ADAPTIVE USER INTERFACES IN

COMPLEX SUPERVISORY TASKS

By

LEMI DAGHAN ACAY

Bachelor of Science

Boğazici University

Istanbul, Turkey

2001

Submitted to the Graduate College of
Oklahoma State University
in partial fulfillment of
the requirements for
the degree of
MASTER OF SCIENCE
July, 2004

Acknowledgments

Thanks to everybody, especially my advisor Prof. Gary Yen, who encouraged and supported me through my adventure in the jungle of the knowledge.

Stillwater, Oklahoma US
July 7, 2004

Lemi Dağhan Acay

ADAPTIVE USER INTERFACES IN COMPLEX SUPERVISORY TASKS

Thesis Approved:

Thesis Advisor

Dean of the Graduate College

Table Of Contents

List of Tables

List of Figures

Chapter 1

Introduction

This research has aimed to understand the human and computer interaction in order to design adaptive interfaces to enhance productivity and efficient uses of computers. Additionally, we aim to find efficient and conducive work environments by merging the human cognitive capabilities with the computer's accurateness and process power. In order to accomplish these objectives, we need to introduce the human to the computer and visa versa. This research will lead us to find good heuristics (or even formal methods) for interface adaptation to meet users needs under various environmental constraints.

In order to begin discussing about the human-computer interaction, we need to define the nature of both *human* and the *computer*. Balint [6] defines human and computer as follows.

Computer is designed by intention so that it is deterministically performing well-defined operations, and interaction is served by proper hardware components (keyboard, mouse, visual display unit, etc.) and by specific software components, as well as database/knowledge-base.

Human is behaving, by its nature, non-deterministically, based on such properties as:

- physiological attributes (eyes, ears, fingers, etc.),
- intellectual characteristics (capacity, recognition, learning, decision, etc.),
- knowledge bases (knowing the environment, the system, himself/herself, etc.),
- psychological states (concentration, vigilance, fatigue, patience, etc.).

Researchers differ in their perspectives as how to enable effective human-computer interaction to close the gap between these two agents in order to facilitate safe and effective operations. For example, Salvendy and Richter [55] suggest that computers will need to be built with intuition, draw inferences from analogy, and be able to communicate and understand speech and gestures. However Hancock, Chignell, and Takeshita [15] propose that intelligent interfaces must improve the effectiveness of augmentation technologies-those technologies that add external functionality to empower human performance. Yet, others such as Vicent, Rasmussen, and Kirlik [71] argue that the guiding principle to the effective interface design is that, perceptual information available through the interface must be congruent with the actions made available by the interface controls.

While researchers in human-computer interaction may disagree on the level of computer involvement or mode of interaction, a common framework of analysis emerges [56]. There are three dimensions in defining successful interface design.

1. Taxonomy of user abilities

2. Taxonomy of Tasks

3. Situated Taxonomies (Environmental, Social, Cultural, and Personal Traits)

One perspective of situated taxonomies deals with the features of the environment that facilitates user activities. These features may include the degree to which the task to be completed is well structured, the extent to which a task environment is dynamic, the amount of time under stress, and the severity of consequences to the user. Other situated taxonomies outline personal traits and social/cultural attributes that are directly impacted by the situation. Personal traits include field dependency, job satisfaction, physical limitations, color deficiencies, and physical, visual, or language limitations [18]. Social attributes include shyness and mental stability. Cultural attributes include traditions, customs, and norms.

Unfortunately there are no analytic models of human and cognition to start with. However there are some models that have been empirically shown to be good approximations of human performance. Modelling the user's action strategy to satisfy a goal or user response to an event is a general approach to user modelling. Cognitive modelling of user such as GOMS (Goals Operators Methods Selection) [11] and variations to it are available. More to GOMS, there are tools to simulate thought and cognitive modelling such as ACT-R [4] and Soar [31], which is built on Newell's [44] cognition theory and also Jess [30], all of which are using production systems. This kind of modelling generally incorporates some kinds of knowledge base strategies. Second approach to user modelling is more concentrated on response of user to events. Optimal control theory [38], information theory, game theory, queuing theory [72], have

been applied in this area of research. This research is used more in war simulations and complex environment performance such as process monitoring in nuclear plants.

In this thesis we have developed an adaptive interface structure especially for complex environments such as nuclear plant monitoring or air traffic control. We have developed the user modelling technique, associated performance index and adaptation technique. Our user model was constructed as a multi-variate Gaussian localization function to be explained in Section 3.3.3. Our adaptation method was formulated in the domain of optimization and Genetic Algorithm (GA) was employed as an efficient search tool to be explained in Section 3.3.2. Our performance metric is chosen to measure the performance of the user to be explained in section 2.2.3 and in section 3.3.1. Finally we will show performance of the algorithm using a simple replica of Air Traffic Control (ATC) task.

Development of the thesis starts with the literature survey. Literature survey consists of two main subsections namely "Human understand Computer" and "Computer Understands Human". We will cover current trends of interface development in two different perspectives. This section is meant to give basic intuition about the interface domain. Section 2.2 is especially important in the sense that it defines the foundations of the algorithm. Chapter 3 defines the proposed architecture and Chapter 4 summarize the results with real user and so called automated user. Statistical distribution of the parameters is given under Appendix A and Appendix B for convenience. Finally Appendix D is given to demonstrate the adaptation of the algorithm using a simple example.

Chapter 2

Adaptive Interfaces

Interface design is considered to be an important part of the system after physical implementation barriers were surpassed. Today computational power is cheaper and software development cycles are getting shorter. Ready to use software libraries are letting people to quickly develop new application tools and this, in return, adds up to the library repository. Due to these developments, most of the processes are automated and rely on the precision of the computers. After all, computers started fusing into intelligent domains. A good example could be the decision support systems (DSS). Now we are at the point of regarding the computers as our collaborators. Intelligent aspects of computers enhance the importance of the interface development since, efficient collaboration relies on the good communication. Due to language differences of these agents (machine language and natural language) interface should incorporate some types of intelligence for efficient communication. It should be able to adapt itself to different users in order to increase translation power. At this point we better give the exact definition for intelligent interface for human computer interaction. Hancock and Chignell [14] defined intelligent interface as follows:

> "An intelligent entity mediating between two or more interacting agents who possess an incomplete understanding of each others' knowledge and/or form of communication."

The above phrase gives the basic intuition about why this thesis is presented and the problem of man-machine interaction is posed as an important research topic. This topic is inherently multidisciplinary and complex. Merely defining terms in the intelligent interface definition is a fuzzy and vague process terms. For example the intelligence, agent or knowledge are open to different interpretations. As we continue to advance, difficulty of the problem will become more apparent. Although human computer interaction (HCI) supply us with some tools, the solution to interface adaptation is not obvious and the field is open to development and contributions from different domains. Clarification of these concepts and mediatory mission of interface is even more important than before. This proposition becomes more apparent as we will see in the following sections. Before we elaborate further on adaptation we are going to review some literature in order to define current solutions to interface development.

Interface development has been researched since the beginning of 1980, but human computer interaction (HCI) is still in its infancy. This section will try to give some taxonomy, to approach this problem.

According to literature survey we were able to define two different methods to attack this problem:

1. Human Understands the Computer

2. Computer Understands Human

The general tendency in the HCI field can be classified in the first group. This group is labeled as *"Human Understands the Computer"* within our taxonomy. By saying "Human Understands the Computer," we try to convey the intuition of learning and adaptation of user to the physical interface. On the other hand there is a new approach to this problem which can be classified as *"Computer Understands Human."* By saying "Computer Understands Human" we try to convey the meaning of interface having capabilities of learning, deduction and inference in order to adapt itself to the user. There is an ongoing debate on the type of approach which is best for design and usability purposes [64]. The next two sections will cover these two approaches.

2.1 Human Understand the Computer

Under this category we consider methods that propose user adaptation to the interface. These interfaces are static but they are developed with the user in mind for the ease of learning. Interface development techniques, which are based on psychology, were defined for these interfaces. Properties of these interfaces set at design time. For example design of a supervisory control task interface can populate Fitt's Rule [24] approach to assign the tasks to corresponding agents (Human/Computer) based on their properties. Norman's [46] model of interaction defines the steps of interaction between man and machine. In his model execution and evaluation can be modelled in seven steps as shown in Table 2.1

Norman didn't consider the interface component in his seven step model of inter-

Table 2.1: Norman's models of interaction

1. Establishing the goal
2. Forming the intention
3. Specifying the action sequence
4. Executing the action
5. Perceiving the system state
6. Interpreting the system state
7. Evaluating the system state with respect to goals and intentions

action. Abowd and Beale [1], address this problem and define a different interaction model shown in Figure 2.1.

According to Abowd and Beale, interface corresponds to the *Input* and *Output* part of the Figure 2.1. In this model, the user initiates the interaction cycle with the formulation of a goal and a plan to achieve that goal. The user has to articulate the task using the input language. The input language is translated to the *core language* via the *performance* link and the task is performed, hence the end of the performance cycle. Now that the performance cycle has ended, evaluation phase begins. The system is now in a new state, which must be communicated to the user. The *Presentation* layer sends the new state to the output in *output language*. From this point, it is the *user* who will interpret the current system state with respect to the goal. The proper design of the articulation process within the interface is crucial. The task should be phrased in terms of certain psychological attributes that highlight the important features of the domain for the *user*. If these psychological attributes map clearly onto the input language, then *articulation* of the task will be made much

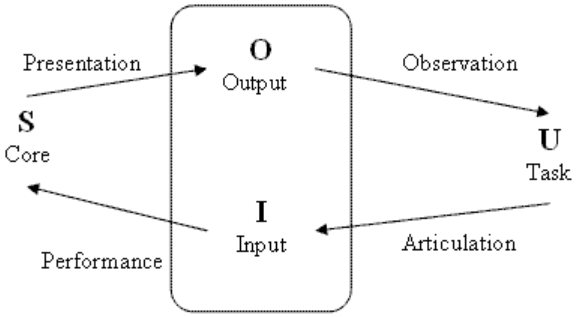


Figure 2.1: Interaction Model (Abowd and Beale)[1]

simpler [20]. An example of poor mapping occurs in a large room with overhead lighting controlled by a bank of switches. Assume a task is defined as lighting the certain part of a room using the banks of switches. The user is faced with the puzzle of determining which switches control which lights. The consternation resulting from repeated experimentations with the switches to achieve the desired lightning effect can be traced to the difficulty of articulating a goal such as 'Turn on the lights in the front of the room' in the input language, which consists of a linear row of switches that may or may not be oriented in a manner suggestive of their operation [47]. It is this very mapping that can be mended by so called "easy to perceive and manipulate' methodology, to enhance interaction. We will cover this idea in the next section.

The output language, on the other hand, should not impose extra load on the cognition of the user. Interpretation of the system's new state should be as intuitive as possible. Observation methods and terms of the user should be considered before the output language of the interface is designed. Simple good and bad output language is depicted in Table 2.2. When the aim is to find the biggest figure, it takes more time for the user to perceive the representation on the right than the representation

Table 2.2: Importance of representation(Finlay)[20]

3.14159	2.2323
127.865	532.56
1.005763	179.03
382.538	256.317
2502.256	15.002
432.935	1273.9

Alignment and Layout are important. Find The biggest figure in each column.

on the left. This is important in complex supervisory control tasks. Assume there is a catastrophic change at the nuclear plant and the operator should decide on an action based on the data supplied during interaction. Under this circumstance, if the output language is not properly mapped, with the addition of stress exerted on the operator, it may easily lead to an unexpected ending.

The following sub-sections will cover interface design methods based on the ideas defined above. As we have mentioned in the beginning of this chapter, these interfaces are designed with the user in mind; however, they are mostly static and not suitable for different user population. Environmental constraints also require an interface to be more flexible in order to increase the efficiency. Designer is responsible for considering all possible situations and incorporate them into the kernel which in turn leads to insufficient or biased interfaces. Eventually user needs to learn the interface to be able to manipulate the computer. User seminars or online learning schemas are considered but they are not very efficient especially in complex application domains due to the small slope of the learning curve.

2.1.1 Direct Manipulation

Early computer input language consisted of predefined commands that were executed by writing on the command interface. For example, to list the files in a directory the user should have known the 'ls' command for UNIX systems or the 'dir' command in DOS systems. Articulating a goal structure in command line interfaces were hard to perceive and execute since the user had to know the correct function to execute a task. This in turn increased the cognitive load of the user who had to translate task language into input language. Observing system states was also hard since no graphical method was employed to transfer the system current state. Organized forms or spread sheet like outputs were available however, the user was required to recall all the commands to call these functions.

The graphic based interfaces were proposed for the easy execution and perception of the manipulation. Direct manipulation is a precedence of the WIMP (Windows Icons Menus Pointers) and Point-Click interfaces. WIMP interfaces are direct metaphor of human psychology. Humans are able to think in distributed fashion and use nonlinear methods of accomplishing a task. This means, humans handle multiple tasks simultaneously instead of starting and finishing one task as the linear methods suggests. This requires the interface to be flexible and be able to support multiprocessing. However as a human initiate new tasks, the interface should emphasize the individual mental threads. This was realized by wrapping each task by a window. On the other hand, Point-Click interfaces are limited but easy to use systems generally introduced in hypertext/hypermedia environments. Direct manipulation coined

in 1982 by Ben Schneiderman [63, 62]. It is an enhanced version of WIMP interfaces. He highlighted the following properties of direct manipulation interfaces.

1. Visibility of the object of interest
2. Incremental action at the interface with rapid feedback in all actions
3. Reversibility of all actions, so that users are encouraged to explore without severe penalties
4. Syntactic correctness of all actions, so that every user action is a legal operation
5. Replacement of complex command languages with actions to manipulate directly the visible objects.

Direct manipulation also allows using metaphors to help the user to understand the structure of the interface. For example, files and directories in terms of storage objects are easier to grasp and manipulate since the semantic distance between the task and the interface was reduced when you represent this concepts graphically. In addition to that, moving a file from a folder to another folder is handled by drag and drop operations, which helps to perceive the system state and easier articulation of the task into input language. Another outcome of direct manipulation interfaces are called *'What You See Is What You Get'* (WYSIWYG). The best example for this kind of interface is a word processor or HTML page. Every document is processed before it is sent to a printer. In WYSIWYG interfaces, the difference between actual output from the printer and the version on the screen is minimal. This helps the user to arrange your document with much less guess work. More importantly user

operation on the document is directly reflected in the final draft hence reducing the semantic distance between the operation and mental model of the user. A familiar example for the WYSIWYG can be Microsoft Word®. LaTeX can be considered on the other extreme side of the interface methodology.

2.1.2 Tangible Interfaces

Tangible interfaces are somewhere between virtual reality (VR) and direct manipulation. In tangible interface paradigm, objects used in direct manipulation interfaces are no more abstract. Physical gadgets are used to interact with the system [13, 36]. An example to this can be a realization of desktop metaphor. One can use the specially designed physical devices and discard the mouse and the pointing devices from the interaction. By moving interaction from the virtuality of the screen into the physicality of the real world, the design space is significantly extended, enabling new and richer forms of interaction. However, at present we are only at the beginning of understanding the implications of establishing a direct physical link between the digital information and the human users. Although many compelling examples illustrating the use of tangible interfaces across different domains have been produced, these are usually in the form of discrete entities and suffer from generality of common framework.

2.1.3 Multi-modal Interfaces

Multimodal interfaces use extra sensory channels in communication thus enhancing the bandwidth of the interaction. While the visual channel is dominant in today's computer interfaces, auditory and haptic (tactile) senses are vital to our perceptions in

daily life. Supporting communication on these channels is one aspect of multi-modal interfaces. Other methods of human-human communication are also employed in today's multimodal interfaces. Non-speech sound has already been embedded into the interfaces for alarm or warning purposes. Speech recognition and synthesis are other branches of research. One early example of this kind of interface is "SoundTrack", a word processor with an auditory interface designed for visually impaired users [22]. Natural language processing is a far-reach research field but there are still problems with the semantics.

Another branch of psychology is focusing on gesture based interfaces [21]. Using the gesture of the user in the interaction between humans and computers would be more natural since it closely resembles human-human communication. Multimodal interfaces seek to tap these under-utilized aspects of communication in order to increase communication bandwidth and define more natural way of communication.

