 267–271, 2001.

A. Murata. Improvement of performance by method for predicting targets in pointing by mouse,. *ICEC Transactions Fundamentals*, E78-A(11):1537–1541, 1995.

K. Murphy and S. Russell. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Sequential Monte Carlo Methods in Practice*, pages 176–183. Springer-Verlag, 2001.

E. Mynatt. Transforming graphical interfaces into auditory interfaces for blind users. *Human-Computer Interaction*, 12:7–45, 1997.

Y. Nam and K. Wohn. Recognition of space-time handgestures using hidden Markov model. In *ACM Symposium on Virtual Reality Software and Technology*, pages 51–58, 1996.

National Transportation Safety Board. Grounding of the Panamanian passenger ship Royal Majesty on Rose and Crown shoal near Nantucket, Massachusetts. Technical Report NTSB Number MAR-97/01; NTIS Number PB97-916401, National Transportation Safety Board, 1995.

D. Norman. Cautious cars, cranky kitchens, demanding devices. Talk, 2006. SIMS Distinguished Lecture. Given at Northwestern University on March 1st, 2006.

Abstract and audio http://www.sims.berkeley.edu/about/events/dls03012006 (retrieved 11th June 2006).

M. Noyes and T. B. Sheridan. A novel predictor for telemanipulation through a time delay. In *Proceedings of International Conference on Systems, Man and Cybernetics*, 1984.

D. R. Olsen, R. J. K. Jacob, S. K. Feiner, J. D. Foley, and J. D. Mackinlay. Uist'007 (panel): where will we be ten years from now? In *UIST '97: Proceedings of the 10th annual ACM symposium on User interface software and technology*, pages 115–118, 1997.

M. Palus and D. Hoyer. Detecting nonlinearity and phase synchronization with surrogate data. *IEEE Engineering in Medicine and Biology Magazine*, 17(6):40–45, 1998.

K. Partridge, S. Chatterjee, V. Sazawal, G. Borriello, and R. Want. Tilttype: accelerometer-supported text entry for very small devices. In *UIST '02: Proceedings of the 15th annual ACM symposium on User interface software and technology*, pages 201–204, 2002.

K. Perlin. An image synthesizer. *Computer Graphics*, 19(3):287 – 296, 1985.

K. Perlin. Quikwriting: continuous stylus-based text entry. In *UIST '98: Proceedings of the 11th annual ACM symposium on User interface software and technology*, pages 215–216, New York, NY, USA, 1998. ACM Press.

C. A. Phillips and D. W. Repperger. Why engineers should know and use Fitts' law? In *19th International IEEE/EMBS*, volume 4, pages 1686–1689, 1997.

R. W. Picard, E. Vyzas, and J. Healey. Toward machine emotional intelligence: Analysis of affective physiological state. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(10):1175–1191, 2001.

R. Plamondon and A. M. Alimi. Speed/accuracy trade-offs in target directed movements. *Behavioral and Braiin Sciences*, 20:279–349, 1997.

E. C. Poulton. *Tracking skill and manual control*. Academic Press, New York, 1974.

W. T. Powers. *Behavior: The Control of Perception*. Aldine, Hawthorne, NY, 1973a.

W. T. Powers. Feedback: beyond behaviorism. *Science*, 179(4071):351–356, 1973b.

W. T. Powers. Looking for controlled variables. *BYTE*, 4(8):96–112, 1979.

W. T. Powers. *Living Control Systems: Selected papers of William T. Powers*. The Control Systems Group Book, 1989.

W. T. Powers. *Living Control Systems II: Selected papers of William T. Powers*. The Control Systems Group Book, 1992.

M. Proschowsky, N. Schultz, and N. E. Jacobsen. An intuitive text input method for touch wheels. In *CHI 2006*, pages 467–470, 2006.

S. Qin and T. Badgewell. An overview of industrial model predictive control technology. *Chemical Process Control - V*, 93(316):232–256, 1997.

JD Rains. Signal luminance and position effects in human reaction time. *Vision Research*, 3:239–251, 1963.

