AN ABSTRACT OF THE THESIS OF

William R. Secor for the degree of Master of Science in Industrial Engineering
presented on June 30, 2017
Title: The Effect of Acquisition and Analysis Automation Failure in a Low-Fidelity Multitasking System

Abstract approved:

Kenneth H. Funk II

Research into the effect of automation on human multitasking performance is primarily focused on Decision Automation (a machine suggesting a certain course of action for a human operator to take) and Action Automation (a machine carrying out a certain course of action on behalf of the human operator) while Acquisition Automation (a machine provides assistance which helps a human operator in sensing information) and Analysis Automation (a machine provides information predicting the future state of a system) have been acknowledged as types of automation but largely ignored. The inclusion of the former in systems which require human multitasking is beneficial, while the effects of the latter are unclear. In order to determine said effects a study was conducted using a low fidelity simulation of a multitasking system with No Automation, Acquisition Automation, and Analysis Automation states, as well as failure states for those systems with automation included. Results show no statistically significant difference in performance between systems with the automation enabled versus those with the automation in a failed state, nor was there any statistically significant difference between the performance of groups with no automation present and those with automation present (in both active and failed states).

The Effect of Acquisition and Analysis Automation Failure in a Low-Fidelity Multitasking System

by
William R. Secor

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented June 30, 2017
Commencement June 2018

Master of Science thesis of William R. Secor presented on June 30, 2017.

APPROVED:

Major Professor, representing Industrial Engineering

Head of the School of Mechanical, Industrial, and Manufacturing Engineering

---

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

William R. Secor, Author

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

List of Tables

Dedicated to my wife, Elizabeth, and my daughters Yvaine and Imogen – both born while completing this work. I love you all.

Chapter 1

Introduction

1.1 Introduction

Motivated by the fact that a large number of aircraft accidents are attributable to human error, Funk (1991) proposed a normative theory of cockpit task management, which acknowledged that pilots and crew are often involved in managing multiple concurrent tasks, and used it as the basis for an experimental pilot-vehicle interface (PVI). The PVI listed and modeled tasks, as well as recommended when they should start, their prioritization, when to interrupt, when to resume, and when to terminate tasks. That is, it took a wholistic view of the state of the system into account before automatically providing feedback to the pilot, rather than a series of independent system component states. Testing (on a flight simulator) of the PVI showed an improvement in task execution and management performance as well as a reduction in pilot workload.

Further investigation by Chou, Madhaven, and Funk (1996) into National Transportation Safety Board aircraft accident reports and Aviation Safety Reporting System aircraft incident reports revealed that, based on a cockpit task management error taxonomy developed by the authors, errors in cockpit task management were present in 23% of accidents and 49% of inccidents.

As a continuation of the aforementioned research, Colvin, Funk, and Braune (2005) made an attempt to identify the factors that affect task prioritization in the cockpit task management process. To that purpose, the authors developed a hypothetical framework and model of task prioritization. This two-part task simulator study (conducted using professional pilots) identified six factors that have great influence on the pilot's task prioritization in the cockpit. These factors are:

1. Procedural Consistency of the task
2. Importance of the task
3. Salience of task-related stimuli
4. Time/Effort required to perform the task
5. Urgency of the task

Colvin et al. (2005) recommended further research be conducted on the identification of the factors that influence task prioritization.

Certainly pilots are not the only persons in situations requiring the management of mutiple concurrent tasks. There are examples from other fields,

including but not limited to process control (Greenstein & Rouse, 1982), medicine (Clyne, 2012), and driving (Burge & Chaparro, 2012).

It is not a simple task to study experienced pilots under experimental conditions. Pilots are not easy to come by as participants. Flight simulator experiments take a long time to run. Additionally, the aforementioned studies involved or suggested development of systems with some degree of automation in place. However, they were not framed in a way that findings could be applied outside their specialized domain. All of these factors point to the need for simpler systems (which do not require years of specialized training) to use as an experimental platform at the intersection of automation and concurrent task management performance.

There is plenty of automation research available, but most of it is focused on higher levels of automation (Onnasch, Wickens, Li, & Manzey, 2014), such as Decision Automation (computer providing recommended courses of action), Action Automation (machines acting on information under the supervision of a human, such as an auto-pilot system) and Adaptive Automation (situational implementation of the previously mentioned types of Automation) (Parasuraman, Sheridan, & Wickens, 2000).

This research shows that higher degrees of automation in a system can lead to human out-of-the-loop performance problems. That is, as computers are increasingly relied upon to control systems, the level of interaction of human operators with these systems decreases. This can be a problem if an operator is suddenly called upon to take control of a system during a non-normal state. It is this very lack of interaction which can lead to decreased operator

situation awareness, skill decay due to inactivity, and an unfortunate sense of trust (Parasuraman & Riley, 1997; Bahner, Hüper, & Manzey, 2008) – all of which play a role in poor operator performance during automation failure.

Increasing the degree of automation of a system generally improves human performance in that system (as demonstrated in Funk (1991)), and failure of higher degrees of automation often results in poor performance, below that of a comparable system without the higher degree automation in place (Endsley, 1999; Onnasch et al., 2014). One might infer that the introduction of lower degrees of automation into a system, and the effects of their failure, would result in less of a performance gain and less of a performance drop following failure. However, there are no studies in the literature (to knowledge of this author) that either seek to answer this question or could be used to answer this question.

1.2 Research Objectives

The primary goal of this study was to determine the effect of the introduction and failure of acquisition and analysis automation in a low-fidelity system involving concurrent task management. More specifically, first, does introducing acquisition or analysis automation enhance operator performance versus a system where these types of automation are absent? Second, what effect does automation failure have on the performance of a participant who has learned to operate a system with acquisition or analysis automation always present and functional? Third, to what extent, if any, is post-automation-

failure performance affected?

1.3 Organization of the Thesis

Chapter 2 begins with a review of low fidelity simulators used for researching concurrent task management. Then, motivated by the fact that none of these simulators take any form of automation into account, reviews the literature defining degrees and levels of automation. Research on lower degrees of automation is lacking, so this section is followed by a review of multiple resource theory and situation awareness which leads to the concluding section of recommendations on designing low-fidelity simulators of concurrent task management for studying the effect of different types and levels or automation on human performance.

Chapter 3 introduces the research hypotheses for this study. The materials section describes the ETME, a low fidelity simulator built based on the recommendations from Chapter 2. The participants and experimental procedures used in support of answering the research hypotheses are also covered.

Chapter 4 describes the particpant demographics. This is followed by a presentation of experimental results.

Chapter 5 discusses the significance of the results presented in Chapter 4 relative to the hypotheses presented in Chapter 3.

Chapter 6 concludes with a discussion on study limitations and reccommendations for future research.

Chapter 2

Literature Review

2.1 Low Fidelity Simulation

Using a real-world system and situation as an experimental platform is challenging. First, it can be prohibitively expensive (Kozak, Hancock, Arthur, & Chrysler, 1993). This includes both actual material cost (imagine the jet-fuel cost of training commercial airline pilots in the actual craft they will be flying) and depending on the study the expense of finding highly qualified participants (pilots do not grow on trees). Second, it can be dangerous. A study to investigate the effect of cell phone conversation on drivers should not be carried out with vehicles in the "real world" — this would endanger the experimenters, participants, and general population. The solution to both of these issues is to make use of some manner of virtual environment.

Flight simulators are a well known and widely used virtual environment (Wickens & Hollands, 2000). In fact, pilots are allowed to obtain a single en-

gine license via 100 hours of time (out of 250 total hours) on a flight simulator during the course of training (Aeronautics and Space, 2017) . These same flight simulators are also used in experimental studies; often times to examine how pilots react to some change in the environment (Nikolic & Sarter, 2007), or determine how individual differences correlate with simulator performance (Molesworth & Chang, 2009).

While flight simulators may come to mind most easily when speaking about virtual environments, there are many other types simulators used today. For example, there are driving simulators (Parkes & Coleman, 1990), emergency dispatch simulators (Joslyn & Hunt, 1998), Naval Combat Information Center (CIC) simulators (DiVita, Obermayer, Nugent, & Linville, 2004), and a wide variety of other simulators which could be placed on a scale ranging between extremely realistic (high-fidelity) and abstract (low fidelity). A commercial flight simulator is a high-fidelity simulator, and a driving-based video game would fall somewhere between the two ends of the spectrum.

Unfortunately examples of low fidelity simulators are few in number, as research on the management of multiple concurrent tasks is “less prominent” (Chou et al., 1996) and those active in this line of research tend to use higher fidelity domain specific simulators. A review of the literature found a small number of low fidelity simulators being used for investigation of concurrent task management.

Tulga and Sheridan (1980) developed a low fidelity computer based simulator where experimental subjects were shown a screen with multiple rectangles (representing tasks) and were instructed to “perform” the tasks by clicking

on the screen via a digitizer (tablet). When acted on, the tasks would shrink in size, and the subject's score would increase. All the rectangles would move at a steady rate from the left to the right hand side of the screen, and upon hitting the right hand side of the screen would no longer be available to act on. New rectangles would arrive from the left of the screen at some variable interarrival time. Tulga and Sheridan (1980) felt that this simulator closely resembled supervisory control systems, as present (at the time) in aircraft, nuclear power, and industrial control. The idea was that this highly abstract task would not require specialized domain knowledge, and therefore participants would not need extensive training (as they might with a simulator). Even so, no effort was made to validate conclusions reached by experimentation in their simulator with higher fidelity domains.

Endsley (1999) created a variant of the system used by Tulga and Sheridan (1980), but modified it to reflect tasks in dynamic control jobs. "Multitask" (the name of their system) displays rectangles which move from the edges of the screen to a circular boundary in the center of the screen. The goal of users of Multitask is to click on the rectangles, thereby reducing their size to zero before the rectangles reach the circular boundary at the center of the screen or before they colide with one another. Larger rectangles take longer to process. Rectangles have variable reward, as displayed on screen with data tags. Additionally, the system is configured to provide ten different levels of automation (discussed in Section 2.2) ranging from full manual control to full automation as well as the ability for any of the automation levels to "fail" (be turned off). Multitask was designed to provide insight on which levels of automation

might play a significant role in higher-fidelity domains.

Joslyn and Hunt (1998), in order to investigate whether it is possible to provide a general measure of time-pressured decision making, developed a low fidelity simulator called the Abstract Decision Making task (ADM). In the ADM participants earn points by sorting objects into bins as rapidly as possible. Sorting is based on object attributes such as size, shape, and color. Objects are sorted into bins which match certain object attributes. Joslyn and Hunt (1998) went on to validate results from ADM (those with higher ADM scores peform better in higher fidelity domains) by using higher fidelity simulators in domains which are heavy on concurrent task management: 911 dispatch and air traffic control.

Nicolalde, Funk, and Uttl (2003) developed the Task Management Environment (TME) in order to simulate an abstract control system composed of up to fourteen subsystems. Each subsystem has a status which is a single state variable designed to vary from 0% to 100%. The effect of different factors on the task management performance of participants was studied using the TME. Tests were performed to evaluate their cognitive abilities (measuring verbal intelligence, working memory, reaction time, and decision time) and were correlated with their concurrent task management performance as measured using the TME. It was found that the correlation was low, from which the authors concluded that few cognitive processes alone are predictors of performance in concurrent task management (CTM); rather it is likely to be a complex combination of all these cognitive processes.

Further study using the TME was carried out by Chen and Funk (2003) and

involved using fuzzy logic models built around prioritizing task status, importance, and urgency to simulate human performance in the system. These models more accurately predicted the performance of human users of the TME, as compared to a random model. This suggests that humans take these factors into account when developing a strategy for task management. Those who quickly identified the least important tasks to "shed" (not attend to) performed better than those who attempted to attend to all tasks. This fact was mirrored by the development of two additional models that allowed the computer to only attend to the four or five most important tasks.

Shakeri and Funk (2007) also used a low fidelity simulator, called Tardast, much like Tulga and Sheridan (1980), to study concurrent task management. In this study participants attended to multiple concurrent tasks, each of which had a certain satisfaction level (0 to 100%), importance, decay rate, and correction rate when acted on by the user. The goal was to keep the system at a high level of satisfaction. Each task degraded at a constant rate, and was corrected at a constant rate when clicked on by the participant. A complex algorithm was used to determine an overall score, the majority of which was reflected in the participant's ability to keep multiple "tasks" at a high satisfaction level. Shakeri and Funk (2007) used results from Tardast to develop a computer based heuristic model which closely matched human performance (in Tardast). Much like the work of Tulga and Sheridan (1980), these scores were not validated against other, higher fidelity, concurrent task management situations.

Table 2.1 summarizes concurrent task management research which has

Author(s)	Simulator Name	Automation Present	Automation Measured	Validated
Tulga and Sheridan (1980)	None	No	No	No
Joslyn and Hunt (1998)	ADM	No	No	Yes
Endsley (1999)	Multitask	Yes	Yes	No
Nicolalde et al. (2003)	TME	No	No	No
Chen and Funk (2003)	TME	No	No	No
Shakeri and Funk (2007)	Tardast	No	No	No

Table 2.1: Summary of low fidelity simulator studies

made use of low fidelity simulation. Even though studies (such as the meta-analyasis by Onnasch et al. (2014), discussed later) acknowledge that automation is, or is becoming, pervasive in domains dealing with concurrent task management; studies which use low fidelity simulation fail to include any level of automation in their simulators (with one exception, Endsley (1999)).

2.2 Automation

Parasuraman and Riley (1997) define automation as "A device or system that accomplishes (partially or fully) a function that was previously, or conceivably could be, carried out (partially or fully) by a human operator." Funk (1991), in his definition of a task ("a process performed (at least partly by a human) to achieve a goal") implicitly acknowledges that there is a role for automation to play in task management, and then explicity acknowledges it by implementing a prototype task management system based heavily on a type of automation. The significance of automation's role in a system depends on the type of automation present. Parasuraman et al. (2000) identify five types of automation (below), later referred to by Onnasch et al. (2014) as Degrees of Automation (DOA).