2.1.4 Ubiquitous Computing

The initiative of ubiquitous computing is to liberate the computational power of a massive desktop environment and creates a more friendly interface. As the name implies this kind of computation, also called "everywhere computing" and primarily targets on the mobile devices. This idea was first started in Xerox PARC and led by Mark Weiser in the late 1980's [73, 74]. The ultimate goal of ubiquitous computing is to create a computer infrastructure that permeates our physical environment so much that we do not notice the computer any more. Although this section may seem irrelevant in a survey of the interfaces and interface design, ubiquitous computing

requires special attention. The very first reason is the different size computers require different treatment in the interface design phase. For example display real estate on a PDA is much more expensive than on a laptop which is in turn more than on a desktop. Color displays are limited so color schemes in HCI to convey the information is of little or no use. Mobility also effects the type of applications, thus interfaces, related to each equipment [10]. Current research also supports the wireless networks and application for invisible computing and brings their problems with it. All these facts definitely effect the interface design.

2.1.5 Hypertext / Hyper-media

The initial seed of this idea was planted back in 1945. Vannevar Bush published an article "As we may think" in the *Atlantic Monthly*. He pointed out that increasing scientific knowledge was becoming inconceivable and it was hard to get a grasp of it. He created an information retrieval apparatus - the *memex*- aimed for increasing the human capacity to store and retrieve connected pieces of knowledge by mimicking our ability to create randomly associate links in our brain. Nelson coined the term *hypertext* to name this non-linear data structure.

In this age the hypertext/hypermedia interfaces are very important part of the multimedia interfaces. The power of the hypertext/hypermedia comes from its non-linear structure. Nonlinear association of the knowledge is handled by the links which enable one to jump from one node to another before finish reading the text to the end. This random access structure has the power of linked structures of information and resembles the way of our naturally cognitive information.

The WWW or World Wide Web is one of the outcomes of the hypertext information representation connected with transfer protocols such as TCP/IP. It is first initiated at CERN, European Particle Physics Laboratory in Geneva in 1989 by Tim Berners-Lee. The first text-based web browser was released in 1991 and shortly thereafter first graphical web browser was developed. In only ten years the WWW explode to its current level. It is a very challenging topic for interface developers to solve the problems of the web, since it is getting more and more difficult for users to locate the information they need. Researchers seek to find better ways to track the past interactions of the user for efficient IR (Information Retrieval) methods. We are going to address these intelligent interfaces in the section named "Computer Learns Human."

Another interesting aspect of the web is its multi-modality offered by hyper-media. Hyper-text can incorporate more than one type of media such as text, audio, or video. Multi-modal interaction style opens different era in the interaction. This also stipulates open research areas to define the best layout for the interface, interface design compatible with different platforms (see Section 2.1.4), ergonomics in the interface design, etc

2.1.6 Groupware Systems

Up to now, we have considered interface design for a single user. Under this section we will briefly introduce the concept of groupware and Computer Supported Collaborative Work (CSCW) systems. This area of research emerged due to the increased use of the networking and the Internet. Although there are different classifications for CSCW, we will consider the three class taxonomy which is defined as follows.

- **Computer-Mediated Communication**: supporting direct communication between participants such as e-mail or conference systems.

- **Meeting and decision support systems**: capturing common understanding and mediating meetings for efficient and targeted communication.

- **Shared application and artifacts**: supporting the participants and enabling them to share an object for design such as shared workspace in a CAD(Computer Aided Design) software.

Naíve but efficient classification of groupware systems can also be given in temporal/spatial dimensions in Table 2.3. This table shows different means of communication under the dimensions of time and place. Viability of the interaction greatly suffer as we move to bottom right of the Table 2.3. It is generally desired to be able to communicate face-to-face due to high bandwidth. Face-to-face communication enjoys the use of gesture and body language on the other hand letter form the weakest form of interaction. However today's developing societies and globalization of the interaction forces the companies under time and place discrepancies. Computer may improve this deficit and merits a great deal of consideration in today's multi-national connections.

Table 2.3: Temporal/spatial matrix

	Same Place (co-located)	Different Place (remote)
Same Time (synchronous)	Face-to-face Conversation	Telephone
Different Time (asynchronous)	Post-it Note	Letter

In addition to the above classifications, there are groupware systems that are used for information retrieval within the group such as GroupLens [54] or information

filtering [66]. Last two interfaces are able to adapt to the user through interaction. User's preferences are extracted from group membership and group characteristics.

2.1.7 Natural Language Interfaces

We have already covered this topic under Section 2.1.3 titled 'Multimodal Interfaces'. Salvendy and Richter [55] mention that an interface can be considered as intelligent if it can communicate with the user in their own language. Although, there are some improvements in natural language processing this field still suffers from the unstructured context dependent meaning. Current parsers are able to specify formal characteristics of the sentence but the overall meaning of the dialog is still difficult for computer to perceive.

Still another difficulty arises from the signal processing perspective. Extracting the word information from speech is not an easy task, since the vocal path of every individual is different. In addition to voice distortions and ambient noise level, pronunciation also differs from person to person. This is a long road ahead to enable computers to understand natural language. Currently there are some commercial programs which are able to recognize limited amount of vocabulary and are able to execute voice commands. Operating systems such as MacOS or Windows are the best known examples in this category.

2.1.8 Affective Computing

Affective computing deals with affective modes of interaction such as gestures or emotional attitudes. This branch of interface development tries to synthesize [60] and capture the emotional levels of the user [37]. Affective computing is close to the human

ergonomics research in the way the problem is approached. According to James-Lange, actions precede emotions and the brain interprets said actions as emotions. A situation occurs and the brain interprets the situation, causing a characteristic physiological response. This may include any or all of the following: perspiration, heart rate elevation, facial and gestural expression. Current psychophysiological tools such as Skin Galvanic Response (SGR lie detector) and Blood Volume Pulse (BVP) can be used to assess user emotional state. Same techniques are also employed in human ergonomics research and will be covered in Section 2.2.

2.1.9 Agent-based Interfaces

Agents are fairly new topic and propose an intuitive solution to the complex and distributed tasks. General problems in the agent domain can be classified as, communication protocols within the agent community, open architecture, negotiation methods and finally the inference mechanisms. The open architecture stands for the commonness of all the agent snippets whatever the underlying programming language is.

Interface agents are the computer programs that provide personalized assistance to the users with their computer-based tasks. Most of the interface agents achieve personalization by learning the user's preference in a given application domain and assisting them in accordance with this knowledge [61].

Maes [40] uses the analogy of human-human communication for agent interfaces. Communication can be started by either agent. They may be collaborating on one task or handling discrete tasks.

The use of interface agents is not without difficulties. Horwitz [29] has pointed out some problems with the use of interface agents. Poor guessing about the goals and needs of users, inadequate consideration of the costs and benefits of each agent action and, poor timing of agent actions are general concerns in the HCI domain. Consistency and timing of the interface is important so as not to distract the user.

All types of classification mentioned above, reflect more or less fixed structures and design time implementation. However, we shouldn't say "Human Understand the Computer" interfaces are totally static, since they still can be classified as adaptable [59]. But we should mention that the user is responsible for initiating the adaptation and breaking the communication barriers. The next section, titled "Computer Understands Human" will cover the possibility of computer initiated adaptation and taxonomy of adaptive interfaces.

2.2 Computer Understands Human

We have touched the surface of the interface design in the previous Section *"Man Understands Computer"*. We have discovered that none of the approaches regards adaptation through use. In the other words there is little or no room for adaptation of the interface to personal traits. Question:

> *"Can we develop interface which adapts to the user?"*

is what we are going to ask and try to answer. *"Can a computer be a teammate via adaptation?"*. There is an ongoing debate about interface adaptation [64].

This category is a newer concept than the *"Men Understand the Machine"* approach. Actually this category is known as Intelligent User Interface (IUI) in HCI domain, Adaptive User Interface (AUI) in human ergonomic domain or Intelligent Human Computer Interaction (IHCI / HCII) in the computer science domain. This part is actually where we focus our attention. To best the of our knowledge we can give definitions of each acronym above as follows.

IUI Apply available knowledge about the the task to control of the interaction, in order to increase efficiency and usability.

AUI Utilize the knowledge about the user to configure the interface for different users, i.e. each user may have different skills, level of experience, or cognitive and physical abilities.

IHCI/HCII Enhance the I/O devices to improve the interaction such as a wearable computer or virtual reality.

The above classification is by no means exhaustive. Due to the multi-disciplinary nature of the topic there are hundreds of books and articles about problem specific architectures. There are also applications belonging to two or all of the disciplines.

2.2.1 Adaptive Interfaces

Although we will consider intelligence in the sense of a human factors approach we will give some taste of other possible ways to consider intelligence in the other domains. Actually any interface or computational entity interacting in a more 'natural' way

can be considered as intelligent. This can correspond to natural language interfaces or tangible interfaces covered in the previous section.

As a matter of fact, even the definition of adaptation varies from person to person. For example Keeble and Macredie [35] define an adaptive interface as "One where the appearance and, function of the interface can be changed by the interface (or the underlying application) itself in response to the user's interaction with it." On the other hand Wickens [75] thinks that the trigger for adaptation is not necessarily a user action by saying "Adaptive systems are those in which some characteristics of the system changes, and adapts, usually in response to measured or inferred characteristics of the user."

It is an undeniable fact that, human centered design should be employed within the user community. In all domains of interest either information presentation or decision support were considered important aspects of user-centeredness.

2.2.2 Adaptive Interfaces vs. Direct Manipulation

Adaptive interface shouldn't be seen as a remedy to all the problems in interface design. Before we start developing an adaptive interface we need to consider negative and positive aspects of it. Adaptive interfaces have limitations and down sides. Some examples of the negative aspects of adaptive interfaces are listed in the following [45]:

1. If the system is frequently changing the *User* may not be able to create a legible model of the system. This may undermine the user's confidence and performance with the system. If the user cannot understand the system's behavior, the user's effectiveness can be seriously reduced.

2. This loss of control that the user may experience may lead the user to distrust and disguise his or her goals.

3. Implementation cost and complexity may not be worth implementing the adaptive interface. In addition adaptive interfaces may require more computation resources in order to run embedded algorithms.

On the other hand there are well defined metrics and evaluation methods for direct manipulation interfaces. They require less execution times and in some cases their performance is superior to adaptive systems [75]. Schneiderman [5], Keeble and Macredie [35] found that users of adaptive interfaces sometimes felt they are loosing control.

Direct manipulation interfaces are well defined and developed through the development phase. Unpredictable results or system instability is not of concern. Every component is designed and enhanced using user evaluation or cognitive walk through processes. Environment and task is well defined beforehand in order to implement the interface.

Adaptive interfaces are shown to be useful in some other domains such as human factors. These type of interfaces may change the information representation or abstraction level in order to reduce the mental workload of the operator. They may help in the decision process such as decision support systems (DSS). They can change the level of automation in a complex system supervisory task in order to improve efficiency.

In order to satisfy the above constraints, we need to define areas of interest for adaptation to take place. One possible area of implementation is multimedia systems [12]. As we have covered in previous sections multimedia is very powerful in the sense of multi-modality and flexibility. Another domain is complex system monitoring such as nuclear plant or factory supervisory task. This is an important area of research since the decision making process of the operator is adversely affected under time pressure. This leads to different interface modality or information presentation methods in catastrophic situations.

2.2.3 Conceptual Framework for Adaptation

Before we start giving methods in adaptation in different fields, three important aspects of adaptive/adaptable interfaces must be considered [56]. User is responsible for tailoring the interface in the adaptable interfaces on the other hand adaptive interfaces are responsible to adapt to the user autonomously.

- Identification of variables that call for adaptation
- Determination of necessary modifications to the interface
- Selection of decision inference mechanism

It is these aspects that differ between fields. Any two applications are different by the definition of the above abstract class. According to Benyon and Murray [7] adaptive interfaces should consist of a *User model*, a *Domain Model*, and an *Interaction Model*. Some other authors also add an additional component, *System Model* [45]. Adaptive interface model is given in Figure 2.2

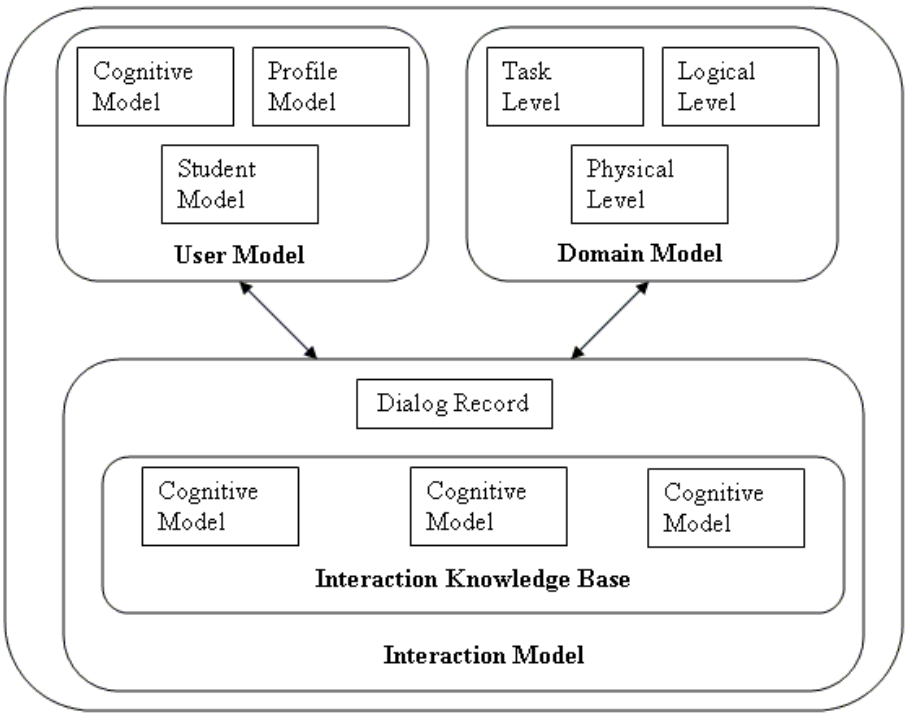


Figure 2.2: Adaptive Interface Model (Benyon and Murray)[7]

Here, the user model describes the user in terms of abilities, profile information, and domain knowledge. Domain models define the task structure of the system and the user such as user goals, the logical construct of the system, and the basic interaction mechanism. The interaction model consists of a historical record of the user's interaction with the system in order to carry out the adaptation, handle the inference process, and finally incorporate the evaluation mechanism of the performance for either the user, or system, or both.

Benyon's [7] definition of the adaptive interface was not fully implemented in many applications. Conceptual framework of the adaptive interface is little or no help to an adaptive interface designer. Most of the components are domain dependent and generally it is difficult to apply this framework to practical applications. However,

Table 2.4: Variables that call adaptation [56]

1. User Performance
2. User Goals
3. User Workload
4. User Situation Awareness
5. User Knowledge
6. Groups of Users
7. Situation Variables
8. Task Variables

abstraction of concepts is a great deal of help as it allows the big picture of the interface adaptation to be seen.

At this point we are confronted with the requirements of the adaptive interfaces. Methods to accomplish these tasks can differ greatly with respect to the domain. We will continue with the Rothrock *et. al.* [56] classification of interface design. If we are really required to classify these two approaches Benyon would represent HCI domain and Rothrock would represent the human factors domain as given in Section 2.2.

Variables that Call Adaptation

First, we should consider *variables that call adaptation.* Table 2.4 shows coverage to some variables. According to Viano *et. al.* [70] the first five are classified as 'Operator deviation' and last three are classified as 'Process deviation'.

User performance is generally defined as an error percentage in performing a task, as well as the time required to perform a task [57]. Examples of this kind of metric includes reaction time, capturing a simple target, and tracking deviation. Difficulty

in measuring the user performance lays in the lack of deterministic mapping between the mental process of the user and task performance. For example, response latency score does not always reflects the complexity of the mental process or level of mental workload.

User goals are sub-level or higher level goal structures to accomplish a task. According to [12] sub-level user goals can change easily; high level goals are however stable.

Cognitive workload is a highly elaborated topic in the human ergonomics field. It has been shown that both high and low cognitive workload adversely affect task performance [8]. The 'multiple resource theory' [75] is the predominant theory in human cognition research. There are three different assessment techniques used to measure cognitive workload, none of which is proven to be correct or an ultimate metric. The first technique is called *Psychophysical* method. An electroencephalogram (EEG) or Event-Related Potentials (ERP) are in this category [50]. New techniques in this field are evaluating the brain potentials and neuroimaging [58]. The second technique is called *Subjective measures*. This technique relies on the user self assessing their cognitive load. Questionnaires are employed in order to collect the value of the assessment. The most popular questioners are the NASA Task Load Index (TLX) [27] and Subjective Workload Assessment Technique (SWAT) [52]. The third technique is called *Secondary task method*. The operator is required to do the secondary task concurrently with the primary task. The secondary task is used to measure the vigilance or situation awareness. Like in any other methods there is an ongoing debate on assessment of workload level using secondary task [49]. However, Kaber and Riley

[34] showed that secondary task performance can be used for adaptation triggering.