A. Ramsay, 2004. Undergradute Thesis, University of Glasgow.

M. Rath. *Interactive realtime sound models for humancomputer interaction*. PhD thesis, Universit 'a degli Studi di Verona, 2004.

M. Rath and D. Rocchesso. Continuous sonic feedback from a rolling ball. *IEEE MultiMedia*, 12(2):60–69, 2005.

E. Ravelli, M. Sandler, and J. P. Bello. Fast implementation for non-linear time-scaling of stereo signals. In *8th Int. Conference on Digital Audio Effects*, pages 182–185, 2005.

G. L. Ricard. Manual control with delays: A bibliography. *Computer Graphics*, 28(2): 149–154, 1994.

G. Rigoll, A. Kosmala, and S. Eickeler. High performance real-time gesture recognition using hidden Markov models. *Lecture Notes in Computer Science*, 1371:69, 1998.

M. Rinott. Audio-tactile interactions with mobile devices. Master's thesis, Interaction Design Institute, IVREA, 2004.

M. Rinott. Sonic texting. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, 2005.

B. Ristic, S. Arulampalam, and N. Gordon. *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House Radar Library. Artech House Series Publishers, 2004.

C. Roads. *Microsound*. MIT Press, 2002.

C. Roads. Automated granular synthesis of sound. *Computer Music Journal*, 2(2):61–62, 1978.

A. C. Robbins. Pilot variability during pilot-induced oscillation. Master's thesis, Virginia Polytechnic Institute and State University, 1999.

M.T. Rosenstein, A.H. Fagg, and R.A. Grupen. Robot learning with predictions of operator intent. In *AAAI Fall Symposium on The Intersection of Cognitive Science and Robotics: From Interfaces to Intelligence*, pages 107–108, 2004.

M.T. Rosenstein, A.H. Fagg, S. Ou, and R.A. Grupen. User intentions funneled through a human-robot interface. In *Proceedings of the 10th International Conference on Intelligent User Interfaces*, pages 257–259, 2005.

J. A. J. Roufs. Dynamic properties of vision - v: Perception lag and reaction time in relation to flicker and flash thresholds. *Vision Research*, 14:853–869, 1974.

G. A. Rousselet, M. Fabre-Thorpe, and S. J. Thorpe. Parallel processing in high-level categorization of natural images. *Nature Neuroscience*, 5(7):629–630, 2002.

S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

V. Sazawal, R. Want, and G. Borriello. The unigesture approach. In *Mobile HCI 2002*, pages 256–270, 2002.

G. Schalk, J.R. Wolpaw, D.J. McFarland, and G. Pfurtscheller. Eeg-based communication: presence of an error potential. *Clinical Neurophysiology*, 111:2138–2144, 2000.

R. A. Schmidt and T. D. Lee. *Motor Control and Learning*. Human Kinetics, 2005.

R. A. Schmidt and T. D. Lee. *Motor Control and Learning: A Behavioral Emphasis*. Human Kinetics, Champaign, IL, 1999.

G. T. Shanks, S. L. Gale, C. Fielding, and D. V. Griffith. Fligth control and handling research with the vaac harrier aircraft. In M. B Tischer, editor, *Advances in Aircraft Flight Control*, pages 159–186. Taylor and Francis, 1996.

C. E. Shannon. Prediction and entropy of printed English. *Bell System Technical Journal*, pages 50–64, 1951.

T. B. Sheridan and W. R. Ferrell. *Man-machine Systems: Information, Control, and Decision Models of Human Performance*. M.I.T. Press, Cambridge, USA, 1974.

L. Smith. The maintenance of uncertainty. In *Proc International School of Physics 'Enrico Fermi', Course CXXXIII*, pages 177–246. Societa Italiana di Fisica, Bologna, Italy., 1997.

O. J. M. Smith. A controller to overcome dead-time. *Instrument Society of America Journal*, 6(2):28–33, 1959.

A. Steed. Supporting mobile applications with real-time visualisation of GPS availability. In *MobileHCI 2004*, volume 3160 of *Lecture Notes in Computer Science*, pages 373–377. Springer-Verlag, 2004.