1. Acquisition Automation. This is is automation that aids users in registering (sensing) information.

2. Analysis Automation. This provides information predicting the future state of the system.

3. Decision Automation. Here the machine suggests certain courses of action.

4. Action Automation. Here the machine carries out a certain course of action.

5. Adaptive Automation. Any combination of the above designed to adapt to system conditions.

There is clearly a break between the first three types of automation and the last two types of automation. This is where automation moves from providing some form of information to a human operator to acting on behalf of the operator, leaving the humans in a position of "supervisory control," where they observe the actions of the automation system and intervene if appropriate (Sheridan & Verplank, 1978). There are two taxonomies in the literature detailing levels of automation (LOA) moving from Decision to Action automation (Sheridan & Verplank, 1978; Endsley, 1999). Sheridan and Verplank (1978) proposed the first, a 10-level taxonomy:

1. Human does the whole job up to the point of turning it over to the computer to implement.

2. Computer helps by determining options.

3. Computer helps to determine and suggests one option, which human need not follow.
4. Computer selects action and human may or may not do it.
5. Computer selects action and implements it if human approves.
6. Computer selects action, informs human in plenty of time to stop it.
7. Computer does whole job and necessarily tells human what it did.
8. Computer does whole job and tells human what it did only if human explicitly asks.
9. Computer does whole job and decides what the human should be told.
10. Computer does the whole job if it decides it should be done, and if so, tells the human, if it decides that the human should be told.

Endsley (1999) proposed a revised 10-level taxonomy. It is quite similar to that of Sheridan and Verplank (1978), though it frames each level in the context of who (human or computer) is responsible for monitoring the system, generating courses of action, selecting courses of action, and implementing (carrying out) courses of action.

1. Manual control. Human performs all tasks.
2. Action support. Computer assists with performance of selected action.
3. Batch processing. Human makes decisions and turns them over to computer to carry out.

4. Shared control. Human and computer generate options, human has control over which to carry out. Control of carrying out actions is shared between the human and computer.

5. Decision support. Human and computer generate options, human has control over which to carry out. The computer is directed and fully controls carrying them out.

6. Blended decision making. The computer generates a list of options and carries them out if the human consents.

7. Rigid system. Computer presents a limited list of options to the human, the human selects from this limited list and may not generate other options.

8. Automated decision making. Humans and computer generate options, and the computer selects the best option and carries it out.

9. Supervisory control. The computer generates options, selects the best, and carries it out. The human is there to monitor the system.

10. Full automation. Computer carries out all actions and the human is unable to intervene.

The first level in both taxonomies give a small nod to Acquisition Automation and Analysis Automation (as discussed in Section 2.2), but the remaining nine levels in both lists are easily classified as Decision Automation and Action Automation.

It is no surprise then that research into automation is primarily focused on Decision Automation and Action Automation (Onnasch et al., 2014). This makes sense, as having a computer act on a human's behalf has numerous implications, related to system design, human performance, and even ethics. However, this leaves Acquisition Automation and Analysis Automation in a state where they have been acknowledged as types of automation, but largely ignored in automation research.

In a meta-analysis of studies which incorporate different degrees and levels of automation and measure their effect on human performance, Onnasch et al. (2014) found that higher degrees of automation had a beneficial impact on performance, and failure of automation of higher degree had a negative impact on performance. However, in their review of the literature Onnasch et al. (2014) were only able to locate 18 studies which varied degree or levels of automation, none of which focused on the first degree of automation (Acquisition Automation) and only two of which dealt with the second degree of automation (Analysis Automation, unsurprising these were both from air traffic control research).

Therefore, if we hope to learn more about the effect of including these types of automation in systems involving concurrent task management then a low fidelity simulator amenable to acomodating these types of automation should be built, and followed up with experiments which can correlate the presence/absence of Acquisition Automation and Analysis Automation with effects on human performance.

Since Acquisition Automation has to do with registration of information, building a simulator to test it should be informed by multiple resource theory as it addresses allocation of individual resources between perceiving, processing, and responding to incoming information. Analysis Automation is related to understanding the current and future states of the system, so a simulator to test these should be based on situation awareness literature.

2.3 Multiple Resource Theory

Wickens (1980, 1984, 1991) proposed that there are four dimensions along which human cognitive resources can be categorized: codes, modalities, stages, and responses. These dimensions compose what is commonly known as "Multiple Resource Theory."

Information contained in working memory, whether that information is from perception, retrieved from long-term memory, or being held for a future response, takes the form of either a spatial or verbal code. For example, the information held in memory while working out a math problem would be verbal in nature, while the information in memory from visual perception while driving a car would be spatial in nature. Studies involving the performance of multiple tasks suggest that these two codes depend on separate resources (Polson & Friedman, 1988). Concurrent actions which rely on the use of separate codes are more efficient than those that share a specific type of code (Wickens & Liu, 1988). In fact, different actions sharing the same modality may interfere with each other to some extent. This explains why walking and

talking or driving and holding a conversation with a passenger is not difficult, while holding two conversations at once is quite difficult.

Modalities refer to how information from the outside environment is perceived (Wickens & Hollands, 2000). Information can come in via sight (Visual Perceptual Modality) or sound (Auditory Perceptual Modality). The other senses (touch, taste, vestibular) could be considered, but these are not commonly used in the design of human systems. Dividing attention between multiple modalities is more easily accomplished than dividing attention across a single modality (Parkes & Coleman, 1990). This is why driving and reading a map at the same time may prove difficult.

It is clear from experimental work that resources associated with perception and response are separate entities (Pashler, 1989). Wickens and Hollands (2000) also claim that working memory and cognition are a separate stage. In other words, there are stages (memory) associated with perception of the environment, with processing information, and with responding or acting on information. Much like the previously mentioned resource categories, actions which engage separate stages allow for more efficient time sharing.

The method of responding can be broken down into vocal actions (speaking) or manual action (moving). Splitting actions between these response types is more manageable than performing multiple actions via a single type of response (Wickens & Hollands, 2000).

2.4 Situation Awareness

Situation awareness (SA) is a perceptual understanding of one's environment and comprehension of the past, present, and future status of elements within the environment (Endsley, 1995). Therefore, in order to manage tasks, one must maintain situation awareness (SA). SA is critical for human control of systems, especially as these systems become increasingly complex and push the capabilities of human operators.

1. Level 1 SA is perception of the elements in the environment (Endsley, 1995). For example, a driver would need to perceive the cars surrounding him/her, their dynamics and behavior, other potential obstacles (pedestrians, animals, etc), the weather, and road conditions.

2. Level 2 SA is comprehension of the current situation (Endsley, 1995). For example, a driver, upon seeing a police car with its lights on in his/her rear view mirror, would understand that changing lanes to allow the police car past is the appropriate course of action. This understanding is based on pattern recognition within the environment, which could come from previous experiences or training.

3. Level 3 SA is projection of future status (Endsley, 1995). For example, upon seeing the police car and understanding that a lane change is necessary, a driver may observe cars in the lanes available to switch in to, and based on their actions and projection of future actions, decide that it is appropriate to speed up to get clear of vehicles that may impede a lane change.

Since humans have limited attentional capacity, the details of the environment that attention is focused on influences Level 1 SA (by gaining information on some objects at the exclusion of others), which in turn influences Level 2 and Level 3 SA. Failing to perceive certain elements in the evironment (level 1) may prevent comprehension of the current situation (level 2) which prevents projection of future status (level 3).

Increased stress may limit the amount of working memory available to process information from the outside environment, thus hindering Level 1 or Level 2 SA. Increased complexity could inhibit Level 2 and Level 3 SA, as the more complex the environment gets, the more likely a person is unable to sufficiently comprehend or simulate the situation they are in. This is in line with with the conclusions from Shakeri and Funk (2007), that task prioritization degrades as the number of concurrent tasks increases.

A level 1 SA error would be the failure to perceive something relevant for SA in the environment. A level 2 SA error would be failure to properly itegrate or comprehend environmental information in pursuit of a goal. A level 3 SA error would be failure to run a good mental simulation of the enviornment.

2.5 Recommendations for Simulator Design

The identification of factors that influence prioritization of tasks by pilots by Colvin et al. (2005) should have a clear analogue present in a low-fidelity simulator. These factors were identified as applicable to a specific domain (aviation), but anyone engaged in the management of multiple concurrent tasks

is likely to be operating with the same factors in mind (though perhaps not in the same order of priority). More generally:

1. Procedural Consistency of the task → How does the task fall into the standard operating procedures of the current system?

2. Importance of the task → How critical is the task in relation to the goals of the system operator?

3. Salience of task-related stimuli → How attention-grabbing are the signals/displays/controls associated with the task?

4. Time/Effort required to perform the task → How difficult and time-consuming is the task expected to be?

5. Urgency of the task → What is the relevance of deadlines to completion or satisfactory performance of the task?

Any low-fidelity simulator designed for examining concurrent task management should be just complex enough to allow for all of these factors to come into play at some level, otherwise one might call into question the validity of the simulation environment being used in regards to whether or not it reflects, at an abstract level, what is experienced in more complex, domain specific, simulations and environments. Since the goal of using low-fidelity simulators is to facilitate the use of non-expert (in a particular domain) participants, with a minimum of training necessary in the simulator, it follows that system complexity should not increase beyond a level which allows for the existence of the aforementioned factors.

Wickens' Multiple Resource Theory divides human resources into four separate categories: codes, modalities, stages, and responses. Since these elements describe the effectiveness of sharing time between multiple concurrent tasks, a low-fidelity simulator for use in investigating concurrent task management at an abstract level should incorporate tasks that utilize these elements. This implies that multiple, different types of tasks will need to be used, rather than multiple instances of the same task. Furthermore, since system designers should be building environments that promote effective time sharing between multiple tasks, the aim of a low-fidelity simulator should not be to overload a particular resource, but to allow for the use of multiple resources so as to best relate to "real-world" scenarios. Furthermore, system variables should be calibrated so as to avoid ceiling and floor effects. That is, a simulator should not be too difficult (a floor effect leading to uniformly low scores from all participants), nor too easy (a ceiling effect leading to uniformly high scores from all participants).

A low fidelity simulator should allow for a participant using it to achieve Level 1, Level 2, and Level 3 situation awareness. This necessitates building a simulator that goes beyond simple, temporally discrete tasks (like reaction time tasks). Rather, moving to Level 3 SA means that tasks should be of a nature which allows for prediction of future states in the context of current actions (mental simulation). Conversely, the simulator should also penalize users for poor performance due to a failure to achieve any of these levels of situation awareness, or perhaps the ability to achieve situation awareness at these levels in a less than optimal way. This could be accomplished in various

ways, some examples of design decisions which may facilitate this include building some manner of interruption into the system, degrading visual or auditory information, or visually separating elements of the system necessary to perform certain tasks.

The previously discussed research leads to a set of recommendations for low-fidelity simulators being used to investigate human multitasking performance:

1. The simulator should be abstract and simple enough to allow for a member of the general population to operate it with minimal training.

2. The simulator should provide for automation of various types and levels.

3. The simulator should allow for the ability to achieve or fail to achieve Level 1, Level 2, and Level 3 situation awareness.

4. Tasks in the simulator should vary in regards to procedural consistency, importance, salience, time/effort requirements, and urgency. This allows for prioritization with regard to these factors.

5. Tasks in the simulator should vary across codes, modalities, stages and responses in a real world, system realistic, fashion.

All four previously discussed low-fidelity simulators (Tulga, TME, Tardest, ADM) meet recommendation #1. Recommendation #2 is partially met by the discussed simulators. One may argue that achieving Level 2 and Level 3 situation awareness within these simulators is questionable due to the abstract

nature of the systems and goals presented for participants. Recommendation #3 is only partially met by TME and Tardest, although neither of these simulators exhibit any sort of temporal variance in such factors. The final recommendation is also only partially met, as none of these simulators make use of sound.

While none of the discussed simulators effectively meet the recommendations made in this paper for low-fidelity simulator design, it is possible that a combination of these simulators (and some additional modifications), merging the simple tasks used in the individual simulators into a slightly more complex simulator, could come closer to this goal. The other option is to build from scratch, but this would take much more time than making use of previously well designed and understood systems (simulators).

2.6 Hypotheses

Given that automation research in concurrent task managment has, to this point, been focused on decision/action/adaptive automation, mostly in domain specific settings, it makes sense to broaden this line of research by investigating the effect of acquisition automation and analysis automation on performance by starting with a simple, domain agnostic, low-fidelity simulation involving concurrent task management. Simple, in this case means the simulator should engage minimal resources (a single modality, as per Wickens multiple resource theory) and allow for the development of only basic situational awareness (stage one SA as per Endsley).

Comparing performance with these two types of automation present against baseline performance in the same system with these two types of automation absent allows one to answer the question: Does introducing acquisition or analysis automation enhance operator performance versus a system where these types of automation are present? It is hypothesized that a system with acquisition or automation analysis present will result in improved performance versus a system where either type of automation is absent.

Comparing performance with these two types of automation present against performance where participants suddenly have these two type of automation removed (simulating automation failure) allows one to answer the question: What effect does automation failure have on the performance of a participant who has learned to operate a system with acquisition or automation analysis always present and functional? It was hypothesized that the failure of either type of automation would result in decreased performance relative to performance with the automation in a fully functional state.

Comparing the performance with these two type of automation present after participants experience a "failure" of said automation with performance prior to the failure would allow one to answer the question: to what extent, if any, is post-automation failure performance affected? It was hypothesized that the performance following an acquisition or automation analysis failure would be poorer compared to pre-failure performance.