User situation awareness is defined as 'The perception of the elements in the environment within a volume of time, the comprehension of their meaning, and the projection of their status in the near future. SAGAT (Situation Awareness Global Assessment Technique) [23] and SART (Situation Awareness Rating Technique) questionnaires are to measure perceived load on the operator. On the other hand the SAMPLE (Situation Awareness Model for Pilot-in-the-Loop Evaluation) is designed to measure the awareness in real-time without questioning the operator.

User knowledge is generally used in HCI domain and multimedia systems [12]. For example, each user has a *student model* in adaptive tutoring systems. This helps the interface to define the way the information is presented. The same student model may also help to define the best way to explain a topic of interest, such as in mathematic teaching systems [3].

Group profiles are generally used for web browsing or news article suggestion. It is an important and promising aspect of adaptation because it overcomes the human evaluation barrier by using other users habits in the group. For example, Group Lens [53] filters Netnews by suggesting the news that another person has already read and rated. Kun-Lung *et. al.* [77] uses a group model to help users in a WEB browsing task.

Situation variables can also trigger adaptation. This happens when a process goes from normal state to disturbed state. In these situations the user/operator is stressed very quickly by the number of alarms generated. The interface should help the operator to find the problem and solve it [70]. Situation variables includes

weather, social traits etc.

Finally the *System Variables* are like situation variables but are generally associated with an error or fault in the system.

Interface Modification Method

We are going to talk about the second step in our conceptual framework for interface adaptation. We should consider the *determination of modifications to the interface*. Assuming the designer specified the factors triggering the adaptation, (see the previous section) he/she also needs to define the way modifications are made to the interface. As covered in Section 2.2.2 adaptation pace and method should be selected carefully in order not to distract the user. Rothrock [56] classifies four types of interface modification. These are the content adaptation, the dialog adaptation, the task allocation and finally the speed of adaptation.

Content adaptation is one of the most applied methodologies in multimedia adaptation. Content can be adapted in a variety of ways. The following list is adapted from [56] and intend to give you the possible ways of content adaptation.

- *Selection of information* is a frequently used method. Information can be hidden or revealed according to users goals, expertise, or task/domain characteristics. Goal relevant information selection is introduced in Francisco-Revilla and Shipman [25]

- *Quantity of Information* can be changed depending on the user vigilance and situation awareness. For example, Sanderson [58] used graph displays or digital displays in order to improve process control efficiency.

- *Layout of Information* can be important since it is understood that human memory can be enhanced by chunking the relevant information. Spatial distance between relevant and irrelevant data directly affects the information retrieval times. In critical supervisory systems location of crucial information within the interface decreases the search and response time. Different methods have been proposed for spatial adaptation of information. For example Masui [41] uses adaptive methods for graph layouts in user interfaces.

- *Modality of information* was considered before in this thesis. Interface may use different modalities for simultaneous tasks. For example, two simultaneous tasks can separately be associated to human tactile or auditory input.

- *Augmentation of information* is providing supplementary feedback accompanying the information content. This is generally used when the operator performance is outside of some specified criteria. One implementation can be seen in Leonard [2] using the icons in distributed team decision environments for defence or surveillance systems.

Dialog adaptation is also considered. The objective of dialog adaptation is to adapt the menus, or presentation of hypertext links. A good example of this is the different ordering of the menu entries according to usage frequency [42]. Another example of dialog adaptation is to collapse the number of steps for frequent tasks such as creating macros. Finally, the user can be transferred from dialog based or direct manipulation interfaces to command based interfaces depending on the skill level of the user. Command base interfaces are more flexible and faster for the experienced

users while the direct manipulation interfaces are rigid but well structured and easy for the novice users.

The allocation of the function or level of automation is considered too. Automation refers to "system or method in which many of the processes of the production are automatically performed or controlled by autonomous machines or electronic devices." [50] This field is researched extensively in the aviation systems such as cockpit or Air Traffic Control Systems (ATC). A major issue is the level of automation which was given by Sheridan and Verplank [67]. They simply explained levels of automation in ten levels ranging from totaly human undertaking to total autonomy of the machine. A second issue is to decide the agent who is responsible for triggering of the adaptation of the interface. Either human or machine triggers the adaptation. Inagaka [32] gave a very good coverage of adaptive interface problems and taxonomies in aviation and human ergonomics domain.

Finally, the pace of the adaptation is very important in order to compensate for environmental changes, and not to lose consistency. Depending on the user's knowledge and perception level the adaptation should be able to explain itself or it should at least, be able to be understood by the user. The interface can also be allowed to take the initiative and react to an urgent and dangerous change in the environment. Such requirement may be necessary in nuclear plant supervisory control interfaces. Adaptation is not required to be smooth in this kind of abrupt change.

Inference Mechanism

Lastly, is the *selection of decision inference mechanism.* This is where the intelligence lies. In this part, the interface should infer from the data collected and change the mapping by itself. There are different ways to embed an inference algorithm into an interface. For example seven different methods were covered in Pazzani and Bilsus [48].

- Naíve Bayesian
- Nearest neighborhood
- PEBLS (Parallel Exemplar Based Learning System)
- Decision tree
- Rocchio's algorithm
- Perceptron
- Multi-layer neural network

All these methods are kinds of classifier algorithms based on user's habits, levels of knowledge, levels of workload or efficiency. Anything you chose from the triggering mechanisms covered in section 'Variables that call adaptation' should be inferred by a classifier. However, according to [56] and to the best of our knowledge there is no online adaptation mechanism. There are some exceptions to this in multi-media systems such as Seal *et. al.* [65] in e-mail filtering systems and Fiechter and Rogers's [16] adaptive route advisor. There are still other online adaptation schemes in the

data mining field but human ergonomics consists of little or no online adaptation schemes.

Chapter 3

Proposed Method

3.1 Introduction

Implementing intelligence into the interface was discussed in the beginning of 1990's. Automating all possible functions using Fitt's approach [24] has not turned out to be as desirable as expected. Consensus on total automation to avoid human error leads researchers to define automation techniques that neglect human factors and/or workload. Unfortunately this problem was overlooked in the early times since the importance was given to the efficiency instead of human emotions and environmental effects. As the automation problem was conquered in general, expectations were that, errors and accidents would decrease down to zero. In contrary to these expectations human errors did not decrease to zero and fatal accidents continue emerging. This in fact is mentioned in [9] as follows;

> " During the 1970s and early 1980s the concept of automation as much as possible was considered appropriate. The expected benefits were a reduction in pilot workload and increased safety ... Although many of these benefits have been realized, serious questions have arisen and inci-

dents/accidents have occurred which question the underlying assumption that the maximum available automation is ALWAYS appropriate or that we understand how to design automated systems so that they are fully compatible with the capabilities and limitations of the humans in the system. ”

Thereafter people tried to define adaptive/intelligent interface for complex task environments. We adapt the definition of Hancock [14] ,which was given for intelligent interface as follows;

> Intelligent interface is an intelligent entity mediating between two or more interaction agents who possess an incomplete understanding of each others' knowledge and/or form of communication.

Continuing to the same methodology, Hancock [14] has introduced model for interface. Cognitive interface is defined as abstract model of the interface and task, in the users cognition. Figure 3.1 gives the cognitive user interface model as given in [14].

Figure 3.1 shows the representation of intelligent interface. Adaptation can be employed in any of the three parts explained as follows.

- *Discourse input machine* is like the articulation part of the interfaces as explained in Abowd and Beale [1]. This part of the interface is responsible to translate the intentions of the user and map these intentions to operators of the system (Task machine).

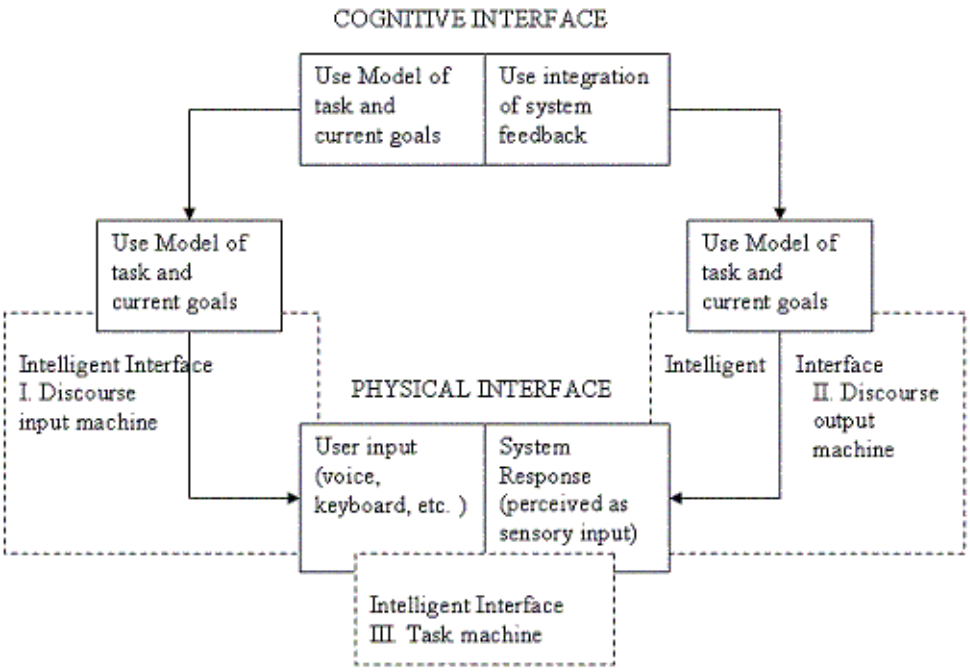


Figure 3.1: Interface Model (Hancock)[14]

- *Discourse output machine* acts as the mirror of discourse input machine and translate the next state of the system into something amenable by the user. Ideally this representation is consistent with the users cognitive model of the system. Thus the intelligent system reduces semantic distance by providing system output in a form that corresponds closely to the user's model of the task, and by allowing the user to express it in the language of that user model.

- *Task machine* is responsible to execute the task and return an output. If this state transformation in the task is close to cognitive representation of task model at the user side, we can talk about the intelligence of the interface.

Under the guidance of these definitions and previous literature survey we are going to attempt to define a generalized adaptation mechanism under some assump-

tions about the task and the user. Next sections will cover our motivation and the adaptation scheme.

3.2 Motivation

Adaptation of the interface consists of adjusting the form of the information transfer, transform the information content, alter the modes of information flow, and exchange the communication media. Although a fair amount of research was made in HCI domain; there is little or no examples of real-time adaptation of the interface [56]. In the automation domain adaptive interface aid was considered, but practical implementations in order to decrease/increase the mental workload of the operator were rare except Prinzel [33]. In Prinzel *et. al.* adaptation was manual and secondary task method was used for triggering the interface transformation.

General tendency for adaptation in complex task environments, such as in automation domain, is turning on or off the sub-tasks. For example adaptation, which was explained in Wilson *et. al.* [76], is turning on or off some parts of the multi-task operations [17]. Adaptation of this kind is either too constrained or not fine grained. Practical efforts are also falling short since no well-known evaluation of human preferences was defined. In addition to lack of evaluation function in the adaptation it is not very easy to define a framework for adaptation method itself.

Our motivation is to find a method that enables an interface to 'autonomously adapt' to a user. We define general framework for interface adaptation under some assumptions. Proposed method need to be more flexible then earlier counterparts in order to incorporate different evaluation metrics and adaptation schemes. We have

tried to implement our approach to Air Traffic Control (ATC) operation for proof of concept purposes. Empirical study showed that our method is capable of adaptation under uncertainty and decrease the semantic distance between physical and cognitive interface. Next section will elaborate on our framework.

3.3 Framework

We would like to introduce a generalized framework for adaptation of user interface. Our approach to the problem is finding the mapping between two spaces of interest Takagi [68]. These are psychological and parameter spaces. Here we define psychological space as level of preference and comfort using the interface for a given user. On the other hand parameter space is defined as the possible combinations of parameters related to interface that could be changed in order to increase preference value of the interface in the corresponding psychology space. We will split our framework into three parts in order to be consistent with the literature and ease of explanation. Next three subsections will cover these parts. General flow of the algorithm is given in Figure 3.2.

3.3.1 Interaction Evaluation Method

We will start this section with our definition of the psychological space and the parameter space. This definition is important in order to define evaluation method.

Definition 3.3.1. If $\mathcal{A}$ is a set of operators defined for the interface (interface model) and A is a subset of $\{A \mid A \subset \mathcal{A}\}$, and X is the operator's psychological space then $f\colon A \longrightarrow X$ is a mapping from interface to cognitive space.

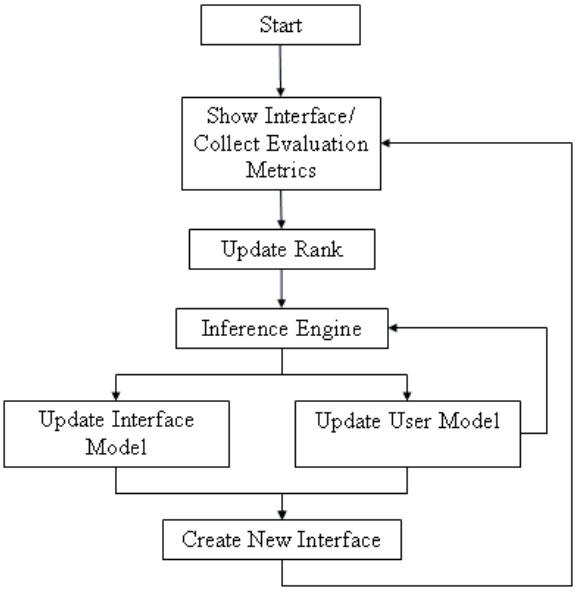


Figure 3.2: Adaptation flow graph

Generally $dim(\mathcal{A}) \geqslant dim(X)$ since human perception may not compare to fineness of parameter space of the interface. For example alarm tone of 1 KHz can alert the operator as much as 1.01 KHz can be. Takagi [68] proposed to use NN (Neural Network) in order to model f and this can correspond to user model in Benyon [7]. Actual mapping function that maps interface to psychology space is unknown and effected by personal, psychological, and environmental factors.

Assumption 3.3.1. f is a stationary stochastic mapping from parameter space to cognitive space. f is dependent on $\mathcal{A}$ and is assumed to span the cognitive space.

Assumption 3.3.1 states that depending on the application domain, $\mathcal{A}$ should be selected carefully so that resultant interface would correspond to a different region in $\mathcal{X}$ in order to be evaluated effectively. If we go back to our monotone alarm example, small difference between frequency would correspond to the same alert level in user

psychology space thus will not be able to span the whole space.

Even all of the above assumptions were satisfied, it is still not an easy problem to evaluate an interface since the only legitimate evaluator is human (user). First reason is, burden of evaluation is a problem since it causes degrading of vigilance and concentration loss. In addition to these factors there is no generic and proven way defined for acquiring this information from the user. Some metrics are defined to acquire this information such as psychophysiological, subjective and performance [51] metrics. These metrics have also been covered in detail in Section 2.2.3. Subjective methods such as NASA-TLX (Task Load Index) are not appropriate for online analysis since it requires user to respond to a set of questions after interacting with the interface. Psychophysiological methods [50, 58] are inconvenient for the user since most of them require connecting user to interface via some wires. Finally, performance metrics seems more appropriate, efficient, inexpensive for assessing user interaction level. It is all these reasons that lead us to use performance metrics as our evaluation method.

Performance data can be collected online during the user's interaction with the interface. This is the first phase of our algorithm as shown in Figure 3.2. As soon as an interface shown to the user, interaction begins. Some values such as goal accomplishment time, number of errors, number of methods used etc. can be used to infer the level of comfort and efficiency of the interface. These values can be fed to inference engine which will be covered in the next section.