S. Strachan and R. Murray-Smith. Muscle tremor as an input mechanism. In *UIST 2004*, 2004.

S. Tanimoto. Transparent interfaces: Model and methods. In *Workshop on Invisible and Transparent Interfaces*, 2004.

W. J. Teahan and John G. Cleary. The entropy of English using PPM-based models. In *Data Compression Conference*, pages 53–62, 1996.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

J. Theiler, S. Eubank, A. Longtin, B. Galdrikian, and J. D. Farmer. Testing for nonlinearity in time series: The method of surrogate data. *Physica D*, 58:77–94, 1992.

H. Thimbleby. *User Interface Design*. ACM Press, NY., 1990.

B. Truax. Real-time granular synthesis with the DMX-1000. In *ICMC Proceedings*, pages 231–235, 1986.

B. Truax. Real-time granular synthesis with a digital signal processor. *Computer Music Journal*, 12(2):14–26, 1988.

B. Truax. Discovering inner complexity: time shifting and transposition with real-time granulation technique. *Computer Music Journal*, 18(2):38–48, 1994.

K. van den Doel, P. G. Kry, and D. K. Pai. Foleyautomatic: physically-based sound effects for interactive simulation and animation. In *SIGGRAPH '01*, pages 537–544. ACM Press, 2001.

D. Venolia and F. Neiberg. T-cube: a fast, self-disclosing pen-based alphabet. In *CHI '94: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 265–270, New York, NY, USA, 1994. ACM Press.

K. Vertanen. Efficient computer interfaces using continuous gestures, language models, and speech. Master's thesis, University of Cambridge, 2004.

E. W. Vinje. Flight simulator evaluation of audio ifr displays for v/stol hover control. In *Proceedings of the 8th Annual Conference on Manual Control*, volume AFFDL-TR-72-92, 1972.

D. J. Ward and D. J. C. MacKay. Fast hands-free writing by gaze direction. *Nature*, 418 (6900):838, 2002.

D. J. Ward, A. F. Blackwell, and D. J. C. MacKay. Dasher - a data entry interface using continuous gestures and language models. In *UIST 2000*, pages 29–137. ACM Press, 2000.

D. Washburn and L.M Jones. Could olfactory displays improve data visualization? *Computing in Science and Engineering*, 6(6):80–83, 2004.

A. Wexelblat. Research challenges in gesture: Open issues and unsolved problems. *Gesture and Sign Language in Human-Computer Interaction*, 1371:1–11, 1998.

N. Wiener. *Cybernetics: or Control and Communication in the Animal and the Machine*. MIT Press, 1948.

D. Wigdor and R. Balakrishnan. Tilttext: using tilt for text input to mobile phones. In *UIST '03: Proceedings of the 16th annual ACM symposium on User interface software and technology*, pages 81–90, New York, NY, USA, 2003. ACM Press.

J. Williamson and R. Murray-Smith. Audio feedback for gesture recognition. Technical Report TR-2002-127, University of Glasgow, Scotland, UK, June 2002.

J. Williamson, S. Strachan, and R. Murray-Smith. It's a long way to Monte Carlo: Probablistic GPS navigation. In *Mobile HCI 2006*, pages 89–96, 2006.

A. D. Wilson and A. F. Bobick. Parametric hidden Markov models for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(9):884–900, 1999.

I. Xenakis. *Formalized Music: Thought and mathematics in composition*. Indiana University Press, 1971.

Y. Yanagida, H. Noma, N. Tetsutani, and Tomono A. An unencumbering, localized olfactory display. In *In Proceedings of ACM CHI 2003*, pages 988–989. ACM Press, 2003.

S. Zhai and P.-O. Kristensson. Shorthand writing on stylus keyboard. In *ACM Conference on Human Factors in Computing Systems (CHI 2003)*, pages 97–104. ACM Press, 2003.

H. Ziebolz and H. M. Paynter. Possibilities of a two-time scale computing system for control and simulation of dynamic systems. In *Proceedings of the National Electronics Congerence*, volume 9, pages 215–223, 1954.