Chapter 3

Materials and Methods

3.1 Overview

Participants, mostly students, were recruited from OSU and the surrounding community to participate in the study. They were split into three groups: a control group with no automation present, an experimental group with acquisition auomation present, and another experimental group with analysis automation present. They went through five trials with the newly developed Extended Task Management Environment (ETME), a domain agnostic low fidelity simulator for concurrent task management. They then went through seven experimental trials, with the special automation present in the two experimental groups failing in the fourth trial.

3.2 Participants

The target enrollment number for this study was 60 participants, though ultimately only 42 participants were involved (three groups of 14 each). Participants were recruited via the use of fliers (see appendix **??**) displayed on campus and the surrounding community. Participants indicated interest in the study (after seeing the flier) by emailing the Researcher, to schedule a time to come to the OSU Human Factors lab in BAT 050. The identities of persons who expressed interest in participating, as well as those who ultimately participated remain confidential.

Consent was obtained via a written documentation of informed consent (see appendix 7.2) at the point of study – the OSU Human Factors Lab at BAT 050. The consent process remained private, as only the student researcher and participant were in the lab prior to (and during) the study. Participants had the opportunity to discuss any of the points addressed on the consent form with the student researcher prior to signing.

Participants in the study were compensated with a $10 gift certificate to an on-campus cafe (Java, Java II, eCafe). Participants were informed that early withdrawal from the study would result in the participant receiving no compensation. Ultimately, no participants withdrew early from the study.

There were no discernible risks to the individuals who participated in this study.

There were no direct benefits to the individuals who participated in this study. Given the domain-agnostic design of our study, the results should be relevant to designers of any human-machine system which utilizes some de-

gree of automation.

3.3 Materials

Trials were conducted at the Human Factors Lab in the basement of Batcheller Hall at Oregon State University (room 050). A table, a chair, a mouse, and mid 2010 Macbook Pro (see Table 3.1 for relevant machine specs) running the ETME software with Processing 1.51 were used to conduct the experiment.

Operating System:	OSX 10.6 Snow Leopard
RAM:	4GB
Processor:	2.4 GHz Intel Core 2 Duo
Display Size:	13in
Display Resolution:	1280 by 800

Table 3.1: Test machine specifications

3.3.1 The ETME

The Extended Task Management Environment (ETME) is a variant of the Task Management Environment (TME). The TME was originally developed by Nicolalde (Nicolalde et al., 2003) as a domain agnostic platform to conduct research related to concurrent task management. Beside updating the code to a more modern, cross-platform compatible language (Processing), the ETME was written with additional data recording capabilities, extreme flexibility in adjusting task parameters, the capacity to work on a wide array of monitor sizes and resolutions, and a new task type.

Default Specifications

The ETME is fundamentally comprised of multiple, simultaneously displayed subsystems. The user must concurrently manage the state of the multiple subsystems (concurrent task managment). Each subsystem contains a status indicator in the form of a bar. To the left of the status indicator is a status gauge with green, yellow, and red markings. To the right of the status indicator is a large button. Framing all of these elements is a border (see Figure 3.1).

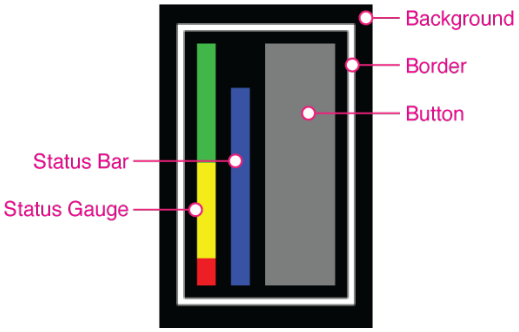

Figure 3.1: Individual subsystem default specifications

The gauge represents three possible states the task could be in, corresponding to the three colors present: green, yellow, and red. Green is RGB(0,192,0), yellow is RGB(255,255,0), red is RGB(255,0,0). Green makes up the top 50% of the status bar, yellow makes up the middle 40% of the status bar, and red makes up the bottom 10% of the status bar.

The alignment of the top of the status bar against the gauge determines the state of the current task. The status bar is blue RGB(0,0,255) in color. Left unattended (i.e., participant is not pressing the button to the right of the status bar), the status bar will drop at a certain rate, D.

The button is to the right of the status bar. By clicking and holding the

cursor over the button, attending to the task, one may raise the state of the task at a certain rate, R. The button is grey RGB(88,88,88) in color and when held is darker grey RGB(123,123,123) in color to serve as a visual confirmation that the participant is indeed pressing the button.

Each task in the ETME can be in a state defined by the combination of status bar and button. The task can be unattended (button not pressed) or attended (button pressed) and green, yellow, or red.

Variables

As mentioned above, the status for each task decays at a certain rate, D. When a task is attended to, the status is restored at a certain rate, R. The status can not exceed 100%, nor drop below 0%.

Scoring

The total score, $S(T)$, is a function of the total number of tasks on screen, and the status history of each task, as described in Equation 3.1. The status history of each task, $q_i(t)$, is described by Equation 3.3, with one point awarded for a task with the status bar at a state equal to or greater than 50% total, zero points awarded for a task with the status bar between 50% and 10% total, and one point lost for a task with the status bar equal to or less than 10% of total.

$$S(T) = \sum_{i=1}^{n} \sum_{t=0}^{T} w_i q_i(t) \tag{3.1}$$

$$\begin{aligned} i &= \text{task index} \\ n &= \text{number of tasks} \\ w_i &= \text{weight of task} \\ t &= \text{time index} \\ T &= \text{duration of task} \\ s &= \text{status of task} \end{aligned} \tag{3.2}$$

$$q_i(t) = \begin{cases} 1 & \text{if } s \geq 50 \\ 0 & \text{if } 50 > s > 10 \\ -1 & \text{if } s \leq 10 \end{cases} \tag{3.3}$$

Described in another way: participants earn one point for every moment in time (0.1 seconds, in the case of the ETME) a task is in the green state, earn zero points for every moment in time a task is in the yellow state, and lose one point for every moment in time a task is in the red state. At each moment in time the points for all tasks are summed, added to the total score, and the final score for a trial is the total score at the end time for the trial.

Acquisition Automation Specification

A task can have Acquisition Automation turned on. This has two effects. First, the border color is changed to reflect the current state. So when the status bar is aligned with the green section of the gage, the border is green; when it is aligned with the yellow section of the gage, the border is yellow; when it is aligned with the red section of the gage, the border is red. Second, when the task is in a YELLOW or RED state, the border also pulses.

When the task is in a yellow state, the color of the border changes as represented in Figure 3.3. The effect perceived by the participant is that of a pulsing yellow border.

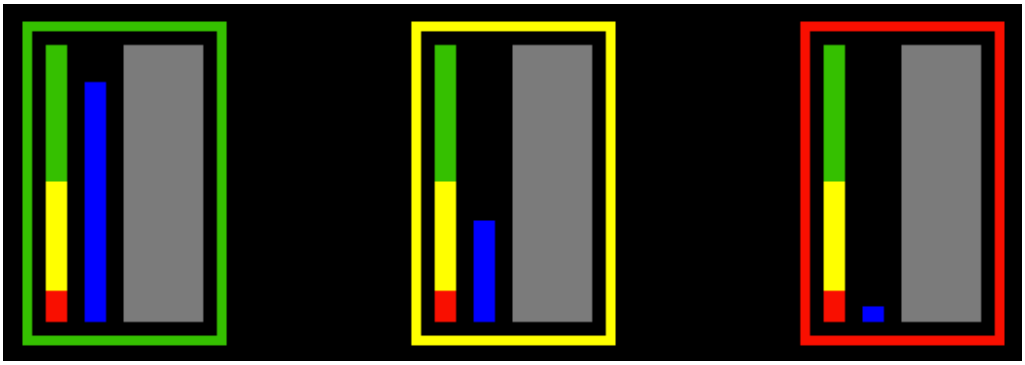

Figure 3.2: Acquisition automation mode in three different states

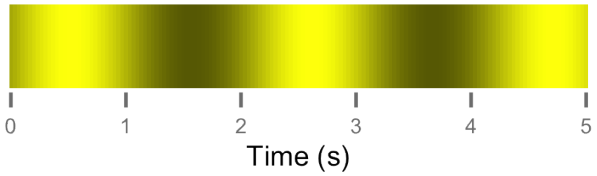

Figure 3.3: Border color vs time for an acquisition automation task in the yellow state

When the task is in a red state, the color of the border changes as represented in Figure 3.4. The effect perceived by the participant is that of a blinking red border.

Both of these task behavior changes (border changing color the reflect state and pulsing or blinking accordingly) are examples of aquisition automation, as they add additional information to the system to draw the attention of the operator to a certain state, but leaves any action taken on the system up to the operator.

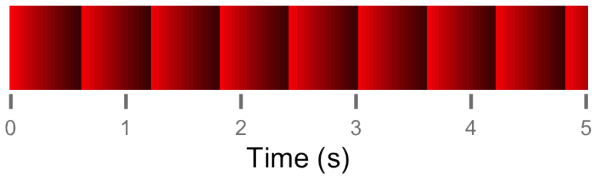

Figure 3.4: Border color vs time for an acquisition automation task in the red state

Analysis Automation Specification

A task can have analysis automation turned on. This adds a countdown timer to the bottom left of the frame which displays the number of seconds remaining until the task reaches a red state (see Figure 3.5, in this case 6.1 seconds remain until the task status reaches the red state).

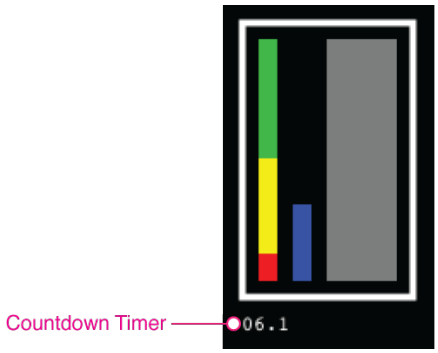

Figure 3.5: Task in analysis automation mode with countdown timer

This display of additional information is an example of analysis automation as it takes current system state information and provides information to the operator as to future system state, and much like acquisition automation leaves any action taken on the system as a result of this information up to the operator.

Recording Capabilities

During each trial the ETME records mouse position (X and Y coordinates on screen), whether the mouse is clicked and held or not, the state of each task (status bar), and whether the task is being attended to or not. For each trial all of these pieces of information are recorded to a text file (for processing later) every 1/10th of a second during the duration of the experiment.

These log files were designed to later be compiled via an additional processing script, and then cleaned up for analysis via an R script.

Mouse X	Mouse Y	Task 1 Click		Task 1 Score		Task 1 State		Task 1 Decay		Task 2 Click		Task 2 Score		Task 2 State	
293	302	0	1	0	-141	0	1	0	-141	0	1	0	-141	0	1	0	-141	0	1	0	-141
293	302	0	2	0	-141	0	2	0	-140	0	2	0	-141	0	2	0	-141	0	2	0	-141
293	302	0	3	0	-140	0	3	0	-140	0	3	0	-140	0	3	0	-141	0	3	0	-141
289	292	0	4	0	-140	0	4	0	-139	0	4	0	-140	0	4	0	-140	0	4	0	-141
259	245	0	5	0	-140	0	5	0	-139	0	5	0	-140	0	5	0	-140	0	5	0	-140
177	155	0	6	0	-140	0	6	0	-139	0	6	0	-139	0	6	0	-140	0	6	0	-140
164	147	0	7	0	-139	0	7	0	-138	0	7	0	-139	0	7	0	-140	0	7	0	-140
159	146	0	8	0	-139	0	8	0	-138	0	8	0	-139	0	8	0	-139	0	8	0	-138
159	146	0	9	0	-139	0	9	0	-137	0	9	0	-138	0	9	0	-139	0	9	0	-138
155	148	0	10	0	-138	0	10	0	-137	0	10	0	-138	0	10	0	-139	0	10	0	-137
154	149	0	11	0	-138	0	11	0	-136	0	11	0	-138	0	11	0	-139	0	11	0	-137
153	149	0	12	0	-138	0	12	0	-136	0	12	0	-137	0	12	0	-138	0	12	0	-137
151	149	0	13	0	-137	0	13	0	-135	0	13	0	-137	0	13	0	-138	0	13	0	-136
150	149	0	14	0	-137	0	14	0	-135	0	14	0	-136	0	14	0	-138	0	14	0	-136
149	149	0	15	0	-137	0	15	0	-134	0	15	0	-136	0	15	0	-138	0	15	0	-135
147	150	0	16	0	-136	0	16	0	-134	0	16	0	-136	0	16	0	-137	0	16	0	-135
145	150	0	17	0	-136	0	17	0	-133	0	17	0	-135	0	17	0	-137	0	17	0	-135
144	150	0	18	0	-136	0	18	0	-133	0	18	0	-135	0	18	0	-137	0	18	0	-134
143	150	0	19	0	-135	0	19	0	-132	0	19	0	-135	0	19	0	-137	0	19	0	-134

Figure 3.6: An example of raw ETME data recorded during one trial for one participant

3.4 Research Questions

3.4.1 Question 1

Does introducing acquisition or analysis automation enhance operator performance versus a system where these types of automation are present?

We can answer this by comparing the average scores over the first three experimental trials for the automation and acquisition analysis groups against

the same average scores for the control group.

3.4.2 Question 2

What effect does automation failure have on the performance of a participant who has learned to operate a system with acquisition or automation analysis always present and functional?

We can answer this by comparing the failure performance against the pre-failure performance within the three groups.

3.4.3 Question 3

To what extent, if any, is post-automation failure performance affected? We can answer this by comparing the post automation performance with the pre automation performance within the three groups.