3.3.2 Inference Engine

We have concluded last section saying that performance metric is selected to evaluate the interface. Collected data about the interaction is now available. Collection methods for performance metrics vary. Independent of collected data and collection method, this data is not dependable and often very noisy. "Noisy" here is used in order to emphasize the stochastic behavior of the user. For example the same interface can receive different scores in different times. We are using non-parametric approach to handle this problem. We decide using non-parametric methods first because any data collected, as a measure of performance should be regarded as in ordinal scale, which means, data can be ordered as better or worse. Data values of any kind cannot be regarded as interval scale since the relative distance between the values is not meaningful. The second reason we employ non-parametric methods is variability of the different scores should be brought closer. For example variance of number of errors can be two, and variance of time taken to accomplish a task can be hundred seconds. In order to be able to use this two entities together, their variance should be brought closer. One method proposed in non-parametric statistics is using rank values. This part was represented as re-ranking in Figure 3.2. It is the preprocessing of the data before it is fed to inference engine.

if we had perfect mapping function $f\colon A \longrightarrow X$ we could use it to find the best interface. However this data is not available and we are not trying to find the exact mapping function f, since it is too hard if not impossible. Instead, every new interface is ranked with respect to other interfaces which have been shown through

interaction. We propose this method in order to create the mapping by sampling different possible interfaces. This approach has been applied before [16, 41] but in a sense of dichotomy and adaptation was not continuous. Gervasio *et al.* [26] has used past information without considering the importance of the context during the user evaluation. Gervasio and Seth's [16] approach is very similar and tries to find a linear classifier to separate two classes which are labeled as good or bad. They have not considered the linear separability of these points. In our approach, however, changing the interface triggers re-ranking of all the interfaces that have been shown to the user up to that point in time. Re-arrangement of the interfaces helps to fine tune the topology of f . In addition to the improved performance we intent not to bother the user by explicitly asking his/her evaluation. Ranking can be applied to multiple metrics collected upon interaction and sum of ranks can be assigned as final evaluation of interface. So to speak in our approach new interface is considered to be a disturbance in the pool of interfaces and change the ordering of all the interfaces in the ranking scheme. Example usage of this method will be given in the next sections.

Some degree of forgetting is also applied to the model. Forgetting is a simple pruning of the earliest (temporal) interface from the pool of interfaces that has been shown to the user thus corresponds to a sliding time window during the evolution. This kind of pruning is chosen to be the forgetting factor in order to handle the quick changing characteristics of the user such as fatigue, vigilance, etc. We drop the oldest interface regardless of its fitness values. Our pruning approach is beneficial since it can adapt to high frequency changes due to clear cut forgetting about the past events. On the other hand this method also leads to drop candidate solutions regardless of

their potential usefulness and makes it even harder for the algorithm to find the best interface.

Ranking is done relying on the problem dependent features of the interface that has been agreed upon, at design time. Although these features are problem dependent, flexibility of ranking approach will not put constraints on the choice of the features. Equation 3.3.1 shows ranking formula where $R(.)$ is a monotonically increasing function from performance of the interface to the rank of the interface among other interfaces.

$$\begin{aligned} &R(A_{ij}) < R(A_{kj}) \Leftrightarrow A_{ij} < A_{kj} \ \forall i,j,k \in \Re \\ &where\ i,k = 1,2,3,...,n\ is\ the\ interface\ number \\ &and\ j = 1,2,3,...,m\ is\ j^{th}\ metric \\ &associated\ to\ interface\ i,k \end{aligned} \tag{3.3.1}$$

New interface will be shown to the user and desired measurements are collected in a given period of time. Results will be ranked with respect to the earlier interfaces and rank value will be used as fitness value of the interface Equation 3.3.1.

Inference engine will generate new interface with the help of user model. Inference can be seen as an optimization problem, search space being the parameter space of the interface and objective space being the psychology space of the user. Relation between these two entities is the calculated ranks. Different optimization techniques can be applied to this problem. Complexity of the problem arises from the lack of well-defined objective function (user) and noisy evaluation. So we choose to use GA (Genetic Algorithm) to do the search due to the following two reasons. First reason to choose GA is to take advantage of the stochastic search method employed, while the second reason is the flexibility of GA search in difficult problems such as

discontinuity and constraint satisfaction [19]. Noise rejection ability is also important since evaluation is very noisy. Although we try to keep the variance of the evaluation low by using ranking method defined above, we will never be able to get the same result for the same interface due to the stochastic behavior of the user. Evaluation of interface is defined on integer values instead of a continuous space which leads to discrete surface. This explains the reason of the discontinuity that leads us to use GA algorithm. Constraints in search space are present for almost all features. Considering all of these characteristics, GA is regarded as a good candidate for the search.

3.3.3 User Model

Up to this point we have defined interface evaluation. Collected scores after the interaction is sent to ranking algorithm. GA and user model are used as the inference mechanism. On the other hand, power of GA comes from its being population based approach. In our case finding the fitness to any possible solution is very expensive. We can evaluate only one interface at a time since we don't want to bother the user. Although we always carry past n individuals (interfaces) and create a temporary population for GA, we are not able to test more than one individual (interface) in the next iteration. We have used probabilistic approach to overcome this problem. Definition 3.3.2 specifies user interface model

Definition 3.3.2. User model which was used to approximate f in Definition 3.3.1 is a multivariate Gaussian with $U = G(\boldsymbol{\mu}, \boldsymbol{\sigma})$. $\boldsymbol{\Theta} = [\boldsymbol{\mu}\, \boldsymbol{\sigma}]$ is the model parameter. Every interface is subject to pre-evaluation and their score is calculated as $P(A =$

$good \mid \Theta = \theta)$ where $\{A \mid \ A \subset \mathcal{A}\}$.

At the next iteration, candidate interface is chosen according to this probability measure. Before going into more detail we would like to clarify why we need to define a mapping for evaluation of interface components. First, we assume user is able to interact with one interface, unless the designer is using the secondary task performance score to trigger adaptation. Under this assumption we can evaluate only one interface in a given time period. However GA (Genetic Algorithm) can create more than one offsprings (interface models). In order to evaluate the interfaces without even showing to the user we need an estimator of user preferences which was defined as the user model.

Table 3.1: Pseudo code for adaptation algorithm

```
InitInterface(); Do {
    collectData();
    updateUserModel();
    If (timerTriggered()) then
        reRankAllInterfaces();
        \\GA part
        selectParentInterfaces();
        doCrossOver();
        doMutation();
        \\End of GA part
        if MOD(numIterations) = maxPopLength then
            \\Prune the oldest interface
            pruneInterfacePop();
            reRankAllIntrafaces();
        end if
    end if
} while (numIterations < maxIteration);
```

Pseudo code for the algorithm is given in Table 3.1. *updateUserModel* subroutine consist adaptation of multivariate-Gaussian function. After defining our adaptation

method we can talk about selecting interface model in the interface population. An individual is defined by the combinations of features dependent on the problem domain that can be selected to adapt the interface to the current user. Possible interpretation of this will be given in the next section. Due to flexibility of GA algorithm these features can be chosen as discrete or continuous variables.

3.4 Test Bed (ATC)

In this section domain specific components will be defined and separated from the general framework. A better understanding of the algorithm will be pursued through an example. Since a generalized framework for online adaptation is the ultimate goal, we have implemented our approach using an ATC (Air Traffic Control) test bed. We have used this example because recent studies have shown that adaptation of this type of interface would greatly improve the performance [33]. In the study by Wright *et. al.* [33], interface shows moving targets (air planes) to the user. User is supposed to clear the targets by clicking on them before they move beyond the display. Contrary to what has been implemented in [33], we did not employ an impaired vision with a moving window, or use two-step clearance method for each air plane. However, we have used the number of aircrafts presented and number of the aircrafts cleared as performance measure.

We will start with domain specific components of the interface model as defined in Subsection 3.3.1. Taxonomy for domain specific adaptation techniques defined in [39] and among those used in this study is given in Table 3.2.

Table 3.2 shows interface features we have implemented for our test bed. Numbers

Table 3.2: Interface features for the test bed

1. Information Acquisition
   - Change update time [0-50]
   - Activate sound crash/shoot/both [0-3]
   - Automatically clear planes (Automation) off/on [0-1]
   - Mouse pointer arrow/cross/nodrop [0-2]
2. Information Analysis
   - Show current score [0-1]
   - Activate right click off/on [0-1] when right click locate the possible crash
   - Number of planes [1-maxNum]
3. Action Implementation
   - Add List Box/(do not) [0-1] integer adding list box automatically activate linking
   - Kill planes by right click off/on[0-1]
4. Decision Making
   - Rank list box by entry sequence/speed/position/possible accident [0-3]
   - Small white/small red/large white/large red plane appearance at possible crash site [0-3]

in squared brackets are possible values that feature can assume in accordance with their linguistic explanations. Formally in GA literature, it is called genotype of the design. Resulting interface (phenotype) is shown to the user for interaction. Now that problem dependent features selected, our genome to be used under evaluation will be possible combinations of these features. For example $\{A \mid A = [30\ 1\ 1\ 0\ 40\ 1\ 0\ 0\ 0\ 1] \in \mathcal{A}\}$ interpreted as

- Update interval will be 30 sample times
- Activate sound for crash of the air plane

- Automation is on
- Mouse pointer shape over the plain is arrow
- Don't show current score
- Number of max plains is 40
- Locate possible crash site on right click
- Don't add list box
- Right click clear planes from the interface
- Small white plane appearance at possible crash site
- Rank list box by time sequence of entry (not relevant here because of the previous rule)

All these types of features (genomes/interfaces) will be subjected to adaptation under genetic operations. Resulting children will be evaluated by the user model U as explained in Subsection 3.3.2. Depending on candidate interface's conditional probability it will be selected to be shown to user for a predetermined interval of time.

Multi-variate Gaussian approximation of user is also updated through evolution of the interfaces as given in Table 3.1. One of the model parameters $\boldsymbol{\mu}$ is adapted by using differential perceptron rule as given in Equation 3.4.2 and $\boldsymbol{\sigma}$ is adapted by dynamic equation as given in Equation 3.4.3.

$$P_n^a = hardlim(P_n^a - P_n)(\lambda P_{(n-1)}^a + (1-\lambda)P_n) \tag{3.4.1}$$

$$\boldsymbol{\mu_n} = hardlim(P_n^a - P_n)(\gamma \boldsymbol{\mu_{n-1}} + (1-\gamma)(\boldsymbol{\mu_{n-1}} - \boldsymbol{A_{n-1}})/(P_n^a - P_n)) \tag{3.4.2}$$

$$\boldsymbol{\sigma_n} = hardlim(P_n^a - P_n)(\tau \boldsymbol{\sigma_{n-1}} + \boldsymbol{\delta}) \tag{3.4.3}$$

where P_n^a is weighted average performance throughout the interaction. Corresponding $\lambda < 1$ is the filter coefficient. P_n is the last performance at time n. $\boldsymbol{\mu_n}$ is the mean of the gaussian at time n and the corresponding $\gamma < 1$ generally called learning constant. $\boldsymbol{\theta_n} = [\boldsymbol{\mu_n}\, \boldsymbol{\sigma_n}]$ is the model parameters as well as $\{A_{n-1} \mid A_{n-1} \subset \mathcal{A}\}$ is current interface model. $\boldsymbol{\sigma_n}$ is diagonal elements of covariance matrix of multi-variate Gaussian, cross-correlation effects of parameters on the mapping was not considered for simplicity. $\tau < 1$ is the convergence rate of the variance. δ is used in Equation 3.4.3 to preventing variance going to zero and create singularity. Steady state value of the variance can be determined for every parameter in the interface using Equation 3.4.4. For better understanding of the user model adaptation please refer to Appendix D.

$$\begin{array}{l} \lim_{n\to\infty} \sigma_n^j = \frac{\delta_j}{(1-\tau)} \\ j = 1, 2, 3, ..., m \textit{ is } j^{th} \textit{ diagonal element} \\ \textit{of the covariance matrix associated to model} \end{array} \tag{3.4.4}$$

and finally $hardlim(.)$ is defined as

$$hardlim(x) = \begin{cases} 1 & x > 0 \\ 0 & otherwise \end{cases}$$

Equation 3.4.2 is similar to instar rule for associative learning and non-projectional

learning algorithm scheme explained in learning automata [69]. As it is easy to inspect from Equation 3.4.2 our user model is nothing but a localization function for Genetic Algorithm. It is similar to another method called SA (Simulated Annealing)[43]. Initially GA was free to explore the subset of the parameter space since initial variance was set to be a big number. As iteration goes on user model limits the search space since variance converges to steady state value as given in Equation 3.4.4. Since $P(A = good \mid \Theta = \theta)$ would be very small for the interfaces away from $\boldsymbol{\mu}$ they would not be selected as candidates for the next iteration. This will naturally lead to exploration of the search space in contrast to exploitation. So τ can be regarded as the 'cooling rate' in SA (Simulated Annealing).

At this point we are able to assign the probability values for each interface using user model U. One of these interfaces would be shown in the next iteration according to their probabilities. New scores are used for fine-tuning the evaluation function through re-ranking mechanism. Many different measures can be applied in order to evaluate the interface.

- Psychological measures: hearth rate (ECG), respiration rate, pupil diameter etc.
- Performance measures: error rate, mouse click etc.
- Subjective measures: questioners

These measures [59] are also problem dependent aspect of the proposed algorithm. We have selected performance measures to evaluate different interfaces. As we have

Table 3.3: Ranking procedure

Interval Number	# of mouse clicks/Rank	# of unsuccessful mouse clicks/Rank	# of plane show / Rank	# of destroyed plane / Rank	Total Rank
1	55 / 2	5 / 2	60/1*	3/1.5**	6.5
2	43 / 1	3 / 1	55/2*	3/1.5**	5.5

* Higher number of planes is better than small number of plains
** Equal numbers are assigned middle ranks

mentioned above we can select more then one measure to evaluate the interface due to the flexibility of the ranking method. We collect the following data set for each interface for a fixed interval of time;

- Number of mouse clicks
- Number of unsuccessful mouse clicks
- Number of air planes presented to user
- Number of air planes destroyed before successful clearance

We have also collected total mouse shift but it was not very informative for ranking. We are using within ordering between measures and assign a rank to an interface adding up the total rank assigned to one interface. Table 3.3 presents an example to explain methodology used here.

Each row is assumed to be an interface represented by $\mathbb{R}^n$ as explained before. Reason for ranking procedure is to reduce variability of data. Also as claimed before, data is in ordinal scale, and values can be interpreted only as bigger or smaller relation. Assume now a new interface was presented to user, re-ranking will be applied every

Table 3.4: Re ranking procedure

Interval Number	# of mouse clicks/Rank	# of unsuccessful mouse clicks/Rank	# of plane show / Rank	# of destroyed plane / Rank	Total Rank
1	55 / 3 †	5 / 2	60/1	3/2.5 †	8.5
2	43 / 1	3 / 1	55/2	3/2.5 †	6.5
3	45 / 2	6 / 3	54/3	2/1	9

†Updated after the addition of the new interface (compare with Table 3.3)

time a new data is calculated. For a new interface this procedure is demonstrated in Table 3.4.

For now we are only using total ranks as fitness measure but these ranks can also be used in statistical analysis. After new ranks are obtained we can use their values in GA evaluation. Functions given as domain dependent in Table 3.1 are selectParentInterfaces(), doCrossOver() and doMutation() in ATC test bed is as follows. Parent selection algorithm implements roulette wheel selection. In this method fitter individuals are selected with higher probability. Crossover operation is uniform. Given two parents we create one child by selecting one of its parent's genes at each location with equal probability. Finally mutation operator is applied. In continuous gene locations we use real value mutation, and random assignment is used in locations where discrete variations are assumed.

3.5 Automated User

In addition to pseudo code given in Table 3.1 we like to introduce here the extra component in our design. Since data collection is a very expensive process using real users, we have coded an automated user for the initial testing. This automated user

simulates all the assumptions listed below for the real user.

Assumption 3.5.1. *User is a stochastic process* : Given an interface user has a higher probability to react correctly to events if it is one that better fulfils his/her preferences. This is done by defining a fixed n-tuple for the automated user as a preference vector and based on normalized Euclidean distance of the current interface vector to preference vector; probability of clearing a plane is increased or decreased.

Assumption 3.5.2. *Response time is limited* : User has limited cognitive and motor abilities. Our automated model can react in discrete time instances and total number of clearance attempt is fixed in a given time span.

Even our automated user is a naïve approximation of human it meets basic requirements in psychology and HCI domain. Information theoretical model [28] defined for human cognitive skills is one of the underlying structures of our assumptions.

Automated user works as follows. We define an preferred interface parameter vector for the automated user. This vector is known to automated user in order to evaluate the interface. Automated user calculates the Euclidian distance between its preference parameter vector and current interface model A and try to simulate a click action for a random air plane with success probability directly related with this distance. Using this method we satisfy both stochastic behavior and the limited cognitive-motor capabilities of the real user

Chapter 4

Results

In this Chapter we will present the corresponding results for two experiments, the automated user and the human user respectively. Results from the automated user is given in order to show the power and extendibility of the design. Yet, experimentation on human user is given as a proof of concept. We have intentionally controlled the search space to limit the degree of user burden. The second part which discusses the real user was integrated to the thesis to show that the validity of our assumptions and viability of the algorithm.