3.5 Procedure

Upon signing the informed consent document, the participant was randomly assigned to one of three groups: control, acquisition automation, or analysis automation. From here, the study proceeded in five steps: filling out a short survey (see Appendix 7.3), receiving an introduction to the simulator, training, completing measured trials (and automation failure for the experimental groups), and receiving a debriefing. First, the student researcher asked the participant to fill out the participant survey. Then the student researcher introduced the version of the ETME that the participant would be using. The

scoring method, interface, and controls were explained. The participant then completed one familiarization trial, lasting three minutes. After the first trial was over, the participant was given the opportunity to ask for clarification regarding system operation. Next, the researcher explained that the participant would complete four additional trials for practice, followed by seven trials worth of measured trials.

After the practice trials were over, the participant was given the opportunity to take a short break. Next, the measured trials began. The control group completed seven trials without experiencing any change to the simulator set-up. The acquisition and automation analysis groups experienced normal operation for trails #1 through #3, an automation failure at trail #4, and normal operation for trails #5 through #7. Participants generally took between 50 and 60 minutes to complete all trials.

3.5.1 Group A: Control

	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Task 7	Task 8
Decay Rate	1.67%	0.213%	0.354%	0.567%	0.531%	0.142%	0.567%	0.567%
Restore Rate	2.8%	2.8%	2.8%	2.8%	2.8%	2.8%	2.8%	2.8%

Table 3.2: Task decay and restore rates, per second

The ETME configuration for the control group consisted of eight tasks arranged in a four (wide) by two (tall) grid (see Figure 3.7) with tasks one through four across the top from left to right and tasks five through eight across the bottom from left to right. Each of these tasks was configured with slightly different decay rates (detailed in Table 3.2) and equivalent restoration rate (the

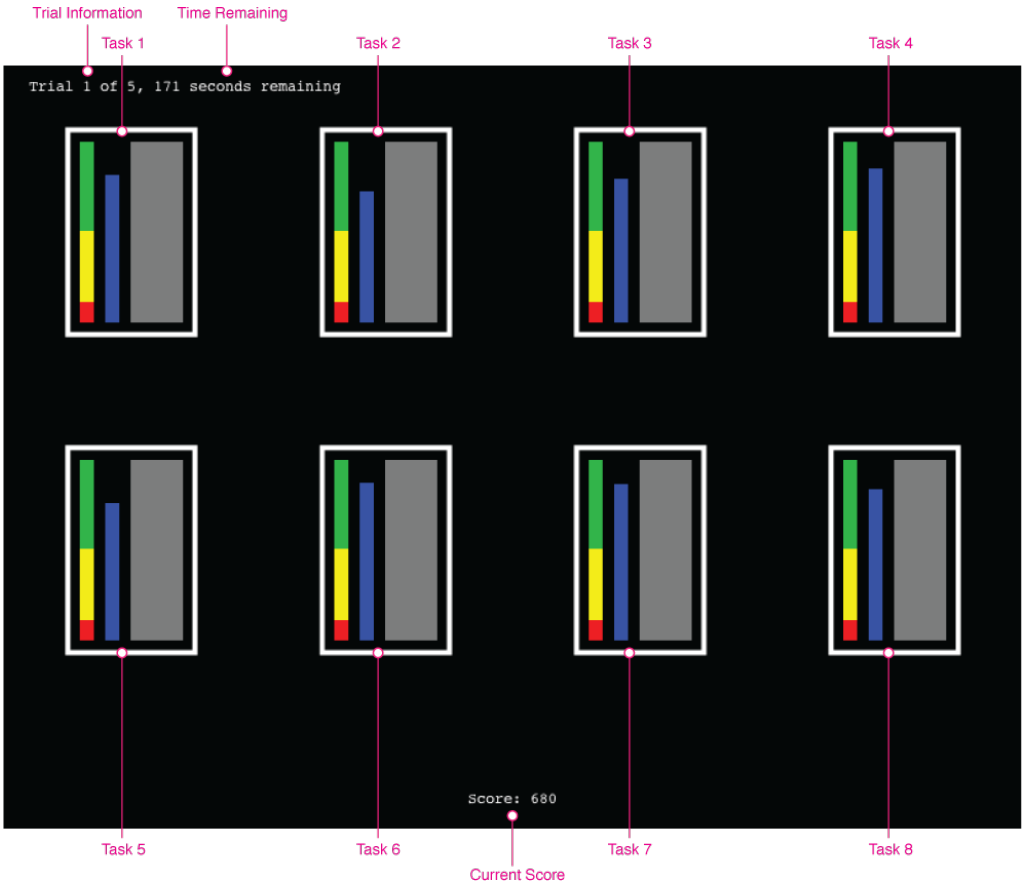

Figure 3.7: ETME layout configured for control group

rate at which the tasks status would be restored while being attended to). Task location, decay rate, and restore rate did not vary across practice or measured trials.

None of the acquisition or analysis automation was switched on for any tasks for the control group configuration. Participants first completed five practice trials (to learn the system) and then seven measured trials. Each trial lasted for 180 seconds.

3.5.2 Group B: Acquisition Automation

The ETME configuration for the first experimental group consisted of eight tasks arranged in a four (wide) by two (tall) grid (see Figure 3.7) with tasks one through four across the top from left to right and tasks five through eight across the bottom from left to right. Each of these tasks was configured with the same slightly different decay rates (detailed in Table 3.2) and equivalent restoration rates as the control group. Like the control group, task location, decay rate, and restore rate did not vary across practice or measured trials.

Acquisition automation was switched on for all tasks for the control group configuration. Participants first completed five practice trials (to learn the system) and then seven measured trials. During trial number four of the measured trial set the acquisition automation was switched off for all tasks (to simulate failure). The subsequent three trials had acquistion automation turned back on. Each trial lasted for 180 seconds.

3.5.3 Group C: Analysis Automation

The ETME configuration for the second experimental group consisted of eight tasks arranged in a four (wide) by two (tall) grid (see Figure 3.7) with tasks one through four across the top from left to right and tasks five through eight across the bottom from left to right. Each of these tasks was configured with the same slightly different decay rates (detailed in Table 3.2) and equivalent restoration rates as the control group. Like the control group, task location, decay rate, and restoration rate did not vary across practice or measured trials.

Analysis automation was switched on for all tasks for the control group

configuration. Participants first completed five practice trials (to learn the system) and then seven measured trials. During trial number four of the measured trial set the analysis automation was switched off for all tasks (to simulate failure). The subsequent three trials had acquistion automation turned back on. Each trial lasted for 180 seconds.

Chapter 4

Results

4.1 Demographics

There were 42 participants in this study, all OSU students, divided into three groups with 14 participants per group. They ranged in age from 19 to 39 years old with a mean age of 24 years and standard deviation of 4.79 years. There were 7 female particpants and 35 male particpants (16.6 percent female, 73.4 percent male).

4.2 Performance

Group A (Control)

In order to determine performance of participants within Group A across all trials a paired-samples t-test was conducted (see Table 4.1) to compare the final score for each trial within Group A against the final scores for all other

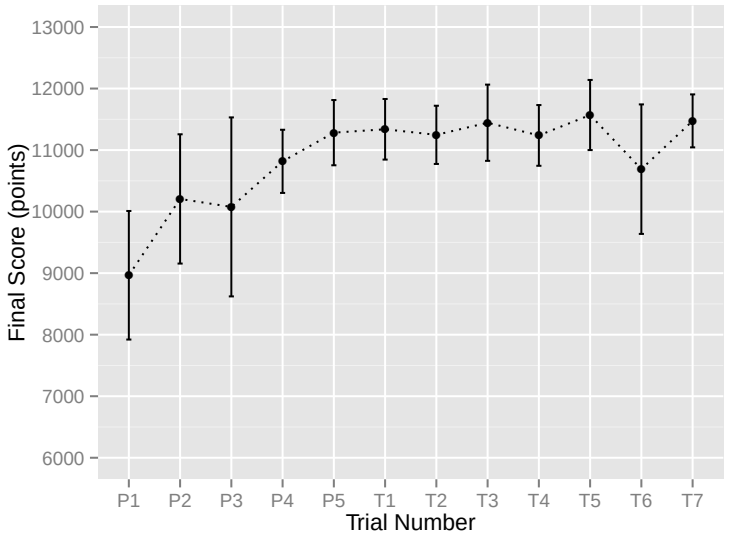

Figure 4.1: Group A (Control) final scores with 95 percent confidence intervals on the means (no automation failure in Trial 4)

trials within Group A.

Figure 4.1 shows the final scores for Group A with 95 percent confidence intervals on the mean. Visual inspection of this graph shows a general leveling out of the mean final score and a reduction of variance over the course of the experiment.

	Practice 1	Practice 2	Practice 3	Practice 4	Practice 5	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Trial 6
Practice 2	0.0130	–	–	–	–	–	–	–	–	–	–
Practice 3	0.1251	0.8336	–	–	–	–	–	–	–	–	–
Practice 4	0.0018	0.1179	0.2554	–	–	–	–	–	–	–	–
Practice 5	0.0015	0.0833	0.1254	0.1836	–	–	–	–	–	–	–
Trial 1	0.0009	0.0536	0.0991	0.1399	0.8761	–	–	–	–	–	–
Trial 2	0.0019	0.1112	0.1734	0.2812	0.9030	0.7403	–	–	–	–	–
Trial 3	0.0026	0.0980	0.1382	0.1281	0.5880	0.7346	0.3889	–	–	–	–
Trial 4	0.0027	0.1360	0.1522	0.2958	0.8786	0.7718	0.9681	0.4173	–	–	–
Trial 5	0.0011	0.0654	0.0860	0.1031	0.2828	0.4575	0.1134	0.6212	0.1383	–	–
Trial 6	0.0492	0.5662	0.5422	0.8394	0.3175	0.2392	0.1888	0.0907	0.1629	0.0925	–
Trial 7	0.0010	0.0595	0.0971	0.0948	0.4805	0.6404	0.2606	0.8904	0.2464	0.6184	0.0592

Table 4.1: Group A (Control) final scores paired t tests p-values

Group B (Acquisition Automation)

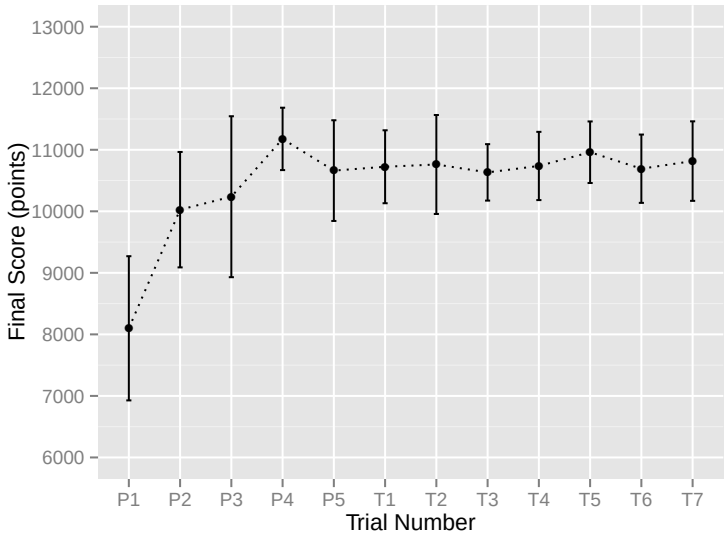

Figure 4.2: Group B (Acquisition Automation) final scores with 95 percent confidence intervals on the means (automation failure in Trial 4)

In order to determine performance of participants within Group B across all trials a paired-samples t-test was conducted (see Table 4.2) to compare the final score for each trial within Group B against the final scores for all other trials within Group B.

Figure 4.2 shows the final scores for Group B with 95 percent confidence intervals on the mean. Visual inspection of this graph shows a clear leveling out of the mean final score and a reduction of variance over the course of the experiment. The homogeneity of the mean and variance values over the course of the measured trials (Trial 1 through Trial 7) suggests that participants were no longer in the process of learning the system or experimenting with alternate strategies.

Table 4.2 reports the p-values from the aformentioned paired-samples t-test for Group B. Of particular note are the values for Trial 4 compared to all other trials since it was during Trial 4 that the acquisition automation was turned off for Group B. There is a statistically significant difference between Trial 4 and Practice 1. Group B participants beformed better on Trial 4 compared to Practice 1. This is expected, and likely a result of learning the system. However, there are no statistically significant differences between Trial 4 and any other trial during the experiment [1].

	Practice 1	Practice 2	Practice 3	Practice 4	Practice 5	Trial 1	Trial 2	Trial 3	Trial 4 (Automation Failure)	Trial 5	Trial 6
Practice 2	0.0069	–	–	–	–	–	–	–	–	–	–
Practice 3	0.0270	0.6903	–	–	–	–	–	–	–	–	–
Practice 4	0.0002	0.0268	0.1811	–	–	–	–	–	–	–	–
Practice 5	0.0023	0.2166	0.4976	0.2197	–	–	–	–	–	–	–
Trial 1	0.0005	0.2790	0.5547	0.2297	0.9046	–	–	–	–	–	–
Trial 2	0.0009	0.3070	0.5213	0.3645	0.8156	0.9099	–	–	–	–	–
Trial 3	0.0009	0.2110	0.5218	0.1280	0.9569	0.7968	0.7977	–	–	–	–
Trial 4	0.0013	0.1872	0.4616	0.1939	0.8930	0.9710	0.9646	0.5424	–	–	–
Trial 5	0.0005	0.1493	0.3157	0.4870	0.5771	0.3578	0.6256	0.2017	0.2993	–	–
Trial 6	0.0017	0.2926	0.5495	0.1538	0.9541	0.9045	0.8564	0.8251	0.8576	0.1007	–
Trial 7	0.0014	0.2186	0.4557	0.3576	0.7598	0.7168	0.8981	0.6104	0.8204	0.6290	0.6882

Table 4.2: Group B (Acq. Automation) final scores paired t tests p-values

Group C (Analysis Automation)

In order to determine performance of participants within Group C across all trials a paired-samples t-test was conducted (see Table 4.3) to compare the final score for each trail within Group C against the final scores for all other trials within Group C.

Figure 4.3 shows the final scores for Group C with 95 percent confidence intervals on the mean. Visual inspection of this graph shows a clear leveling out of the mean final score and a reduction of variance over the course

[1] Confirmed by a one-way repeated measures ANOVA to compare the effect of (IV) Acquisition Automation state on (DV) Final Score before (T1, T2, T3), during (T4), and after (T5, T6, T7) automation failure. There was not a significant effect of the IV, $F(2,26) = 0.215$, $p = 0.808$.

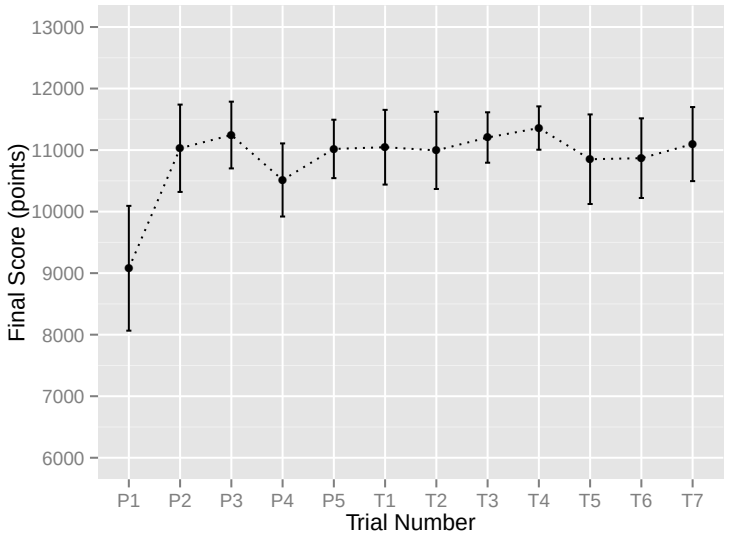

Figure 4.3: Group C (Analysis Automation) final scores with 95 percent confidence intervals on the mean (automation failure in Trial 4)

of the experiment. The homogeneity of the mean and variance values over the course of the measured trials (Trail 1 through Trial 7) suggests that participants were no longer in the process of learning the system or experimenting with alternate strategies.

Table 4.3 reports the p-values from the aformentioned paired-samples t-test for Group C. Of particular note are the values for Trial 4 compared to all other trials since it was during Trial 4 that the acquisition automation was turned off for Group C. There is a statistically significant difference between Trail 4 and Practice 1. In other words, Group C participants beformed better on Trial 4 compared to Practice 1. This is expected, and likely a result of learning the system. However, there were no statistically significant differences be-

tween Trial 4 and any other trial during the experiment [2].

	Practice 1	Practice 2	Practice 3	Practice 4	Practice 5	Trial 1	Trial 2	Trial 3	Trial 4 (Automation Failure)	Trial 5	Trial 6
Practice 2	0.0016	–	–	–	–	–	–	–	–	–	–
Practice 3	0.0011	0.3677	–	–	–	–	–	–	–	–	–
Practice 4	0.0222	0.2815	0.0832	–	–	–	–	–	–	–	–
Practice 5	0.0044	0.9749	0.3643	0.0664	–	–	–	–	–	–	–
Trial 1	0.0033	0.9664	0.5995	0.1854	0.9217	–	–	–	–	–	–
Trial 2	0.0063	0.9245	0.4498	0.3086	0.9333	0.8216	–	–	–	–	–
Trial 3	0.0023	0.6705	0.8895	0.0825	0.4776	0.6326	0.5054	–	–	–	–
Trial 4	0.0005	0.4098	0.7063	0.0236	0.2743	0.4303	0.3254	0.5224	–	–	–
Trial 5	0.0120	0.7613	0.4751	0.4271	0.7328	0.6971	0.7837	0.3519	0.0986	–	–
Trial 6	0.0056	0.7727	0.4344	0.3836	0.7495	0.7228	0.8039	0.3469	0.0863	0.9307	–
Trial 7	0.0053	0.9007	0.7422	0.1172	0.8497	0.9142	0.8305	0.6976	0.3553	0.2887	0.2503

Table 4.3: Group C (Analysis Automation) final scores paired t tests p-values

Between all groups

Figure 4.4 shows the final scores compared between all groups on a per trial basis with 95 percent confidence intervals on the mean. Visual inspection shows that there was statistically no difference between the mean scores for each group on a per trail basis. The t-test results between groups for each trial in in Table 4.4 through Table 4.15 confirm this.

Figure 4.5 shows the same data as Figure 4.4, but it is plotted to better show the performance trend on a trial by trial basis within groups. Again, visual inspection confirms the t-test results, there was statistically no difference (beyond the first practice trial) between the mean scores for each trial within groups [3].

[2] Confirmed by a one-way repeated measures ANOVA to compare the effect of (IV) Analysis Automation state on (DV) Final Score before (T1, T2, T3), during (T4), and after (T5, T6, T7) automation failure. There was not a significant effect of the IV, $F(2,26) = 0.519$, $p = 0.601$.

[3] Confirmed by a two-way repeated measures ANOVA to compare the main effects of type of automation and automation state and the interaction effect between type of automation and automation state on final score. Type of automation consisted of three levels (none, acquisition automation, analysis automation) and automation state of three leves (before failure, during failure, and after failure). No effects were statistically significant at the .05 level. The main effect for type of automation yielded an F ratio of $F(2,285)=1.459$, $p = 0.234$. The main effect for automation state yielded an F ratio of $F(2,285)=0.042$, $p = 0.959$. The interaction effect was also not significant at $F(4,285) = 0.109$, $p = 0.979$.

	Group A	Group B
Group B	0.46	–
Group C	0.91	0.40

Table 4.4: Practice 1 final scores t tests p-values

	Group A	Group B
Group B	0.86	–
Group C	0.34	0.27

Table 4.5: Practice 1 final scores t tests p-values

	Group A	Group B
Group B	0.89	–
Group C	0.19	0.28

Table 4.6: Practice 3 final scores t tests p-values

	Group A	Group B
Group B	0.61	–
Group C	0.70	0.40

Table 4.7: Practice 4 final scores t tests p-values

	Group A	Group B
Group B	0.41	–
Group C	0.71	0.64

Table 4.8: Practice 5 final scores t tests p-values

	Group A	Group B
Group B	0.46	–
Group C	0.72	0.75

Table 4.9: Trial 1 final scores t tests p-values

	Group A	Group B
Group B	0.54	–
Group C	0.73	0.80

Table 4.10: Trial 2 final scores t tests p-values

	Group A	Group B
Group B	0.35	–
Group C	0.72	0.53

Table 4.11: Trial 3 final scores t tests p-values

	Group A	Group B
Group B	0.56	–
Group C	0.86	0.47

Table 4.12: Trial 4 final scores t tests p-values

	Group A	Group B
Group B	0.45	–
Group C	0.37	0.91

Table 4.13: Trial 5 final scores t tests p-values

	Group A	Group B
Group B	1.00	–
Group C	0.83	0.85

Table 4.14: Trial 6 final scores t tests p-values

	Group A	Group B
Group B	0.40	–
Group C	0.59	0.75

Table 4.15: Trial 7 final scores t tests p-values

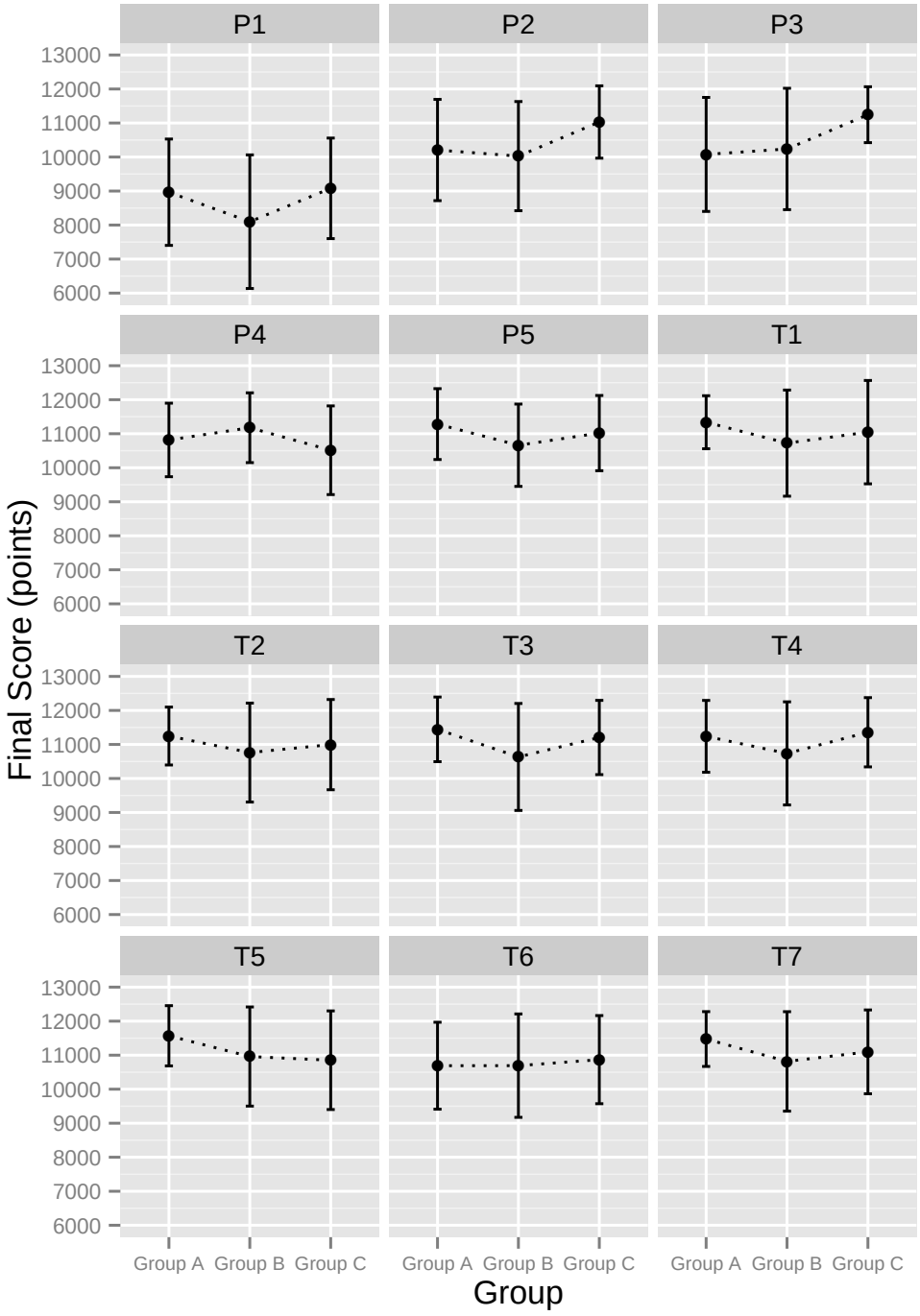

Figure 4.4: Final scores with 95 percent confidence intervals on the means

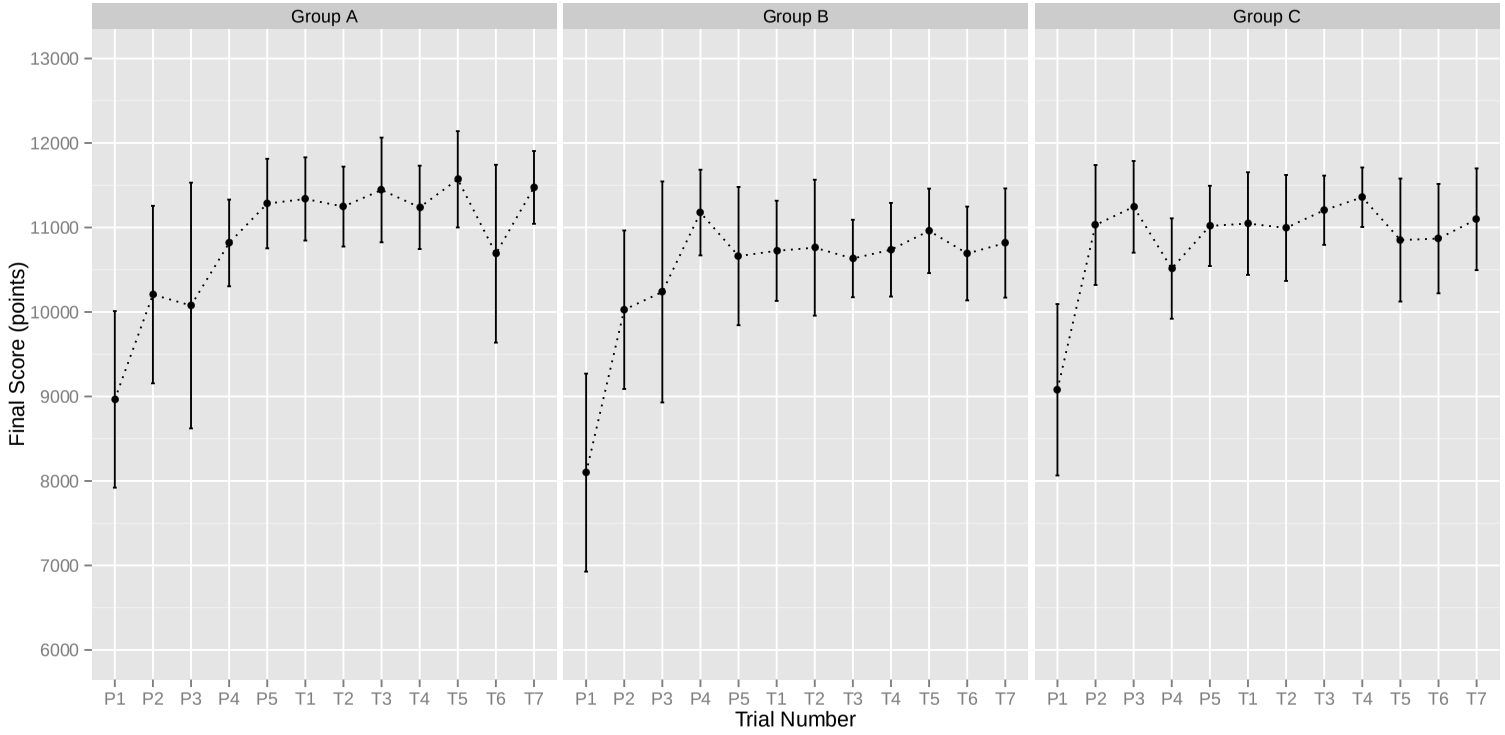

Figure 4.5: Final scores with 95 percent confidence intervals on the means

Chapter 5

Discussion

5.1 Group A (Control)

Over the course of the five practice trials (P1 through P5) there was an increase in the mean final score leading into a leveling out of the mean final score and reduction in variance around the mean final score toward the end of the practice trials. The mean and variance remained nearily identical over the remaining seven experimental trials (T1 through T7). This strongly suggests that the participants were through any learning curve by the time they entered the experimental trials.