4.1 Results from Automated User

Automated user is designed such that, it bears a preference for the desired interface. This preference vector is in R^n and has the same dimensions with the interface model (see Section 3.5). Automated user randomly select an air plane and attempt to clear it. Probability of the automated user for clearing the plane is dependent on the Euclidian distance between the interface model A and automated user preference vector.

Given the automated model we are trying to show that we can predict the per-

formance of a stochastic process (automated user). We have done two different tests and our results are explained under two categories. First category of experiments are used to validate ranking approach for evaluation of interfaces. Two different kinds of interface models were employed in these sets of experiments. Two interfaces differ in the sense of the Euclidian distance to the preference vector. These models were presented to the automated user without adaptation. Ten different sets of data were collected for each. We test the hypothesis if distribution of the populations from which these samples came from, is the same. We can rephrase the hypothesis as follows, is the ranking procedure a consistent measure for different types of interfaces or not. We have used Kolmogorov-Smirnov (KS-2) test to compare the distribution of two samples from the same interface model. Resulting test statistics value for m = 20, n = 20 was K = 0.2 the null hypothesis was accepted with observed significance level (p-value = 0.7710) for preferred interface, and K = 0.2105 the null hypothesis was accepted with observed significance level (p-value = 0.7415) for the interface with larger Euclidian distance. Saying that data values related to the same interface in different time intervals came from same distribution (same user has evaluated the interfaces). Second test for the first category of experiment was to confirm if two different interfaces can be distinguished for better or worst using ranking procedure. Data was ranked using all values (preferred and not preferred interfaces together). Finally we have checked if better interface's error distribution has a smaller mean than that of the worse interface. We have used t-test to see if difference in means is significant. We have used Lilliefors test for goodness of fit to a normal distribution on both sample data before we use data in t-test. Lillliefor test accept null hypothesis

with p-value = 0.1096 and p-value = 0.1454. Finally two tailed t-test reject null hypothesis (distributions are same) with p-value = 3.6637e-014. Our results show that we can use ranking scheme for estimating the relative goodness of the interface.

Second category of experiment made for adaptation of interface under these assumptions. At this point we have developed a test procedure to see if we can adapt the interface for different users. Starting interface parameters set to be [50 0 1 0 40 0 1 0 0 0 1]. First one thousand and eighty iterations were evaluated by automated user with preference vector being [10 1 1 0 40 0 1 3 0 2 0]. At the instant of one thousand and eighty one, preference vector for the automated user is converged to [50 3 0 1 40 0 1 1 2 0 1]. This is preferred to see if changing user as a simulation would be recognized by the interface. Figures 4.1 and 4.2 shows the results.

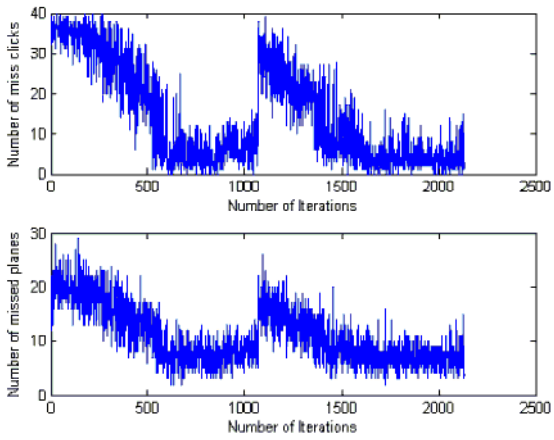


Figure 4.1: Performance metrics

Figure 4.1 shows the improvement in error results above. We have collected this data, while ranking procedure and GA was running to improve this performance. Results were given for a total of 2,160 iterations. This means we have evaluated total of 2,160 interfaces, 1080 for each preference vector. We may not employ a human

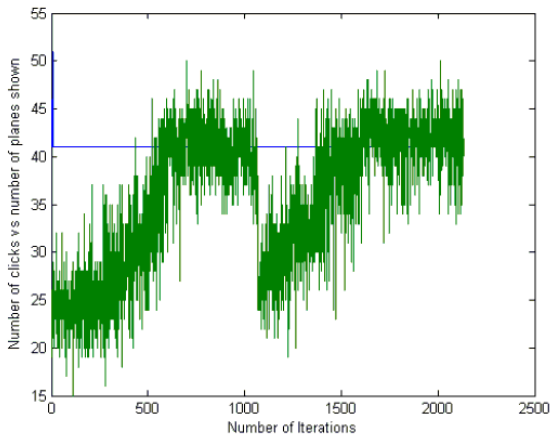
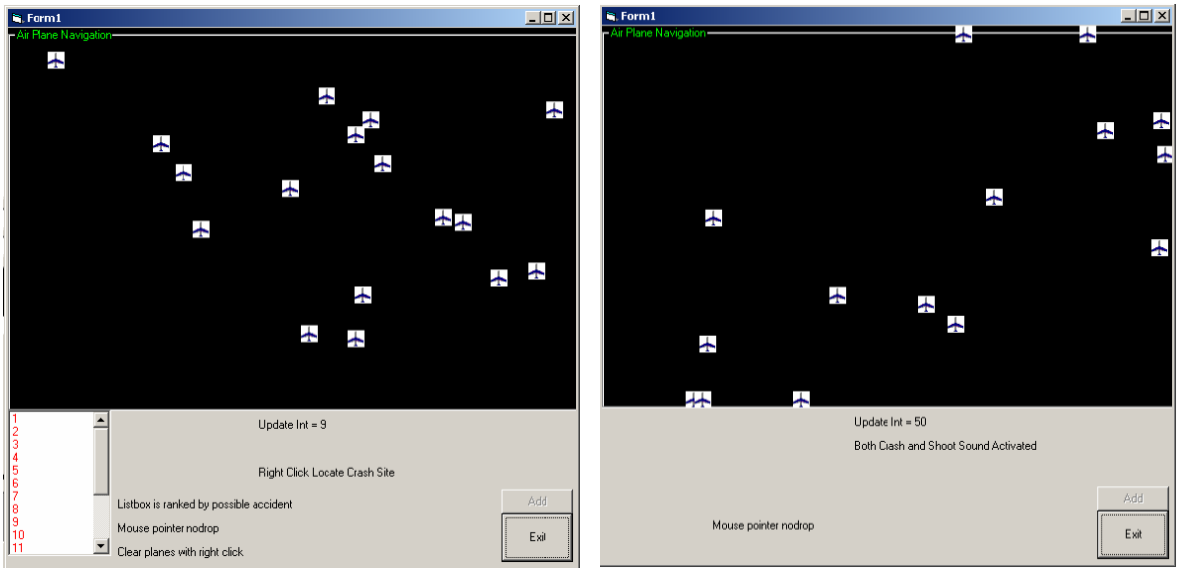


Figure 4.2: Number of shown planes

user for this amount of iterations since every interface would take one minute to be evaluated and if real human was used in the experiment total experiment time would result in more than sixteen hours of evaluation. This clearly shows why we implement a time constraints for real user and why we have designed automated user for initial test phase. Yet our results can be validated for prolonged time in daily usage. We can easily see time of switch from one preference vector to another in both Figures 4.1 and 4.2. In each case program was able to catch the difference and converge to preferred interface characteristics. Figure 4.1 shows that number of planes shown cumulated at around forty one. This results from our assumptions about the automated user's limited cognitive and motor abilities.

We should also add that, in a given time we are storing only forty different interfaces collected in a consecutive temporal intervals of interaction. This was accounted for quick changing characteristics of the user such as vigilance, attention and etc. Every time a new interface is created we delete the oldest interface from the population so to maintain a fixed number of interface population. Figure 4.3 shows different



(a) 1000th iteration (b) 2000th iteration

Figure 4.3: Snapshots of interface appearance at different iterations

snapshots during the evaluation. The left figure is collected after interface converged to the first user model while the right figure is taken after interface converged to the second user model.

Dynamic changes such as update interval and action implementation through adaptation cannot be shown in Figure 4.3. We also like to see if we were able to find the preference of the automated user in psychological space.

Euclidean distance between interface and preference vector was calculated in Figure 4.4. Figure 4.4 show three things.

1. Ranking method defined above as a mapping strategy is a valid mapping between feature space and psychological space.

2. Stochastic behavior can be modeled using this technique and the interface adapt to user preferences in order to enhance performance of the human-computer

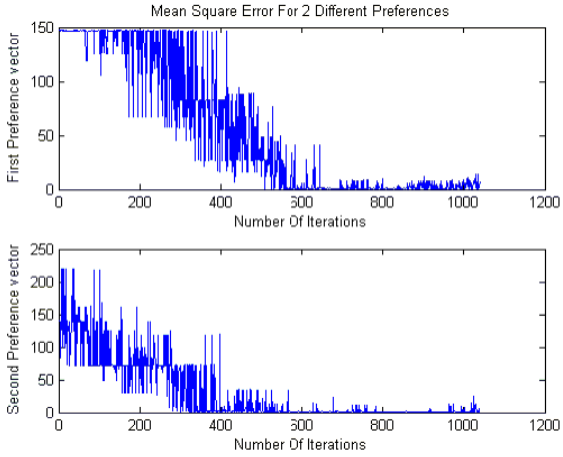


Figure 4.4: Euclidean distance between interface and preference vector

system.

3. User model defined before was effective in localizing the search since at the end variance of the interface is very small.

Appendix A also gives statistical distributions of interface parameters in order to show the convergence of algorithm statistically.

All of these improvements were established using least knowledge about the domain and the user. Evaluation for GA was done only for one interface for a given period of time. Interface has not interrupted the user and asked to give preferences between couple of interfaces. All processes was handled on-line. Improvements to this technique are possible and will be outlined in the summary. But the best part of this method is that its flexibility is not inversely related to its feasibility. That is you can easily incorporate more explicit domain knowledge to guide the search and still your interface will be feasible in the sense of collecting extra information and

ease of implementation. Our method is also highly scalable for more complex domain interactions.

4.2 Results from Human User

This section is dedicated to the results reported by the human users. The main purpose of this section is to show the validity of the approach in real life situations. However there are some differences between the interface model used under this section and that in Section 4.1. We have employed a smaller search space in this section. The number interface parameters are fewer than that of the automated user. There are two reasons for this distinction. The first and the most important reason concerns the user's burden. Real time adaptation lasts approximately half seconds for every evaluation. We have collected eighty data points for eighty iterations to be able to reason about the adaptation. Simple calculation shows that this results in thirty to forty minutes. It is hard for the user to sit in front of the computer and interact with the interface longer than that amount of time. However prolonged interaction, which is the case for real environments, may lead to the convergence of the search in the case of the bigger space and this was clearly shown in Section 4.1. The second constraint on the number of the interface parameters is given in the Assumption 3.3.1.

Our interface parameters are selected using some background knowledge from the HCI domain. First block of parameters are related to peripheral vision. Interface switches between four different types of plane representation. These are small size white planes, small size red planes, large size white planes and large size red planes. These parameters are coded to the genotype as 0,1,2,3 respectively. Size and color

parameters are changed when plane reaches a predefined critical location on the interface. Use of color is rooted from the well known fact about the peripheral vision of the human. Ergonomics field has shown that vision is sensitive to changes in the vicinity of the peripheral vision. Change of the color supposed to alert the user for the endangered plane which is on the edge of the destruction. Reason for changing the size of the plane comes from the HCI field and called Fitt's law. Fitts' Law is a model to account for the time it takes to point at something, based on the size and distance of the target object. Formula is given as $T = klog2(D/S+0.5), k \approx 100msec$ where T is the time to move the hand to a target D is the distance between the hand and the target and S is the size of the target. According to this formula increase in size will decrease response time hence lead to improvement in performance.

Second block of parameters are the update interval for the plane positions. The assumption behind the different update comes from the cognitive science. If plane positions are to be updated in every change in their positions, movement will be smooth and will not rely on short term memory hence decrease cognitive load but it will be harder for the user to click on the planes due to the rapid change of the plane position. On the other hand slower update of the plane position will give more response time to the user, yet result in a higher cognitive load because the estimation of the next place of the plane will require some mental processes. Update time spans an interval [1-20] and is coded accordingly.

When we put these two together we will have search space of eighty to be evaluated in a small amount of time and iterations. These parameters are also selected because they may satisfy Assumption 3.3.1. Our genome to be used under evalua-

tion with these features will be possible combinations of these features. For example $\{A \mid A = [10\ 1] \in \mathcal{A}\}$ interpreted as

- Update interval will be 10 sample times
- Change plane appearance to small size and red at possible crash site

Results will be given under two categories. First category will give the convergence of the user model and performance value as well as interface shown to the user. We are not able to give the distance between interface model and user preference since user preference vector is not available.

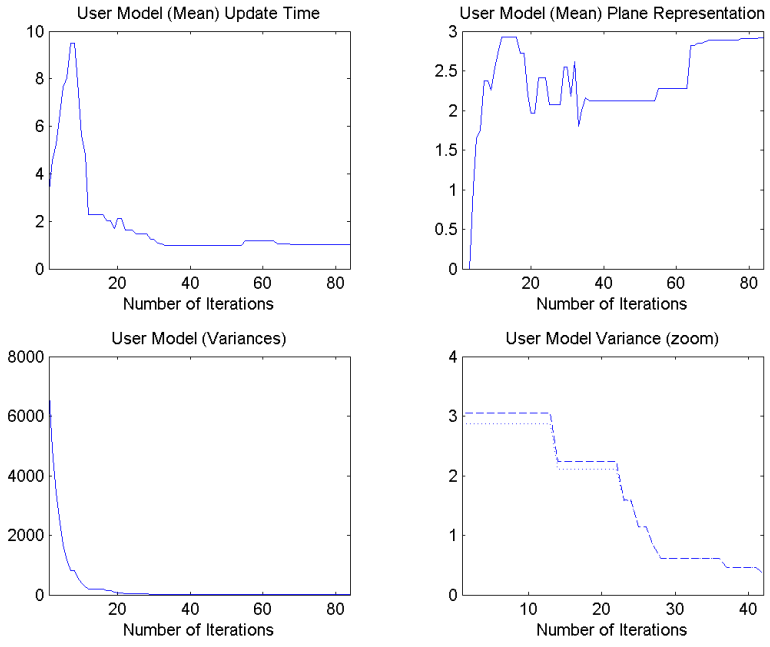


Figure 4.5: User model convergence rate

Figure 4.5 implies that user model was able to converge to a value where our intuition about the best interface suggests. Upper left figure shows the update time.

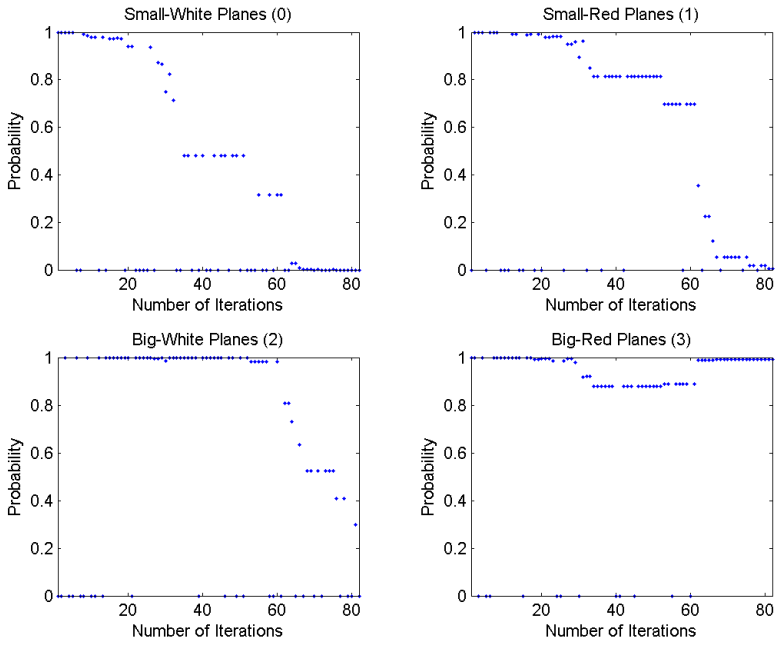


Figure 4.6: Probabilities assigned by user model to each of the parameters

Update time converged to one. Upper right portion of Figure 4.5 represents plane representation. Interface with the large sized planes was better in response time since they are taking advantage of the Fitt's law. User model converged to interface model with a large plane and final variance of the user model converged to a small value. We can easily see the variance for both update time and the plane representation from the bottom right portion of Figure 4.5. Dotted line is variance of update interval and dashed line is the variance of plane representation. Final value of the user model shows that user model has a bias to large size red planes instead of small size red planes which is also logical since color and size together improve both response time and accuracy. Figure 4.6 shows the probabilities assigned to candidate interfaces. It is easier to see from this figure that preference assigned to small size planes is getting

smaller and smaller while preference given to large size planes preserves its initial level. The reason for equal preference given to every interface in the beginning phase was the large variance of the user model. Due to large variance there was no bias to any given interface.

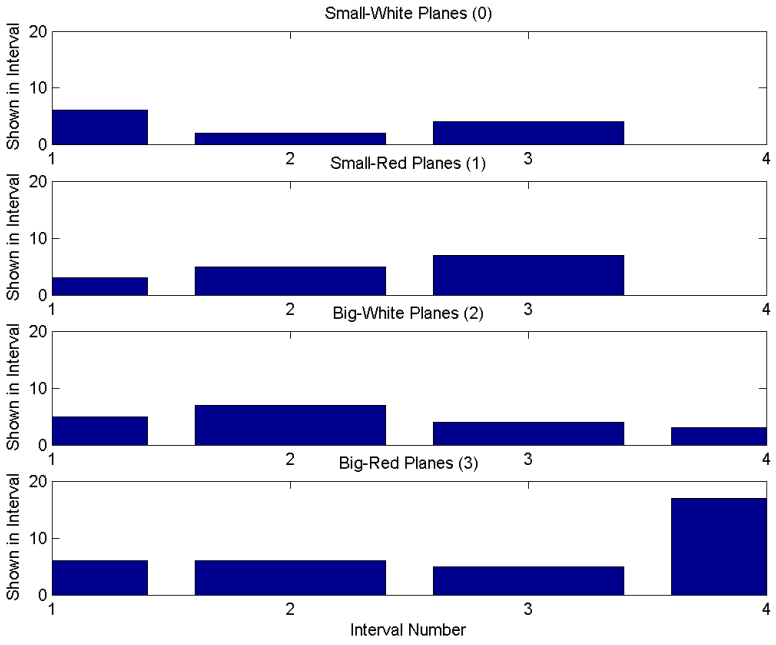
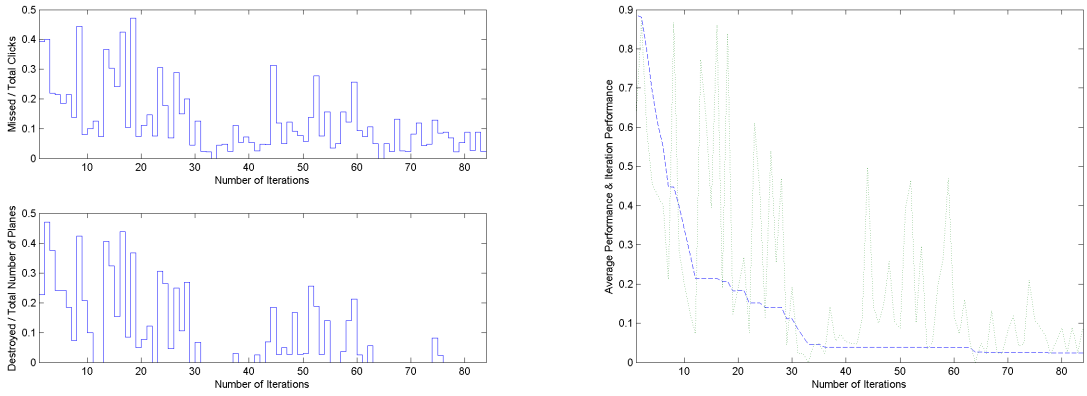


Figure 4.7: Statistical distribution of parameters at every 20 iterations

In order to show that convergence of the user model was able to localize the search we show the statistical distribution of the interfaces. Figure 4.7 can be interpreted in two ways. If we look at Figure 4.7 horizontally we can see that how many times a given parameter is shown to the user throughout the interaction. This shows the bias of the user model for a given parameter throughout the adaptation and is consistent with the finding in Figure 4.6. If we look at Figure 4.7 vertically we can inspect the distribution of the shown parameters. In both cases the proposed algorithm was able



(a) Error ratios during the interaction

(b) Average performance

Figure 4.8: Performance index collected during interaction

to converge to interface type with large size planes. Figure 4.8 depicts the performance index collected during interaction. Bottom part of Figure 4.8(a) is the ratio of the destroyed planes to shown planes. Top part of Figure 4.8(a) shows the ratio of off-clicks to total clicks. It can be seen that by the end of the adaptation bottom part of Figure 4.8(a) converges to zero (no destroyed planes). Figure 4.8(b) depicts another perspective of the evaluation. Results in Figure 4.8(a) are added and minimum points are connected with a dashed line. Change in dashed line indicates update of the user model as explained in Equation 3.4.1. This result shows that interface adaptation was successful to decrease the error rate and eventually converged at least to some local minima. Figures related to update interval parameters are given in Appendix B.

Second category of results are the subjective evaluation of the adaptation supplied by the users after the interaction with the interface. Questioner used for this evaluation is given in Appendix C. We have done statistical tests using the subjective results assessed from the questioners. These results correspond to the results of the

distance between the preferred interface and the interface model, shown in Section 4.1. We have employed seven users individuals in order to test the algorithm.

First two questions were meant to measure the difference between user preference and user model. We have calculated the correlation coefficient between subjective ranking of user and ranking of the user model U. Correlation coefficient calculated for plane representation is 0.93 and for the update interval is 0.89. This shows that user model is capable of assessing the user preferences. Questions three and four were asked to retrieve the subjective performance index. Results are given in percentiles. For question three, mean and the variance of the results were μ=4, σ^2=0.8 respectively and for question four they are μ=3, σ^2=1.6. Upper quartile for question three is 5 and for question four is 4. These results show that human users are affected by the adaptation and parameters were able to span the user psychology space. Questions five and six are designed to asses the consistency of the adaptation. These questions were asked because it is important that adaptation is smooth and consistent, otherwise it may distract the user and cause new problems instead of fixing the old ones. For question five mean and the variance of the results were μ=2.8, σ^2=1.37 respectively and for question six they are μ=3.8, σ^2=2.17. Upper quartile for question five is 4 and for question six is 5. Results for question five and six depicts that adaptation is smooth and easy to handle. Finally question seven was asked to collect more data about importance of the parameters. 2 users think that update interval is more important than plane appearance (5 is visa versa). Over all subjective evaluation shows that proposed algorithm was able to improve the interaction performance and adaptation was not disturbing.

Chapter 5

Conclusion

It is the ultimate goal of this thesis to find a better way to incorporate adaptation in the domain of intelligent interfaces. The lack of flexibility was the primary reason to implement adaptive interfaces. Current adaptive interfaces were mostly based on rule base systems and adaptation was not flexible.

With these facts at hand we have tried to define adaptation in mathematical terms. Solution to the problem was formulated in general as an optimization problem. To the best of our knowledge this is proposed first time in the literature. We have also defined a methodology to attack the problem of interface adaptation. These steps were to define evaluation metrics, selecting parameter space and finally defining inference method and user model. We argue that our method is a generalized framework since most of the problem specific components can be handled by the proposed method. In addition, any heuristic that may improve the adaptation performance can be easily captured by adding it as a different evaluation module.

We have developed a software platform, namely ATC (Air Traffic Control) tool for empirical proof. Using this system we have given problem dependent aspects of our algorithm.

We have shown in this thesis that non-parametric methods can be employed for evaluating the interface under uncertain environments. Especially if human is in the loop then statistical methods have shown better performance estimating the user characteristics. Assumptions for the real user was implemented in the automated user and two sets of data were collected from automated and human users. Results were consistent and show the proposed method can handle real time situations with large search spaces.

Importance of this work is to define a general framework for adaptation in the ill-defined situations. In our case we have unobservable system states of the overall system namely user preference. User is stochastic so as the fitness evaluation. There is no explicit objective function that interprets user performance perfectly. Overall uncertainty in the system is very high even in a very simple implementation.

Flexibility of the algorithm allows us to handle any kind of heuristic or domain dependent knowledge. Ranking scheme will handle the situation as if another data is collected. In addition to that, other methods can be used to solve this optimization problem. However, GA is best fit for this kind of adaptation due to its ease of use in highly constraint problems. GA is scalable to higher dimensions. This is important because in more complex systems feature space can grow exponentially with the number of features.

Finally we have to mention that this study is by no means a complete solution to adaptation problem. Our approach should be further justified before it can be accepted by the HCI community. Strength of this approach may prove important if it is implemented in more than one problem domain.

Simplicity of the method may also lead to a better understanding of human behavior and preference dimensions. Simple experiments with less dimensional feature spaces can be exploited using paired test in non-parametric statistics and efficiency of the method can be improved. Different features of interface can be used in order to understand the dimensions of psychological space.

More involved and complicated architectures can be built upon this idea. This approach also enables interaction of multi-disciplinary knowledge into the field of HCI. As shown, statistical approach, machine learning and rule base systems can complement each other and can easily be incorporated into a general framework.

Reusability of the method is accomplished by separating problem dependent features from the inference engine. Use of GA enhances the flexibility of the algorithm. It is this aspect that may lead to much improved cooperation between domain experts, HCI experts and software developers.

Chapter 6

Future Perspectives

First of all, we are going to test the proposed adaptation model on real users for prolonged times and try to see if subjective rankings are correlated with our ranking scheme and successful adaptation can be handled in real-time complications. This is an important step before we can advance any further. In order to be able to test the method on real user we have to find ways to increase the convergence rate. Seventeen hours of use is not practical for ordinary subjects since they are not employed to work on the interface for a long time. However this still does not undermine the importance of this method since prolonged daily usage is amenable by professional users.

Next priority to be considered should be defining methods that may lead us to understand parameter selection for the interface adaptation. As we have mentioned earlier, no matter how well the adaptation method is, different interfaces should map into different points within the psychological space in order to be evaluated efficiently. This assumption forms the basis of our approach. Actually, this assumption is hidden in many applications but not mentioned explicitly.

Assigning a subjective score to an interface is another point that should be investigated. We have used performance metrics in this study; however, we still need

better methods to evaluate human performance and comfort level. Signs of better metrics are emerging from neuro-psychology, cognitive science and computer science.

Another important aspect of adaptive interfaces is consistency and user trust to the interfaces. Our approach cannot be considered as consistent if compared to rule base systems. Effect of continuous adaptation is another area of research. User comfort with changing media and unpredictability of the interface can create more problems than it is supposed to solve. There may be some methods that can embody our approach and more conservative adaptation implementation.

Last but not least, we can tune the parameters of adaptation that were defined in our approach. We have designed our approach and try with some specific parameters. Different parameters may lead to different performances. Fortunately, our approach is intuitive and easy to manipulate. Different exploration and exploitation strategies can also be incorporated into the algorithm.

Bibliography

[1] G. D. Abowd and R. Beale, *Users systems and interfaces: A unifying framework for interaction*, HCI'91: People and Computers VI (Diaper D. and Hammond N., eds.), Cambridge University Press, Cambridge, 1991, pp. 73–87.

[2] L. Adelman, S. L. Miller, and C. Yeo, *Testing the effectiveness of icons for supporting distributed team decision making under time pressure*, Transactions on Systems, Man and Cybernatics - Part A **34** (2004), no. 2, 179–189.

[3] V. Aleven, O. Popescu, and K. R. Koedinger, *Towards tutorial dialog to support self-explanation: Adding natural language understanding to a cognitive tutor*, 10th International Conference on Artificial Intelligence in Education, 2001.

[4] J. R. Anderson, *Rules of the mind*, Erlbaum, Hillsdale, NJ, 1993.

[5] Schneiderman B., *Direct manipulation for comprehesible, predictable and controllable user interfaces*, Proceedings of 1997 International Conference on Intelligent User Interfaces, ACM, 1997.

[6] L. Balint, *Adaptive human-computer interfaces for man-machine interaction in computer integrated systems*, Computer Integrated Manufacturing Systems **8** (1995), no. 2, 133–142.

[7] D. R. Benyon and D. M. Murray, *Applying user modeling to human-computer interaction design*, Artificial Intelligence Review **6** (1993), 43–69.

[8] S. Bi and G. Salvendy, *Analytical model and experiemntal study of human workload in scheduling of advanced manufacturing systems*, The International Journal of Human Factors in Manufacturing **4** (1994), 205–234.

[9] C. E. Billings, *Aviation automation: The search for a human-centered approach*, p. 4, Lawrence Erlbaum Assoc, New Jersey, 1997.

[10] T. Bohnenberg, A. Jameson, and K. A. Butz, *User acceptance of a decision-theoretical location-aware shopping guide*, International Conference on Intelligent User Interfaces (2002), 178–179.

[11] J. E. Bonnie, *Cognitive modeling for human-computer iteraction*, Graphical Interfaces'98 (Canada) (W. Davi, K. Booth, and A. Fournier, eds.), Robins Southern Printing, 1998, pp. 161–167.

[12] P. Brusilovsky, *Methods and techniques of adaptive hypermedia*, 2-3, vol. 6, 1996, pp. 87–129.

[13] K. Camarat, D. E. Yi-Luen, B. R. Johnson, and M. D. Gross, *Navigational blocks navigating information space with tangible media*, International Conference on Intelligent User Interfaces (2002), 31–38.

[14] M. H. Chignell and P. A. Hancock, *Handbook of human computer interaction*, pp. 969–996, Interscience, Amsterdam, NJ: North-Holland, 1988, Intelligent Interface Design,In Helander.

[15] M. H. Chignell, P. A. Hancock, and H. Takeshita, *Handbook of perception and cognition*, pp. 291–328, Academic Press, 1999.

[16] F. Claude-Nicolas and R. Seth, *Learning subjective functions with large margins*, Proc. 17th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA, 2000, pp. 287–294.

[17] J.R. Comstock and R.J. Arnegard, *The multi-attribute task battery for human operator workload and strategic behavior research.*, Technical Memorandum 104174, NASA, 1992.

[18] J. R. Cook and G. Salvendy, *Job enrichment and mental workload in computer-based work: Implications for adaptive job design.*, International Journal of Industrial Ergonomics (1999), 291–328.

[19] B.G.W. Craenen, A.E. Eiben, and J. I. Van-Hemert, *Comparing evolutionary algorithms on binary constraint satisfaction problems*, Transactions on Evolutionary Computation **7** (2003), 424–444.

[20] A. Finlay J. Dix, G. Abowd, and Beale R., *Human-computer interaction*, Prentice Hall Europe, 1998.

[21] Z. Duric, W. D. Gray, R. Heishman, and F. Li, *Integrating perceptual and cognitive modeling for adaptive and intelligent human-computer interaction*, Proceedings of IEEE, vol. 90, IEEE, 2002, pp. 1272–1289.

[22] A. Edwards, *Soundtrack: an auditory interface for blind users*, Human Computer Interaction **4** (1989), no. 1, 45–66.

[23] M. R. Endsley, *Situation awareness global assessment technique (SAGAT)*, Proceedings of the National Aerospace and Electronics Conference (NAECON), New York: IEEE, New York: IEEE, 1988, pp. 789–795.

[24] P. M. Fitts, *Human engineering for an effective air-navigation and traffic-control system*, p. 4, BUL BUNU, Colombus, OH, 1951.

[25] L. Francisco-Revilla and F. M. Shipman, *Adaptive medical information delivery combining user, task, and situation model*, International Conference on Intelligent User Interfaces, ACM press, 2000.

[26] T. M. Gervasio, W. Iba, and P. Langley, *Learning user evaluation functions for adaptive scheduling assistance*, Proceedings of the Sixteenth International Conference on Machine Learning, 1999.

[27] S. G. Hart and L. E. Staveland, *Development of NASA-TLX (task load index): Results of experimental and therotical research*, Human Mental Workload (P. A. Hancock and N. Meshkati, eds.), North Holland, Amsterdam, 1988, pp. 139–183.

[28] C. K. Hendy, S. E. P. Farrel, and K. P. East, *An information-processing model of operator stress and performance*, Stress, Workload, and Fatigue (A. P. Hancock and A. P. Desmond, eds.), Lawrence Erlbaum, Mahwah, NJ, 2001, pp. 34–72.

[29] E. Horvitz, *Principles of mixed-initiative user interfaces*, Proceedings of CHI 99, ACM SIGCHI (Pittsburgh, PA), 1999.