5.2 Group B (Acquisition Automation)

The same pattern in performance (increase in mean of final score and reduction in variance of the mean of the final scores over the practice trails leading into nearly identical mean/variance over the course of the experimental tri-

als) was observed in Group B. During the automation failure trial (T4), no decrease in mean performance was observed. No change in mean performance was observed after in the three trials post automation failure (T5 through T7).

5.3 Group C (Analysis Automation)

Much like Group A and Group B, Group C quickly learned the system and their mean final scores remained constant over the course of the experimental trials. No decrease in mean performance was observed in the automation failure trial (T4), nor was any change in performance observed in the post automation failure trials (T5 through T7).

5.4 Hypothesis 1

The null hypothesis was that there is no difference in performance between the three groups (Group A: no automation, Group B: acquisition automation, Group C: analysis automation) during the first three trials.

Visual inspection of Figure 4.4 for the T1, T2, and T3 trials clearly shows a failure to reject the null hypothesis. There was statistically no difference between the mean scores for each group during these three trials.

If the automation present in a system is neither useful (improves operator performance) nor harmful (degrades operator performance) one wonders whether inclusion of the automation is necessary.

5.5 Hypothesis 2

The null hypothesis was that automation failure has no effect on the performance of participants who have learned to operate a system with acquisition or automation analysis always present.

Visual inspection of Figure 4.2 across trials T1 through T7 shows a failure to reject the null hypothesis. Likewise, Visual inspection of Figure 4.3 across trials T1 through T7 shows a failure to reject the null hypothesis. This was confirmed by running paried t-tests (see Table 4.2 and Table 4.3). There was statistially no difference between the mean scores for each trial relative to the failed state trial (T4) for these two groups.

This suggests that the failure of low level automation in systems where the automation has no measurable improvement to performance (per the first result) will result in no measurable decrease in concurrent task management performance during failed automation.

5.6 Hypothesis 3

The null hypothesis was that performance post-automation-failure would not be effected relative to performance pre-automation-failure.

Visual inspection of Figure 4.2 across trials T1 through T7 shows a failure to reject the null hypothesis. There was statistially no difference between the mean scores for the pre automation failure trials (T1 through T3) relative to the post automation trials (T5 through T7) for this group. Likewise, visual inspection of Figure 4.3 across trials T1 through T7 shows a failure to reject

the null hypothesis. There was statistially no difference between the mean scores for the pre automation failure trials (T1 through T3) relative to the post automation trials (T5 through T7) for this group.

5.7 Additional Discussion

Given that there was no difference in performance between the three groups at different levels of automation (none, acquisition automation, analysis automation), no difference within group performance when automation fails, and no difference in performance within groups post-automation failure the question arises: does low level automation help at all with concurrent task management? Or, alternately, did the ETME configuration used in the study encounter a ceiling effect?

The ETME is an abstract system with a simple, singular goal for the operator and in this experiment only one type of automation was present at a time. The system was built to isolate a specific set of conditions and be easy to study, which is not the case with most systems outside the lab. While the results mentioned above suggest that implementing a type of low level automation in a system will not necessarily improve concurrent task management performance, it does not rule out that there could be systems where a combination (that is, more than one automation present) of automated aspects of a system demonstrate some emergent benefit relative to any measurement of their performance, in isolation, would suggest. That is, while a single instance of low level automation may not be beneficial, multiple instances in combi-

nation may be beneficial. In the same way, failure of individual instances of automation may show no impact, but failure of multiple instances of low level automation could. Further study would be required to answer these questions.

It is also possible that single instances of low level automation are beneficial (and failure detrimental) in systems where concurrent task management performance is measured in a different way or the goals of the operator are different. In the ETME performance was measured by an overall score, with the participants being instructed to get the highest score possible. This implies there are benefits to an ever increasing score, and this may map well to some real world scenarios, but it is easy to imagine other systems where the goal is to stay above some minimum performance threshold and doing better is of little consequence or offers diminishing returns (for example, think of keeping a car in a lane in the highway, this can only be done so well). The fact that the presence of automation may not have a measurable performance impact does not mean that failure of automation would therefore result in no measureable performance drop. Hence the previously mentioned concern about a ceiling effect is valid but certainly many real world scenarious are subject to ceiling effects as well. In order to more accurately discuss performance we may need to be more specific about what is meant by performance, much in the same way that Parasuraman provided a way to be more specific about what is meant by automation (Parasuraman et al., 2000). In the review of the literature for this study no discussion of different types of performance or goals in the context of concurrent task management were found. This suggests that further

study may be required.

Given the questions raised above the inability of many of these studies in the literature review to extend their results and conclusions to real world scenarios (outside the lab) is unsurprising. Being more specific about different types of performance and the interaction between multiple instances and levels of automation may make it easier to extend lab results to the field (or vice versa).

Chapter 6

Conclusions and Recommendations

Motivated by the fact that literature on concurrent task management includes many low fidelity simulations, none of which include any degree of automation, this study used a low fidelity simulator with three modes of automation (none, acquisition automation, analysis automation) to measure the effect of automation on human concurrent task management performance.

Participants were divided into three groups: Group A had no automation present in the system, Group B had acquisition automation present in the system, and Group C had analysis automation present in the system. All groups ran through five practice trials of a particular ETME configuration (to familiarize themselves with the low fidelity simulator), followed by seven experimental trials. The two experimental groups (Group B and Group C) experienced an automation failure (removal of automation from the system) in experimental

trial four of seven.

Three null hypotheses were tested. First that there was no difference in performance between the three groups, second that trial performance within an experimental group during an automation failure state was no different than during a non-failed state, and third that trial performance within an experimental group immediately following an automation failure state was no different than during a non-failed state.

A comparison between the three groups showed no difference in performance, meaning the presence of either acqusitions or analysis automation had no effect relative to the control no-automation group. For the two groups with automation present, both were subjected to a trial where the automation was removed to simulate failure. Comparing the scores for trials prior to the failure with the scores during the failure showed no difference in performance for either group. For these two groups the scores after the simulated failure were compared with the scores prior to the simulated failure, and again no difference in performance was detectable. This study failed to reject all three null hypotheses.

If automation is introduced into a system that has no measurable performance impact it does not necessarily imply that failure of the automation would also result in no performance decrease, nor does it imply that performance post failure would also be unimpacted. However, the results of this study and their failure to reject the three null hypoetheses do suggest that this is the case. Repeating this study and adjusting the parameters of the ETME and the design of the automation in such a way that the automation in the

experimental groups shows a measurable increase in performance vs no automation would better match the study to the reasonable assumption that automation added to a system is there to improve performance (even though the mere addition of automation may not always be beneficial, as this study clearly demonstrated). This approach would likely require multiple design, testing, and redesign cycles in order to get to a state where the automation is measurably beneficial to performance and therefore would take more time and participants to complete.

From there the group and trial setup would mirror this study. But one would be able to confirm that low level automation can be beneficial in improving concurrent task management performance and answer whether, given such beneficial performance, there are risks of decreased performance during an automation failure state.

References

Aeronautics and space, 14 c.f.r. § 61. (2017).

Bahner, J. E., Hüper, A.-D., & Manzey, D. (2008). Misuse of automated decision aids: Complacency, automation bias and the impact of training experience. *International Journal of Human-Computer Studies*, *66*(9), 688 - 699. Retrieved from http://www.sciencedirect.com/science/article/pii/S1071581908000724 doi: http://dx.doi.org/10.1016/j.ijhcs.2008.06.001

Burge, R., & Chaparro, A. (2012). The effects of texting and driving on hazard perception. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *56*(1), 715-719. Retrieved from http://pro.sagepub.com/content/56/1/715.abstract doi: 10.1177/1071181312561149

Chen, J., & Funk, K. (2003, May 17-21). A fuzzy model of human task management performance. *Proceedings of the IIE 2003 Annual Conference, Portland, OR*.

Chou, C.-C., Madhaven, D., & Funk, K. (1996). Studies of cockpit task management errors. *The International Journal of Aviation Psychology*, *6*(4),

307-320.

Clyne, B. (2012). Multitasking in emergency medicine. *Academic Emergency Medicine*, *19*(2), 230–231. Retrieved from http://dx.doi.org/10.1111/j.1553-2712.2011.01265.x doi: 10.1111/j.1553-2712.2011.01265.x

Colvin, K., Funk, K., & Braune, R. (2005). Task prioritization factors: Two part-task simulator studies. *The International Journal of Aviation Psychology*, *15*(4), 321-338.

DiVita, J., Obermayer, O., Nugent, W., & Linville, J. M. (2004). Verification of the change blindness phenomenon while managing critical events on a combat information display. *Human Factors*, *46*, 205-218.

Endsley, M. R. (1995). Towards a theory of situation awareness. *Human Factors*, *37*(1), 32-64.

Endsley, M. R. (1999). Level of automation effects on performance, situation awareness and workload in a dynamic control task. *Ergonomics*, *42*(3), 462-492. Retrieved from http://dx.doi.org/10.1080/001401399185595 (PMID: 10048306) doi: 10.1080/001401399185595

Funk, K. (1991, 285). Cockpit task management: Preliminary definitions, normative theory, error taxonomy, and design recommendations. *The International Journal of Aviation Psychology*, *1*(4), 271.

Greenstein, J. S., & Rouse, W. B. (1982, March). A model of human decisionmaking in multiple process monitoring situations. *Systems, Man and Cybernetics, IEEE Transactions on*, *12*(2), 182-193. doi:

10.1109/TSMC.1982.4308802

Joslyn, S., & Hunt, E. (1998). Evaluating individual differences in response to time-pressure situations. *Journal of Experimental Psychology: Applied*, *4*(1), 16-43.

Kozak, J. J., Hancock, P. A., Arthur, E. J., & Chrysler, S. T. (1993). Transfer of training from virtual reality. *Ergonomics*, *36*(7), 777-784.

Molesworth, B. R. C., & Chang, B. (2009). Predicting pilots' risk-taking behavior through an implicit association test. *Human Factors*, *51*, 845-857.

Nicolalde, J., Funk, K., & Uttl, B. (2003). How well do cognitive abilities predict concurrent task management performance? *Institute of Industrial Engineers Annual Conference, Portland, OR*.

Nikolic, M. I., & Sarter, N. B. (2007). Flight deck disturbance management: A simulator study of diagnosis and recovery from breakdowns in pilot-automation coordination. *Human Factors*, *49*, 553-563.

Onnasch, L., Wickens, C. D., Li, H., & Manzey, D. H. (2014). Human performance consequences of stages and levels of automation an integrated meta-analysis. *Human Factors*, *56*(3), 476-488.

Parasuraman, R., & Riley, V. (1997). Humans and automation: Use, misuse, disuse, abuse. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *39*(2), 230-253. Retrieved from `http://hfs.sagepub.com/content/39/2/230.abstract` doi: 10.1518/001872097778543886

Parasuraman, R., Sheridan, T. B., & Wickens, C. D. (2000, May). A model for types and levels of human interaction with automation. *IEEE TRANS-*

ACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, *30*(3), 286-297.

Parkes, A. M., & Coleman, N. (1990). Route guidance systems: A comparison of methods presenting directional information to the driver. In E. J. Lovesey (Ed.), (chap. Contemporary ergonomics 1990). London: Taylor and Francis.

Pashler, H. (1989). Dissociations and contingencies between speed and accuracy: Evidence for a two-component theory of divided attention in simple tasks. *Cognitive Psychology*, *21*, 469-514.

Polson, M. C., & Friedman, A. (1988). Task-sharing within and between hemispheres: A multiple-resources approach. *Human Factors*, *30*, 633-643.

Shakeri, S., & Funk, K. (2007). A comparison of human and near-optimal task management behavior. *Human Factors*, *49*(3), 400-416. doi: 10.1518/001872007X197026

Sheridan, T. B., & Verplank, W. L. (1978). *Human and computer control of undersea teleoperators* (Tech. Rep.). DTIC Document.

Tulga, M. K., & Sheridan, T. B. (1980). Dynamic decisions and work load in multitask supervisory control. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS*, *10*(5), 217-232.

Wickens, C. D. (1980). The structure of attentional resources. In R. Nickerson (Ed.), (p. 239-257). Hillsdale, NJ: Erlbaum.

Wickens, C. D. (1984). Processing resources in attention. In R. Parasuraman & R. Davies (Eds.), (p. 63-101). New York: Academic Press.

Wickens, C. D. (1991). Processing resources and attention. In D. Damos (Ed.), (chap. Multiple task performance). London: Taylor and Francis.

Wickens, C. D., & Hollands, J. G. (2000). *Engineering psychology and human performance* (Third Edition ed.). Upper Saddle River, New Jersey: Prentice Hall.