[30] JESS http://herzberg.ca.sandia.gov/jess/index.shtml.

[31] SOAR http://sitemaker.umich.edu/soar.

[32] T. Inagaka, *Adaptive automation: Sharing and trading of control*, Handbook of Cognative Task Design (Erik Hollnagel, ed.), Lawrance Erlbaum Assoc., 2003, pp. 147–169.

[33] B. D. Kaber, L.J. Prinzel III, C. M. Wright, and M. P. Clamann, *Workload-matched adaptive automotion support of air traffic controller information processing stages*, Tech. Report NASA/TP-2002-211932, NASA, September 2002.

[34] D. B. Kaber and J. M. Riley, *Adaptive automation of a dynamic control task based on workload assesment through a secondary monitoring task*, Automation Technology and Human Performance: Current Research Trends (M. W. Scerbo and M. Mouloua, eds.), Lawrance Erlaub, Mahwah, NJ, 1999.

[35] R. J. Keeble and R. D. Macredie, *Assistant agents for the world wide web intelligent interface design challenges*, Interaction with Computer **12** (2000), no. 4, 357–381.

[36] Y. Y. I. Kitamura and K. Fumio, *Real time 3d interaction with activecube*, CHI (2001), 355–356.

[37] J. Klein, Y. Moon, and R. W. Picard, *This computer responds to user frustration*, Interacting with Computers **14** (2002), no. 2, 119–140.

[38] Kleinman D. L. Lancraft R. E., *On the identification of parameters in the optimal control model*, 1979, pp. 487–501.

[39] L. Laois and M. Giannacourou, *Perceived effects of advanced atc functions on human activities: Results of a survey on controllers and experts*, International Symposium Symposium on Aviation Psychology (Columbus, OH), The Ohio State University, 1995, pp. 392–397.

[40] P. Maes, *Agents that reduce work and information overload*, Communications of the ACM **37** (1994), no. 7, 31–40.

[41] T. Masui, *Evolutionary learning of graph layout constraints from examples*, Symposium on User Interface Software and Technology, ACM, 1994, pp. 103–108.

[42] N. Math'e and J. Chen, *User driven and context basis adaptation*, User Modelling and User -Adapted Interaction **3** (1996), no. 3, 145–154.

[43] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and Teller E., *Equation of state calculations by fast computing machines*, Journal of Chemical Physics **21** (1953), no. 6, 1087–1092.

[44] A. Newell, *Unified theories of cognition*, Harvard, Cambridge, MA, 1990.

[45] A. F. Norcio and J. Stanley, *Adaptive human-computer interfaces: A literature survey and perspective*, Transactions on System, Man and Cybernatics **19** (1989), no. 2, 399–408.

[46] D. A. Norman, *The pyschology of everyday things*, p. 4, Basic Books, New York, 1988.

[47] D. A. Norman and S. W. Draper, *User -centered design: New perspectives on human computer interaction*, Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1986.

[48] M. Pazzani and D. Billsus, *Learning and revising user profiles: The identification of interesting WEB sites*, Machine Learning **27** (1997), no. 3, 313–331.

[49] R. W. Pew, *Secondary tasks and workload measurement*, Mental Workload Its Theory and Measurement (N. Moray, ed.), Plenum Press, 1979, pp. 23–28.

[50] J. L. Prinzel, A. T. Pope, G. F. Freeman, M. W. Scerbo, and P. J. Mikulka, *Emprical analysis of EEG and ERPs for psychophysical adaptive task allocation*, June 2001.

[51] J. L. Prinzel III, *Team-centered perspective for adaptive automation design*, Report TM-2003-212154, NASA, February 2003.

[52] G. B. Reid and T. E. Nygren, *The subjective wrokload assesment technique: A scaling procedure for measuring mental workload*, Human Mental Workload (P. A. Hancock and N. Meshkati, eds.), North Holland, Amsterdam, 1988, pp. 139–183.

[53] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, *Grouplens: An open architecture for collobrative filtering of netnews*, Conference on Computer Supported Cooperative Work, ACM, 1994.

[54] Suchak M. et al Resnick P., Iacovou N., *Grouplens: An open architecture for collobrative filtering for netnews*, Conference on Computer Supported Cooperative Work, ACM, 1994.

[55] L. A. Richter and G. Salvendy, *Effect of personality and task strength on performance in computerized tasks*, Ergonomics **32** (1995), no. 2, 281–291.

[56] L. Rothrock, R. Koubek, F. Fuchs, M. Haas, and G. Salvendy, *Review and reappraisal of adaptive interfaces: Toward biologically-inspired paradigms*, 2002.

[57] W. B. Rouse, N. D. Geddes, and R. E. Curry, *Architecture for intelligent interface: Outline of an approach to support operators of complex systems*, Human Computer Interaction **3** (1988), no. 2, 87–122.

[58] P. Sanderson, A. Pipingas, F. Danieli, and R. Silberstein, *Process monitoring and configural display design: A neuroimaging study*, Theoretical Issues in Ergonomical Science **4** (2003), no. 1-2, 151–174.

[59] M. V. Scerbo, *Theoretical perspectives on adaptive automation*, Human Performance in Automated Systems: Theory and Applications (R. Parasuraman and M. Mouloua, eds.), Lawrence Erlbaum, Mahwah, NJ, 1996, pp. 37–64.

[60] J. Scheirer, R. Fernandez, J. Klein, and R. W. Picard, *Frustrating the user on purpose: A step toward building an affective computer*, Interacting with Computers **14** (2002), no. 2, 93–118.

[61] S. Schiaf and A. Amandi, *User interface agent interaction: personalization issues*, International Journal of Human Computer Studies (2003), 129–149.

[62] B. Schneiderman, *The future of interaction systems and the emergence of direct manipulation*, Behavior and Information Technology **1** (1982), no. 3, 237–256.

[63] ———, *Designing the user interface: Strategies for effective human-computer interaction*, Addison-Wesley, New York, 1987.

[64] B. Schneiderman and P. Maes, *Direct manipulation and interface agents*, Interactions **4** (1997), 643–661.

[65] R. Segal and J. O. Kephart, *Incremental learning in swiftfile*, Proceedings of the Seventeenth International Conference on Machine Learning, ACM, 2000, pp. 863–870.

[66] Kephart J. O. Segal R. B., *Incremental learning in swiftfile*, 1998.

[67] T. B. Sheridan and W. L. Verbank, *Human and computer control of undersea teleoperations*, Tech. report, MIT, Cambridge, MA, 1978.

[68] H. Takagi, *Interactive evolutionary computation: Fusion of the capabilities of EC optimization of EC optimization and human evolution*, Proceedings of IEEE, vol. 89, September 2001, pp. 1275–1296.

[69] M. A. Thathachar and P. S. Sastry, *Varieties of learning automata: An overwiev*, Transactions on Systems, Man and Cybernatics **32** (2002), no. 6, 711–721.

[70] G Viano, A. Parodi, J. Alty, and C. Khalil, *Adaptive user interface for process control based on multi-agent approach*, Proceedings of the working conference on Advanced visual interfaces, ACM, 2000, pp. 201 – 204.

[71] K. J. Vicente and J. R. Rasmussen, *Echological interface design: Theoretical foundations*, IEEE Transactions on Systems, Man, and Cybernatics **22** (1992), no. 4, 589–606.

[72] R. S. Walden and W. B. Rouse, *A queueing model of pilot decisionmaking in the mutitask flight management situation*, IEEE Transactions on System, Man and Cybernetics **8** (1978), no. 12, 867–875.

[73] M. Weiser, *The computer of 21th century*, Scientific American **265** (1991), no. 3, 66–75.

[74] ———, *Some computer science issues in ubiquitous computing*, Communication of the ACM **36** (1993), no. 7, 75–84.

[75] C. D. Wickens, *Engineering psychology and human performance*, 2 ed., Harper Collins, New York, 1992.

[76] G. F. Wilson, J. D. Lambert, and Russell C. A., *Performance enhancement with real-time physiologically controlled adaptive aiding*, Report, Air Force Research Laboratory, OH, US, 2000.

[77] K. Wu, C. C. Aggarwal, and P. S. Yu, *Personalization with dynamic profiler*, Third International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems, IEEE, 2001, pp. 12–21.

Appendix A

Statistical Distribution of Parameters For Automated User

This is the statistical distribution of interface parameters after one thousand eighty generations. This part was included the thesis in order to give a better understanding of interface convergence.

Initial condition was set to be [50 0 1 0 40 0 1 0 0 0 1]. First one thousand and eighty iterations were evaluated by automated user with preference vector being [10 1 1 0 40 0 1 3 0 2 0]. Figures A.1-A.4 correspond to distribution of parameters throughout the evolution of the interface. Each row corresponds to the time window that parameter distribution belongs to. First row corresponds to the distribution of parameter after eighty iterations. Second row corresponds to parameter distribution of whole interface population and third row corresponds to final eighty iteration. Every column corresponds to one parameter as labelled.

At the instant of one thousand and eighty, preference vector for the automated user is converged to [50 3 0 1 40 0 1 1 2 0 1]. Figures A.5-A.8 show statistical distribution of interface parameters for second case. Interpretation of rows and columns are the same as before.

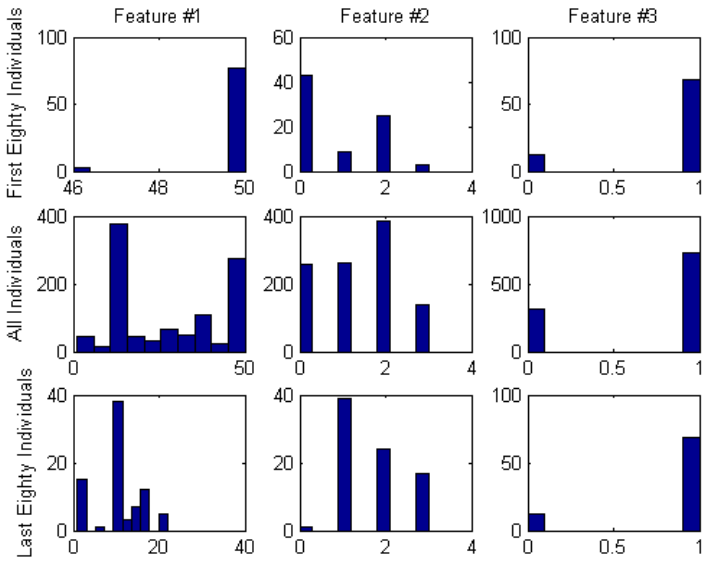


Figure A.1: Statistical distribution of interface parameters features 1-3 for preference vector [10 1 1 0 40 0 1 3 0 2 0]

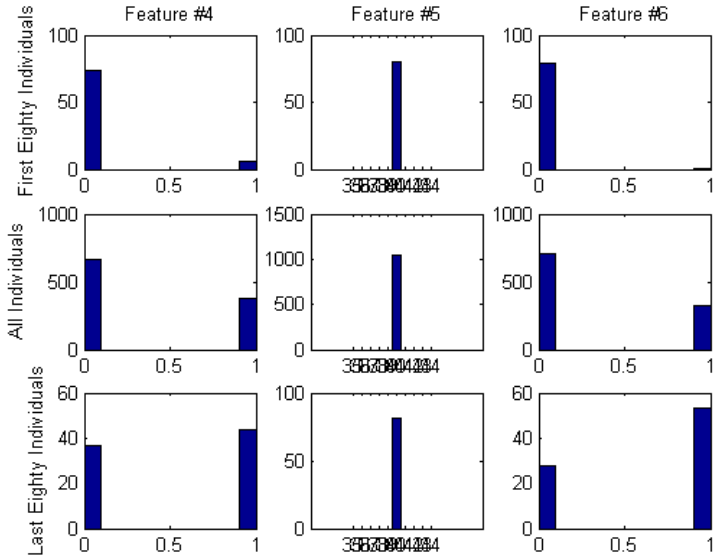


Figure A.2: Statistical distribution of interface parameters features 4-6 for preference vector [10 1 1 0 40 0 1 3 0 2 0]

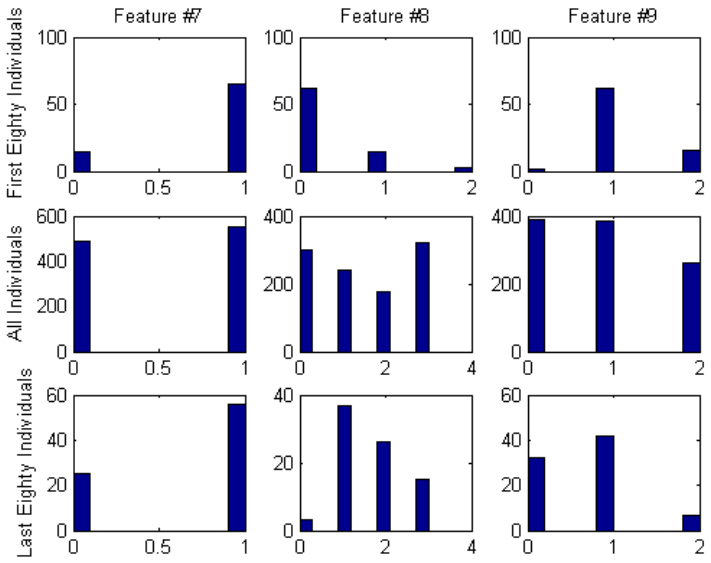


Figure A.3: Statistical distribution of interface parameters features 7-9 for preference vector [10 1 1 0 40 0 1 3 0 2 0]

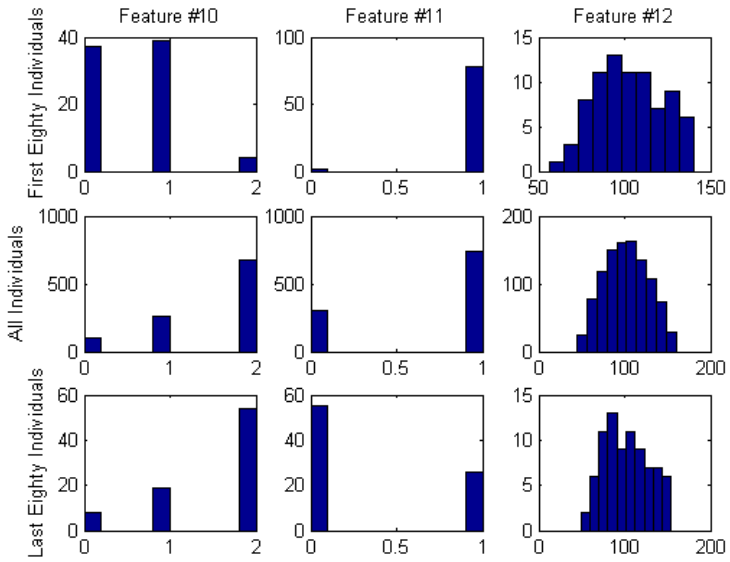


Figure A.4: Statistical distribution of interface parameters features 10-12 for preference vector [10 1 1 0 40 0 1 3 0 2 0]

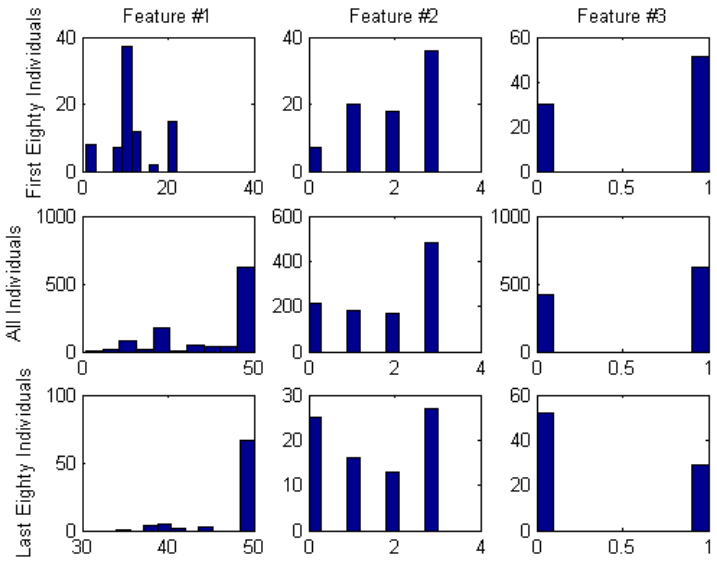


Figure A.5: Statistical distribution of interface parameters features 1-3 for preference vector [50 3 0 1 40 0 1 1 2 0 1]

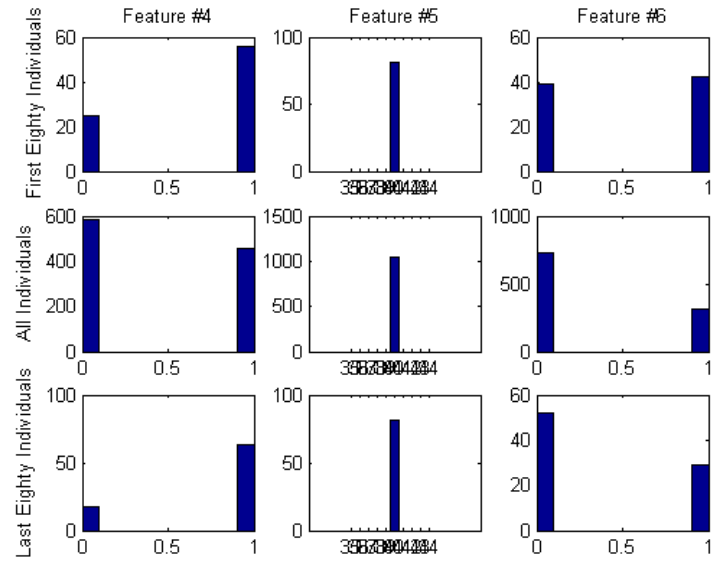


Figure A.6: Statistical distribution of interface parameters features 4-6 for preference vector [50 3 0 1 40 0 1 1 2 0 1]

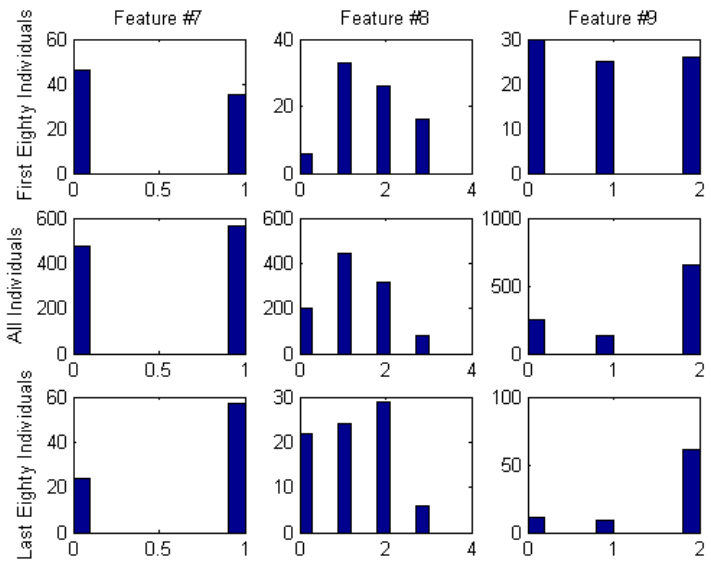


Figure A.7: Statistical distribution of interface parameters features 7-9 for preference vector [50 3 0 1 40 0 1 1 2 0 1]

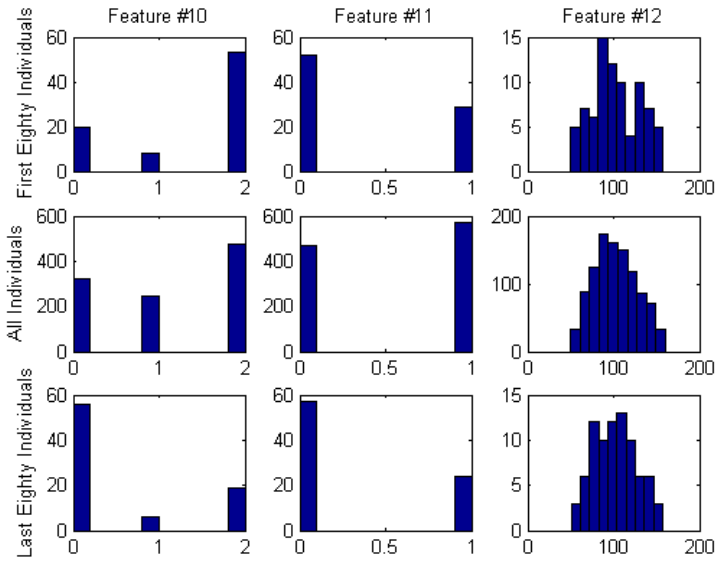


Figure A.8: Statistical distribution of interface parameters features 10-12 for preference vector [50 3 0 1 40 0 1 1 2 0 1]

Appendix B

Statistical Distribution of Parameters for Human User

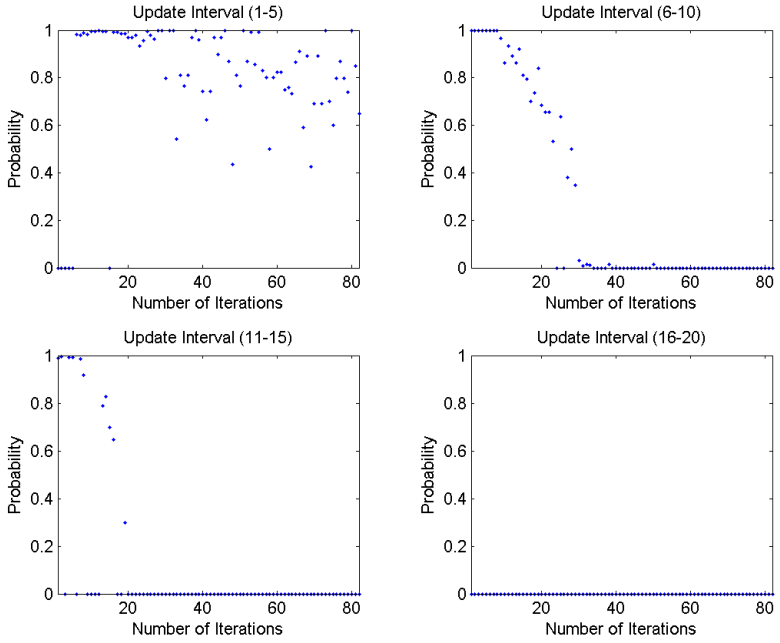


Figure B.1: Probabilities assigned by user model to each of the parameters

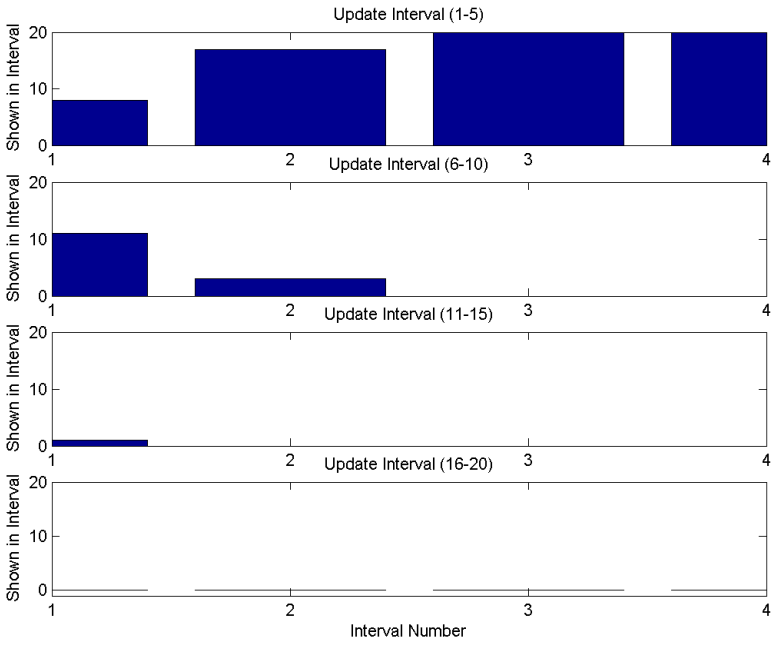


Figure B.2: Statistical distribution of parameters at every 20 iterations

Appendix C

Questioner Used for Subjective Evaluation

Subjective Evaluation of the Interface Adaptation

Name: _____ Date: _____

1- Please indicate your preference to air plane representation	[1-4 (best)]	
- Small white Planes		____
- Small red Planes		____
- Large white Planes		____
- Large red Planes		____
2- Please indicate your preference to update interval	[1-4 (best)]	
- Update interval between 1 to 5 sampling time		____
- Update interval between 6 to 10 sampling time		____
- Update interval between 11 to 15 sampling time		____
- Update interval between 16 to 20 sampling time		____
3- Did adaptation help you to increase your efficiency ?	[1-5 (best)]	____
4- Did adaptation help you to decrease your workload ?	[1-5 (best)]	____
5- Please rate the consistency of the adaptation	[1-5 (best)]	____
6- Did you feel comfortable using the interface ?	[1-5 (best)]	____
7- Which was more important? (Select One)	Update Interval	Plane Shape

Appendix D

User Model Adaptation- Simple Example

In this section we will provide a very simple two dimensional case for parameter space that is $\mathcal{A} \in \mathbb{R}^2$ for a better understanding of user model U adaptation. Let's define parameters as $A = [a_1 \ a_2]$ *where* $a_1 = 1, 2, 3, 4$ *and* $a_2 = 1, 2, 3, 4$. a_1 can be interpreted as ,let's say, contrast level of the display. a_2 can be interpreted as, let's say, predefined spatial placement of interface components. Now this is a small search space with $dim(\mathcal{A}) = 2$ and cardinality $|\mathcal{A}| = 16$. Our user model U given in Equation D.0.1 is Gaussian with $G(\mu, C)$. Assume we have initial conditions set to $\mu = [0 \ 0]$, and $C = \begin{bmatrix} 500 & 0 \\ 0 & 500 \end{bmatrix}$ for initially big variance.

$$U = \frac{1}{\sqrt{2\pi|C|}} \exp\left(-\frac{1}{2}(A-\mu)^T C^{-1}(A-\mu)\right) \tag{D.0.1}$$

Figure D.1 shows initial topology of this user model. As can be seen from the figure user model has no preferences over the interface models. Decision is not constrained by user model, thus GA is free to explore the search space.

We now will take one more step of our algorithm. For example inference mechanism (GA + user model) decide to show interface model $A = [1 \ 2]$ to the user. Final assumption is preference of our user. Let's say user is a moderate user and he/she prefers $A = [2 \ 2]$ interface model to other interfaces. Throughout the evaluation of interface $A = [1 \ 2]$ we detect a better performance than the average performance according to Equation 3.4.1. Adaptation will update the user model parameters as given in Equations 3.4.2 and 3.4.3. In this example we will take an aggressive step in order to explain the algorithm. Adaptation parameters are set to $\gamma = .5, \tau = .1$. Resultant model parameters will be $\mu = [0.5 \ 1]$ and

$$C = \begin{bmatrix} 50 & 0 \\ 0 & 50 \end{bmatrix}$$

Corresponding topology of user model at the end of the first step is given in Figure D.2. Now user model has more constraint on the interface parameter. Next step will continue as follows. GA will find new candidate solutions to be shown in step two.

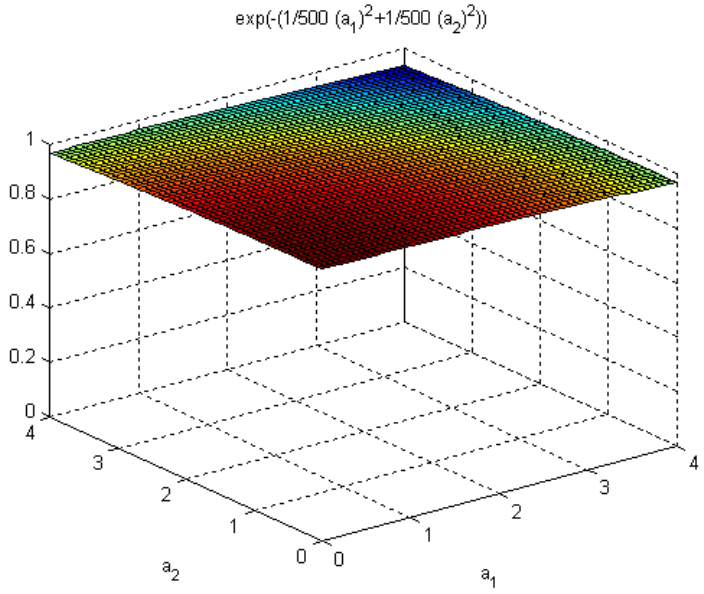


Figure D.1: Initial user model

However this time user model will evaluate them and give less score to the ones that are far from the center of the user model, since we did not consider cross-correlation as explained before. After this evaluation, one of the interface will be selected to be shown in the next iteration, proportional to its score. As we continue iterating our algorithm we may end up with a user model topology shown in Figure D.3. This model limits GA exploration and leads the algorithm to exploit the local solutions around the model center.

This concludes definition of our example. As it was mentioned before, our user model acts like a localization function to limit GA exploration in the long run. Interplay of exploration and exploitation is balanced with usage of two different adaptation scheme. Adaptation parameters are intuitive however hard to set. Good heuristic is required to set these parameters.

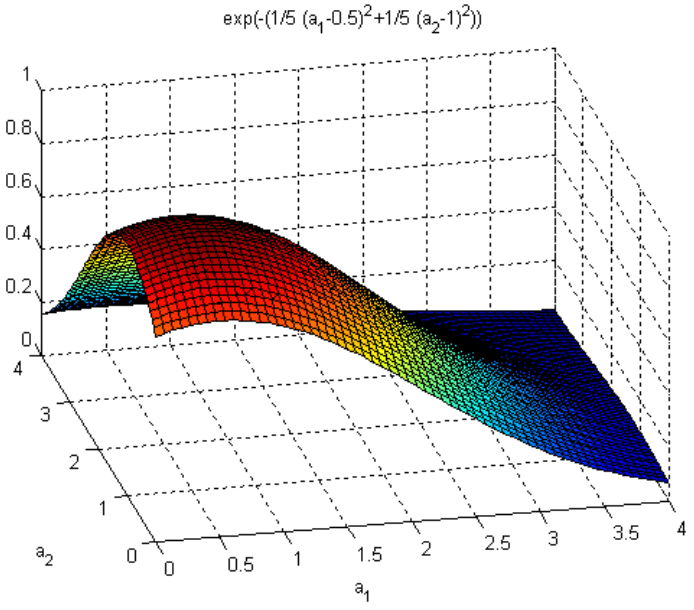


Figure D.2: User model during evolution of the interface

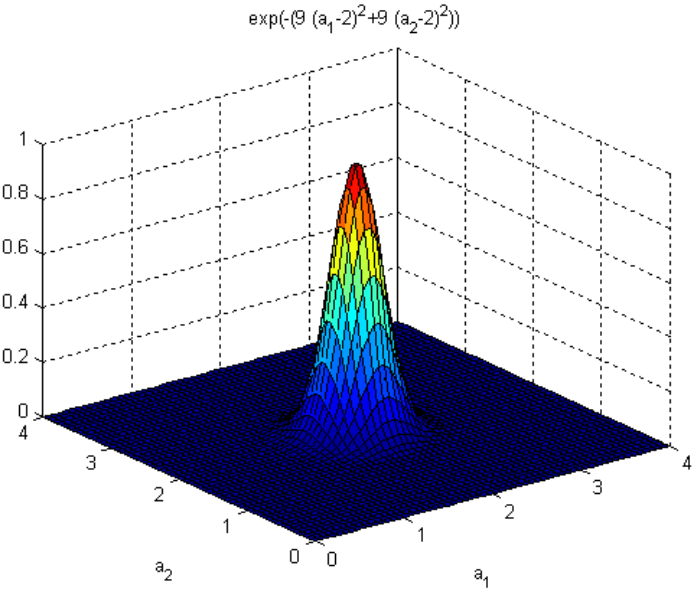


Figure D.3: Final user model

VITA

Lemi Daghan ACAY

Candidate for the Degree of

Master of Science

Thesis: ADAPTIVE USER INTERFACES IN COMPLEX SUPERVISORY TASKS

Major Field: Electrical and Computer Engineering

Biographical:

Personal data:

Born in Konya, Turkey, on May 31, 1978 the son of Ali and Neslihan ACAY

Education:

Graduate from Yesilkoy 50. Yil High School, Istanbul, Turkey in May 1996. Received Bachelor of Science degree in Electrical and Electronics from Bogazici University, Istanbul, Turkey in May 1996 Completed the requirements for the Master of Science with major Electrical and Computer Engineering in July, 2004

Experience:

Worked as research assistant under a robot project at Carnegie Mellon University in 2002 and at Boğazici University from 1999 to 2002.