Wickens, C. D., & Liu, Y. (1988). Codes and modalities in multiple resources: A success and a qualification. *Human Factors, 30,* 599-616.

Chapter 7

Appendix

7.1 ETME Code

7.1.1 Group Code

```
1  Task myTask1;
2  Task myTask2;
3  Task myTask3;
4  Task myTask4;
5  Task myTask5;
6  Task myTask6;
7  Task myTask7;
8  Task myTask8;
9
10 PrintWriter OUTPUT;
11 PFont font;
```

```
12  int mode = 0; // determines mode program is in.  0=
        intro , 1= main , 2= terminate
13  int timer_1 = 0; // timer variable, 30 increments
        equivalent to one second
14  int trial_number = 1;
15  int trial_total = 5;
16  int time_remaining = 0;
17  int currentscore = 0;
18
19  void setup() {
20    size(800,600);
21    smooth();
22    frameRate(30);
23    font = loadFont("Monospaced-12.vlw");
24    textFont(font);
25    myTask1 = new Task(50,50, -0.235, 0.4);
26    myTask2 = new Task(250,50,-0.03,0.4);
27    myTask3 = new Task(450,50,-0.05,0.4);
28    myTask4 = new Task(650,50, -0.08, 0.4);
29    myTask5 = new Task(50,300,-0.075,0.4);
30    myTask6 = new Task(250,300, -0.02, 0.4);
31    myTask7 = new Task(450,300, -0.08, 0.4);
32    myTask8 = new Task(650,300, -0.08, 0.4);
33  }
```

```
34
35 void draw () {
36   background(0);
37
38   //  ========================================
39   //  Intro screen goes here  (mode == 0)
40   //  ========================================
41   if (trial_number < trial_total+1) {
42       if (mode == 0) {
43         text("PRESS ’S’ TO START",20,20);
44         if(keyPressed) {
45             if (key == ’s’ || key == ’S’) {
46               timer_1 = 0;
47               taskReset();
48               mode = 1;
49             }
50           }
51       }
52
53       //  ========================================
54       //  Main program goes here (mode == 1)
55       //  ========================================
56
57
```

```
58
59        if (mode == 1) {
60          background(0);
61          if (timer_1 < 5401) { //run for three minutes
62            time_remaining = ceil((5400-timer_1)/30);
63            stroke(255);
64            fill(255);
65            text("Trial␣" + trial_number + "␣of␣" +
                  trial_total + ",␣" + time_remaining + "␣
                  seconds␣remaining",20,20);
66            taskStart_none();
67            totalScore();
68            timer_1 = timer_1 + 1;
69          }
70          if (timer_1 == 5401) {
71            stroke(255);
72            fill(255);
73            text("TRIAL␣COMPLETE",20,20);
74            dataRecorder();
75            timer_1 = timer_1 + 1;
76          }
77          if (timer_1 > 5401) {
78            stroke(255);
79            fill(255);
```

```
80            text("TRIAL " + trial_number + " COMPLETE!"
                ,20,20);
81            text("YOUR SCORE: " + currentscore,20,40);
82            text("PRESS 'S' TO CONTINUE",20,60);
83            if(keyPressed) {
84              if (key == 's' || key == 'S') {
85                timer_1 = 0;
86                trial_number = trial_number + 1;
87                taskReset();
88              }
89            }
90          }
91          }
92
93
94        //  ========================================
95        //  Termination screen goes here  (mode == 2)
96        //  ========================================
97
98
99        if (mode == 2) {
100         text("MODE 2",width/100,height/100);
101       }
102   }
```

```
103     if (trial_number > trial_total) {
104       stroke(255);
105       fill(255);
106       text("SECTION␣COMPLETE␣--␣PLEASE␣NOTIFY␣STUDENT␣
              RESEARCHER");
107       //text("TRIAL " + trial_number + " COMPLETE!",20,20)
              ;
108       //text("YOUR SCORE: " + currentscore,20,40);
109       //text("PRESS 'S' TO START NEXT TRIAL",20,60);
110       //text("PRACTICE COMPLETE! ",20,100);
111     }
112 }
113
114 void keyPressed(){
115   switch(key){
116     case '0':
117     mode = 0;
118     break;
119     case '1':
120     mode = 1;
121     break;
122     case 'x':
123     dataRecorder();
124     break;
```

```
125     }
126 }
127
128 void dataRecorder() {
129     int d = day();      // Values from 1 - 31
130     int mo = month();   // Values from 1 - 12
131     int y = year();     // 2003, 2004, 2005, etc.
132     int s = second();   // Values from 0 - 59
133     int m = minute();   // Values from 0 - 59
134     int h = hour();     // Values from 0 - 23
135     OUTPUT = createWriter(y + "_" + mo + "_" + d + "_" + h
            + "_" + m + "_" + s + "_" + "trial_" +
            trial_number + ".txt");
136     OUTPUT.println("Mouse␣X" + "\t" + "Mouse␣Y" + "\t" + "
            Task␣1␣Click" + "\t" + "Task␣1␣Score" + "\t" + "
            Task␣1␣State" + "\t" + "Task␣1␣Decay" + "\t" + "
            Task␣2␣Click" + "\t" + "Task␣2␣Score" + "\t" + "
            Task␣2␣State" + "\t" + "Task␣2␣Decay" + "\t" + "
            Task␣3␣Click" + "\t" + "Task␣3␣Score" + "\t" + "
            Task␣3␣State" + "\t" + "Task␣3␣Decay" + "\t" + "
            Task␣4␣Click" + "\t" + "Task␣4␣Score" + "\t" + "
            Task␣4␣State" + "\t" + "Task␣4␣Decay" + "\t" + "
            Task␣5␣Click" + "\t" + "Task␣5␣Score" + "\t" + "
            Task␣5␣State" + "\t" + "Task␣5␣Decay" + "\t" + "
```

```
        Task␣6␣Click" + "\t" + "Task␣6␣Score" + "\t" + "
        Task␣6␣State" + "\t" + "Task␣6␣Decay" + "\t" + "
        Task␣7␣Click" + "\t" + "Task␣7␣Score" + "\t" + "
        Task␣7␣State" + "\t" + "Task␣7␣Decay" + "\t" + "
        Task␣7␣Click" + "\t" + "Task␣8␣Score" + "\t" + "
        Task␣8␣State" + "\t" + "Task␣8␣Decay");
137   for(int i = 0; i < 1800; i++) {
138     OUTPUT.println(myTask1.recorded_x[i] + "\t" +
          myTask1.recorded_y[i] + "\t" + myTask1.
          recorded_click[i] + "\t" + myTask1.recorded_score
          [i] + "\t" + myTask1.recorded_state[i] + "\t" +
          myTask1.recorded_decay[i] + "\t" + myTask2.
          recorded_click[i] + "\t" + myTask2.recorded_score
          [i] + "\t" + myTask2.recorded_state[i] + "\t" +
          myTask2.recorded_decay[i] + "\t" + myTask3.
          recorded_click[i] + "\t" + myTask3.recorded_score
          [i] + "\t" + myTask3.recorded_state[i] + "\t" +
          myTask3.recorded_decay[i] + "\t" + myTask4.
          recorded_click[i] + "\t" + myTask4.recorded_score
          [i] + "\t" + myTask4.recorded_state[i] + "\t" +
          myTask4.recorded_decay[i] + "\t" + myTask5.
          recorded_click[i] + "\t" + myTask5.recorded_score
          [i] + "\t" + myTask5.recorded_state[i] + "\t" +
          myTask5.recorded_decay[i] + "\t" + myTask6.
```

```
            recorded_click[i] + "\t" + myTask6.recorded_score
            [i] + "\t" + myTask6.recorded_state[i] + "\t" +
            myTask6.recorded_decay[i] + "\t" + myTask7.
            recorded_click[i] + "\t" + myTask7.recorded_score
            [i] + "\t" + myTask7.recorded_state[i] + "\t" +
            myTask7.recorded_decay[i] + "\t" + myTask8.
            recorded_click[i] + "\t" + myTask8.recorded_score
            [i] + "\t" + myTask8.recorded_state[i] + "\t" +
            myTask8.recorded_decay[i]);
139     }
140     OUTPUT.flush();
141     OUTPUT.close();
142   }
143
144   void taskReset() {
145     myTask1 = new Task(50,50, -0.1044, 0.7833);
146     myTask2 = new Task(250,50, -0.1567, 0.7833);
147     myTask3 = new Task(450,50, -0.1175, 0.7833);
148     myTask4 = new Task(650,50, -0.0854, 0.7833);
149     myTask5 = new Task(50,300,-0.1343, 0.7833);
150     myTask6 = new Task(250,300, -0.0723, 0.7833);
151     myTask7 = new Task(450,300, -0.0783, 0.7833);
152     myTask8 = new Task(650,300, -0.0940, 0.7833);
153   }
```

```
154
155 void taskStart_all() {
156     myTask1.display();
157     myTask1.signal();
158     myTask1.countdown();
159     myTask1.score();
160     myTask1.mousetrack();
161     myTask2.display();
162     myTask2.signal();
163     myTask2.countdown();
164     myTask2.score();
165     myTask3.display();
166     myTask3.signal();
167     myTask3.countdown();
168     myTask3.score();
169     myTask4.display();
170     myTask4.signal();
171     myTask4.countdown();
172     myTask4.score();
173     myTask5.display();
174     myTask5.signal();
175     myTask5.countdown();
176     myTask5.score();
177     myTask6.display();
```

```
178     myTask6.signal();
179     myTask6.countdown();
180     myTask6.score();
181     myTask7.display();
182     myTask7.signal();
183     myTask7.countdown();
184     myTask7.score();
185     myTask8.display();
186     myTask8.signal();
187     myTask8.countdown();
188     myTask8.score();
189   }
190
191   void taskStart_none() {
192     myTask1.display();
193   //    myTask1.signal();
194   //    myTask1.countdown();
195     myTask1.score();
196     myTask1.mousetrack();
197     myTask2.display();
198   //    myTask2.signal();
199   //    myTask2.countdown();
200     myTask2.score();
201     myTask3.display();
```

```
202  //    myTask3.signal();
203  //    myTask3.countdown();
204       myTask3.score();
205       myTask4.display();
206  //    myTask4.signal();
207  //    myTask4.countdown();
208       myTask4.score();
209       myTask5.display();
210  //    myTask5.signal();
211  //    myTask5.countdown();
212       myTask5.score();
213       myTask6.display();
214  //    myTask6.signal();
215  //    myTask6.countdown();
216       myTask6.score();
217       myTask7.display();
218  //    myTask7.signal();
219  //    myTask7.countdown();
220       myTask7.score();
221       myTask8.display();
222  //    myTask8.signal();
223  //    myTask8.countdown();
224       myTask8.score();
225  }
```

```
226
227 void taskStart_signal() {
228     myTask1.display();
229     myTask1.signal();
230 //    myTask1.countdown();
231     myTask1.score();
232     myTask1.mousetrack();
233     myTask2.display();
234     myTask2.signal();
235 //    myTask2.countdown();
236     myTask2.score();
237     myTask3.display();
238     myTask3.signal();
239 //    myTask3.countdown();
240     myTask3.score();
241     myTask4.display();
242     myTask4.signal();
243 //    myTask4.countdown();
244     myTask4.score();
245     myTask5.display();
246     myTask5.signal();
247 //    myTask5.countdown();
248     myTask5.score();
249     myTask6.display();
```

```
250     myTask6.signal();
251   //  myTask6.countdown();
252     myTask6.score();
253     myTask7.display();
254     myTask7.signal();
255   //  myTask7.countdown();
256     myTask7.score();
257     myTask8.display();
258     myTask8.signal();
259   //  myTask8.countdown();
260     myTask8.score();
261   }
262
263   void taskStart_countdown() {
264     myTask1.display();
265   //  myTask1.signal();
266     myTask1.countdown();
267     myTask1.score();
268     myTask1.mousetrack();
269     myTask2.display();
270   //  myTask2.signal();
271     myTask2.countdown();
272     myTask2.score();
273     myTask3.display();
```

```
274  //    myTask3.signal();
275       myTask3.countdown();
276       myTask3.score();
277       myTask4.display();
278  //    myTask4.signal();
279       myTask4.countdown();
280       myTask4.score();
281       myTask5.display();
282  //    myTask5.signal();
283       myTask5.countdown();
284       myTask5.score();
285       myTask6.display();
286  //    myTask6.signal();
287       myTask6.countdown();
288       myTask6.score();
289       myTask7.display();
290  //    myTask7.signal();
291       myTask7.countdown();
292       myTask7.score();
293       myTask8.display();
294  //    myTask8.signal();
295       myTask8.countdown();
296       myTask8.score();
297  }
```

```
298
299
300 void totalScore() {
301   currentscore = myTask1.game_score + myTask2.game_score
          + myTask3.game_score + myTask4.game_score +
         myTask5.game_score + myTask6.game_score + myTask7.
         game_score + myTask8.game_score;
302   textAlign(CENTER);
303   stroke(255);
304   fill(255);
305   text("Score: " + currentscore,width/2,height-20);
306   textAlign(LEFT);
307 }
```

7.1.2 Task Code

```
1  class Task {
2
3    String disp_time;
4    float i = 0;
5    float j = 0;
6    boolean flag = true; // decay toggle, decaying while
         true
7    float x_position; // x location of task bounding box
8    float y_position; // y location of bounding box
```

```
9   float box_width = 100; //width of bounding box
10  float box_height = 161; //height of boudning box
11  float w = 10; // width of status colors and status bar
12  float gap = 10; // element seperation distance for
        task box
13  color bound_color = color(255,255,255); // color of
        bounding box
14  color button_color; // color of correction button
15  color status_color; //color of task status
16  color green_status = color(0,192,0); // color of green
        status
17  color yellow_status = color(255,255,0);
18  float yellow_status_r = 100;
19  float yellow_status_g = 100;
20  float yellow_status_b = 0;
21  int red_status_r = 255;
22  int red_status_g = 0;
23  int red_status_b = 0;
24  color red_status = color(255,0,0);
25  float decay_rate; // decay rate of status
26  float correction_rate; // correction rate of status
27  float decay_height = -(box_height - (2*gap));
28  float time_remaining; // time until task reaches red
        state
```

```
29    int game_decay = 0;
30    int game_score = 0;
31    int game_state = 0;
32    int game_click = 0;
33    int game_x = 0;
34    int game_y = 0;
35    int score_index = 0;
36    int[] recorded_decay = new int[1800];
37    int[] recorded_score = new int[1800];
38    int[] recorded_state = new int[1800];
39    int[] recorded_click = new int[1800];
40    int[] recorded_x = new int[1800];
41    int[] recorded_y = new int[1800];
42
43    Task(float temp_x_position, float temp_y_position,
          float temp_decay_rate, float temp_correction_rate)
          {
44      x_position = temp_x_position;
45      y_position = temp_y_position;
46      decay_rate = temp_decay_rate;
47      correction_rate = temp_correction_rate;
48    }
49
50    void display() {
```

```
51      stroke(bound_color);
52      strokeWeight(4);
53      fill(0);
54      rect(x_position,y_position,box_width,box_height);
55      strokeWeight(1);
56      stroke(0,192,0);
57      fill(0,192,0);
58      //text(task_score,Xpos,Ypos-3);
59      rect(x_position+gap,y_position+gap,w, (box_height -
            (2*gap))*0.50);
60      stroke(255,255,0);
61      fill(255,255,0);
62      rect(x_position+gap,y_position+gap+((box_height -
            (2*gap))*0.50), w, (box_height - (2*gap))*0.40);
63      stroke(255,0,0);
64      fill(255,0,0);
65      rect(x_position+gap,y_position+gap+((box_height -
            (2*gap))*0.50)+((box_height - (2*gap))*0.40), w,
            (box_height - (2*gap))*0.10);
66      stroke(button_color);
67      fill(button_color);
68      rect(x_position+gap+w+gap+w+gap, y_position+gap,
            box_width - (gap+gap+gap+gap+w+w), box_height -
            (2*gap));
```

```
69      if (mousePressed == true && mouseX > x_position+gap+
            w+gap+w+gap && mouseX < (x_position+gap+w+gap+w+
            gap) + (box_width - (gap+gap+gap+gap+w+w)) &&
            mouseY > y_position+gap && mouseY < (y_position+
            gap) + (box_height - (2*gap))) {
70        flag = false;
71        button_color = 88;
72      } else {
73        flag = true;
74        button_color = 123;
75      }
76      stroke(0,0,255);
77      fill(0,0,255);
78      rect(x_position+gap+w+gap, y_position+gap+(
            box_height-(2*gap)), w, floor(decay_height));
79      // if (((-(box_height - (2*gap)))/decay_height) > 0
            && flag == true) {
80      if (decay_height < 0 && flag == true) {
81        decay_height = decay_height - decay_rate;
82      }
83      if (decay_height > -(box_height - (2*gap)) && flag
            == false) {
84        decay_height = decay_height - correction_rate;
85      }
```

```
86    }
87
88    void signal() {
89      if (i > 6.28) {
90        i = 0;
91      }
92      if (decay_height < -(box_height - (2*gap))*0.50) {
93        bound_color = green_status;
94      }
95      if (decay_height < -(box_height - (2*gap))*0.10 &&
            decay_height > -(box_height - (2*gap))*0.50) {
96        bound_color = color(yellow_status_r,
              yellow_status_g,yellow_status_b);
97        yellow_status_r = 180 + (75*sin(i));
98        yellow_status_g = 180 + (75*sin(i));
99        //if (yellow_status_r < 100) {
100         //yellow_status_r = 255;
101         //yellow_status_g = 255;
102       //}
103     }
104     if (decay_height > -(box_height - (2*gap))*0.10) {
105       bound_color = color(red_status_r,red_status_g,
              red_status_b);
106       red_status_r = red_status_r - 10;
```

```
107         if (red_status_r < 80) {
108           red_status_r = 255;
109         }
110       }
111       i = i + 0.1;
112     }
113
114     void countdown() {
115       if (decay_height < -(box_height - (2*gap))*0.10) {
116         time_remaining = (decay_height+((box_height - (2*
              gap))*0.10))/(30*decay_rate);
117         time_remaining = float(floor(time_remaining*10))
              /10;
118         disp_time = nf(time_remaining, 2,1);
119       }
120       //println(time_remaining);
121       if (decay_height > -(box_height - (2*gap))*0.10) {
122         time_remaining = 0;
123       }
124       fill(255);
125       text(disp_time,x_position, y_position+box_height+gap
            +gap);
126     }
127
```

```
128   void score() {
129     if ((j == 3) && (score_index < 1800)) {
130       if (decay_height < -(box_height - (2*gap))*0.50) {
            //Green
131         game_score = game_score + 1;
132         game_state = 0;
133       }
134       if (decay_height < -(box_height - (2*gap))*0.10 &&
            decay_height > -(box_height - (2*gap))*0.50) {
            //Yellow
135         game_score = game_score + 0;
136         game_state = 1;
137       }
138       if (decay_height > -(box_height - (2*gap))*0.10) {
            //Red
139         game_score = game_score - 1;
140         game_state = 2;
141       }
142       if (flag == false) {
143         game_click = 1;
144       } else {
145         game_click = 0;
146       }
147       game_decay = floor(decay_height);
```

```
148         recorded_decay[score_index] = game_decay;
149         recorded_score[score_index] = game_score;
150         recorded_state[score_index] = game_state;
151         recorded_click[score_index] = game_click;
152         j = 0;
153         //println(game_score);
154         score_index = score_index + 1;
155       }
156       j = j + 1;
157     }
158
159     void mousetrack() {
160       if ((j == 3) && (score_index < 1800)) {
161         game_x = mouseX;
162         game_y = mouseY;
163         recorded_x[score_index] = game_x;
164         recorded_y[score_index] = game_y;
165       }
166     }
167
168 }
```

7.2 Informed Consent

School of Mechanical, Industrial and Manufacturing Engineering
Oregon State University, 204 Rogers Hall, Corvallis, Oregon 97331-6001
Phone 541-737-3441 | **Fax** 541-737-2600 | http://mime.oregonstate.edu

CONSENT FORM

Project Title:	The Effect of Acquisition and Analysis Automation Failure in a Low-Fidelity Multitasking System
Principal Investigator:	Kenneth H. Funk II
Student Researcher:	William Secor
Version Date:	5/1/2012

1. WHAT IS THE PURPOSE OF THIS FORM?

This form contains information you will need to help you decide whether to be in this study or not. Please read the form carefully and ask the study team member(s) questions about anything that is not clear.

2. WHY IS THIS STUDY BEING DONE?

The purpose of this study is to determine the effect of Analysis Automation and Acquisition Automation failure on human performance. Acquisition Automation is automation in a system that supports human sensory processes, such as the highlighting of an aircraft on a radar display. Analysis Automation is automation in a system that supports human cognitive functions, such as a radar display which projects the anticipated future path of aircraft. Most research on automation failures focus on Action Automation, in which a human is often times supervising the automated actions of a machine – such as in an auto-pilot system.

A low-fidelity multitasking simulator is being used in order to facilitate the generalization of our results.

This study is being conduct by William Secor for the completion of his Master's Thesis in Industrial Engineering.

Up to 60 participants may be invited to take part in this study.

3. WHY AM I BEING INVITED TO TAKE PART IN THIS STUDY?

You are being invited to take part in this study because we are seeking adult participants who are capable of operating a computer game.

4. WHAT WILL HAPPEN IF I TAKE PART IN THIS RESEARCH STUDY?

The study activities include filling out a short survey, a training period on our simulator (a simple computer game), and multiple trials in the simulator under varying conditions.

Study duration: It is expected that the total length of participation in this study will not exceed 60 minutes. You will only visit the lab once, and will not be asked to return for subsequent investigations.

Randomization: This study involves a process called randomization. Randomization means that you are put into one of the groups by chance. It is like drawing names from a hat. Neither you nor the people doing the study will choose what group you will be in. You will have a one in three chance of being placed in any group.

5. WHAT ARE THE RISKS AND POSSIBLE DISCOMFORTS OF THIS STUDY?

There are no foreseeable risks to participating.

6. WHAT ARE THE BENEFITS OF THIS STUDY?

This study is not designed to benefit you directly.

7. WILL I BE PAID FOR BEING IN THIS STUDY?

You will be compensated with a $10 gift card to Java Stop, Java II, eCafe upon completion of all study activities. You will not be compensated if you do not complete all study activities.

8. WHO WILL SEE THE INFORMATION I GIVE?

The information you provide during this research study will be kept confidential to the extent permitted by law. Research records will be stored securely and only researchers will have access to the records. Federal regulatory agencies and the Oregon State University Institutional Review Board (a committee that reviews and approves research studies) may inspect and copy records pertaining to this research. Some of these records could contain information that personally identifies you.

If the results of this project are published your identity will not be made public.

To help ensure confidentiality, we will keep all paper records in a locked file. All computer records will be identified by a participant code, and will be stored in a password protected location on the school's network.

9. WHAT OTHER CHOICES DO I HAVE IF I DO NOT TAKE PART IN THIS STUDY?

Participation in this study is voluntary. If you decide to participate, you are free to withdraw at any time without penalty. You will not be treated differently if you decide to stop taking part in the study. If you choose to withdraw from this project before it ends, the researchers may keep information collected about you and this information may be included in study reports.

Participation terminated by investigator: If you do not follow instructions for operation of the simulator, the student investigator may elect to terminate your participation in the study.

10. WHO DO I CONTACT IF I HAVE QUESTIONS?

If you have any questions about this research project, please contact: Kenneth H. Funk II at (541) 737-2357, or by email at funkk@engr.orst.edu

If you have questions about your rights or welfare as a participant, please contact the Oregon State University Institutional Review Board (IRB) Office, at (541) 737-8008 or by email at IRB@oregonstate.edu

Oregon State University IRB Study # 5272 Expiration Date 05/09/2013

11. WHAT DOES MY SIGNATURE ON THIS CONSENT FORM MEAN?

Your signature indicates that this study has been explained to you, that your questions have been answered, and that you agree to take part in this study. You will receive a copy of this form.

Do not sign after the expiration date: 05/09/2013

Participant's Name (printed): _____

_____ _____
(Signature of Participant) (Date)

_____ _____
(Signature of Person Obtaining Consent) (Date)

7.3 Participant Survey

Participant Survey

Please answer the following questions:

What is your name? _____

What is your age? _____

What is your gender? _____

What is your major (if applicable)? _____

7.4 R Cleanup Script

```
1  setwd('/Users/wsecor/Documents/OSU␣Docs␣Copy/Fall␣2012/
       That␣one␣time␣I␣wrote␣a␣thesis/R␣data␣processing')
2
3  # read txt files with names of the form Patient*.txt
4  txt_files = list.files(pattern = "*.txt")
5
6  # read txt files into a list (assuming separator is a
       comma)
7  data_list = lapply(txt_files, read.table, header=T,sep="
       \t",quote="\"")
8
9  #begin loop here
10
11 for(i in 1:length(data_list)){
12
13     #column name cleanup
14     colnames(data_list[[i]]) <- c(
15       "MOUSE_X",
16       "MOUSE_Y",
17       "TASK_1_CLICK",
18       "TASK_1_SCORE",
19       "TASK_1_STATE",
```

```
20          "TASK_1_DECAY",
21          "TASK_2_CLICK",
22          "TASK_2_SCORE",
23          "TASK_2_STATE",
24          "TASK_2_DECAY",
25          "TASK_3_CLICK",
26          "TASK_3_SCORE",
27          "TASK_3_STATE",
28          "TASK_3_DECAY",
29          "TASK_4_CLICK",
30          "TASK_4_SCORE",
31          "TASK_4_STATE",
32          "TASK_4_DECAY",
33          "TASK_5_CLICK",
34          "TASK_5_SCORE",
35          "TASK_5_STATE",
36          "TASK_5_DECAY",
37          "TASK_6_CLICK",
38          "TASK_6_SCORE",
39          "TASK_6_STATE",
40          "TASK_6_DECAY",
41          "TASK_7_CLICK",
42          "TASK_7_SCORE",
43          "TASK_7_STATE",
```

```
44          "TASK_7_DECAY",
45          "TASK_8_CLICK",
46          "TASK_8_SCORE",
47          "TASK_8_STATE",
48          "TASK_8_DECAY")
49
50        #add group column
51        data_list[[i]]$Group <- strsplit(txt_files[i], "_")
              [[1]][1]
52
53        #add participant column
54        data_list[[i]]$Participant <- strsplit(txt_files[i],
              "_")[[1]][2]
55
56        #add trial column and remove the .txt -- this does
              the split, checks length, and removes the last
              four characters
57        data_list[[i]]$Trial <- substr(strsplit(txt_files[i
              ], "_")[[1]][3],1,nchar(strsplit(txt_files[i], "_
              ")[[1]][3])-4)
58
59        #add thetime column
60        data_list[[i]]$TheTime <- c(1:1800)
61
```

```
62      #add final score column
63      data_list[[i]]$FinalScore <-  data_list[[i]]
            $TASK_1_SCORE +
64                      data_list[[i]]$TASK_2_SCORE +
65                      data_list[[i]]$TASK_3_SCORE +
66                      data_list[[i]]$TASK_4_SCORE +
67                      data_list[[i]]$TASK_5_SCORE +
68                      data_list[[i]]$TASK_6_SCORE +
69                      data_list[[i]]$TASK_7_SCORE +
70                      data_list[[i]]$TASK_8_SCORE
71  }
72
73  #merge everything together into one big happy dataframe!
74  mydata <- ldply(data_list, data.frame)
```