Auditory Interface Design to Support Rover Tele-Operation in the Presence of Background Speech: Evaluating the Effects of Sonification, Reference Level Sonification, and Sonification Transfer Function

by

Adrian Matheson

A thesis submitted in conformity with the requirements
for the degree of Master of Applied Science

Department of Mechanical and Industrial Engineering
University of Toronto

Auditory Interface Design to Support Rover Tele-Operation in the Presence of Background Speech: Evaluating the Effects of Sonification, Reference Level Sonification, and Sonification Transfer Function

Adrian Matheson

Master of Applied Science

Department of Mechanical and Industrial Engineering
University of Toronto

2013

Abstract

Preponderant visual interface use for conveying information from machine to human admits failures due to overwhelming the visual channel. This thesis investigates the suitability of auditory feedback and certain related design choices in settings involving background speech. Communicating a tele-operated vehicle's tilt angle was the focal application.

A simulator experiment with pitch feedback on one system variable, tilt angle, and its safety threshold was conducted. Manipulated in a within-subject design were: (1) presence vs. absence of speech, (2) discrete tilt alarm vs. discrete alarm and tilt sonification (continuous feedback), (3) tilt sonification vs. tilt and threshold sonification, and (4) linear vs. quadratic transfer function of variable to pitch.

Designs with both variable and reference sonification were found to significantly reduce the time drivers spent exceeding the safety limit compared to the designs with no sonification, though this effect was not detected within the set of conditions with background speech audio.

Acknowledgements

I would like to thank my thesis supervisor, Birsen Donmez, for her guidance in bringing this thesis to fruition, and for other research-related opportunities I have had in the past two years. Her kind assistance in explaining both the work required and background information on the workings of academia was very valuable. Early in the course of my recent studies I developed a deep appreciation of statistics, and I was thrilled to discover, only afterward, that she had extensively studied it formally and had a wealth of knowledge on the subject. I could scarcely imagine a better sounding board than her for my many practical and (unrelated) theoretical questions, and I am especially appreciative of having had her as my supervisor for this reason.

Special thanks are also reserved for my first contact in the Human Factors Engineering and Cognitive Ergonomics world, Paul Milgram. In addition to providing a warm welcome and explanation of the program and generous support for my application for admission later, he also turned out to be an extremely engaging professor in my first course on Engineering Psychology. I have to also thank Mark Chignell, Jonathan Rose, and Alison Smiley for making it the case that I cannot fathom having taken more interesting or worthwhile courses in my time at U of T.

I imagine this thesis as partially the result of help from every member of the HFASt and CEL laboratories and many of those of the ETC and IML labs as well as past classmates, but I would like to single out Wayne Giang, Antony Hilliard, and Pamela D'Addario. Recency effect compels me to thank Wayne and Antony for help with statistics software in particular, but through various other conversations they were also helpful, more than they probably know, in just generally getting me comfortable in the practice of science. Pamela was my nearly constant classmate and the ideal member of my cohort through this program. Her interest and ability in all of the courses we took together was a continual boost to the same in me, and discussions with her on various steps building up to our theses were always either heartening or illuminating.

Finally I would like to acknowledge support from outside the university from Piotr Jasiobedzki and Ho-Kong Ng in the execution of all of the rover experiments, from Mike LeSauvage and Mike Peters in arrangements for me to be able to pursue this degree in the first place, and of course from my family and my fantastically supportive partner in life and in big-city higher learning, Amy, in countless ways during this program and for years or decades beforehand.

Table of Contents

List of Tables

List of Figures

List of Appendices

Chapter 1
Introduction

Human-machine systems often involve the need for information transfer from machine to human. It is extremely common for such systems to make use of the visual medium, though human senses other than vision are sometimes employed. As an important example, capability for visual display is nearly universal in personal computing devices. These devices are extremely pervasive and their inbuilt display capabilities impact which senses can be engaged in human-machine systems that use these devices as platforms. The less mobile of these devices, such as desktop and laptop computers, can typically also output audio, and the more mobile of these devices, such as tablet computers and advanced cellular telephones, can often give tactile as well as auditory feedback, but visual displays remain ascendant.

There are good biological and psychological reasons for this state of affairs, but it may still be the case that systems should ply other human senses more often and in more ways to reduce reliance on the visual medium. Biology and psychology also support the idea of human hearing being the second best sense for information reception by humans in many cases (indeed the best for others), as explained in the next chapter.

The research described herein involved the use of an audio-based interface component within a primarily video-based interface for human remote driving of a vehicle. The component was an auditory display designed to convey the off-vertical tilt angle of a tele-operated rover in an environment with uneven ground. This component was studied as a rover can have a design tolerance of a maximum safe tilt angle, and it and other rugged-terrain rovers are expected to be fielded in locations where repairs made necessary due to exceeding rover design tolerances would be difficult to administer. The task was a driving task in which performance increased with operation nearer to but not exceeding the tilt angle safety limit for the vehicle. The contribution of this research to the literature is thus exploration of auditory display of a system variable about which knowledge is increasingly important approaching a single bound in one direction, and where the display is used for the fairly complex task of operating a rugged-terrain vehicle. This focus differs from much research on auditory displays, which involves conveying system variables whose different values are of similar importance or change in importance gradually over the range (e.g., breathing rate sonification (e.g., M. Watson & Sanderson, 2001) or heart rate sonification (e.g., Chou, Lim, Brant, Ford, & Ansermino, 2008)), or which are used for simpler tasks (e.g., auditory graphs, in which the sole task is determining the sonified value (Flowers, 2005)).

The task was designed such that there was benefit to operating near the safety limit in order to increase the benefit of feedback generally and to ensure that the precision of the display was an important factor in the performance of drivers. Both were goals as they would, in turn, make differences in usefulness between different interface designs more apparent. The experiment was meant to generate conclusions about sonification regardless of whether there is operation near the safety limit of a variable, but it can be considered an especially good indication of the value of sonification for any variable with this dynamic (where the variable correlates with performance up to a limit, beyond which performance abruptly drops).

Another example of such a variable would be the temperature in a chemical process catalyzed by an enzyme. Increasing temperature would correlate with increased reaction speed (in most cases) up until the temperature at which the enzyme (a protein) denatures, above which the speed would be very low or zero. Still another example, this time more likely to be linked with manual control and more likely to vary quickly (and thus potentially even more suitable for continuous display), would be passenger vehicle speed on public roadways, where higher speeds would mean faster travel but above a certain point would invite speeding tickets.

Various design choices related to the auditory feedback scheme were tested: employing auditory alarms, which are binary (on-off) auditory signals persistently reflecting a system being in one of two possible states, and sonifications, which are persistent and continuous (infinite-level) auditory indications of a continuous system variable or continuous set of system states. Note that reference values were sonified in some cases, where reference value sonification simply entailed sonification of a system parameter having a continuous range of possible values but happening to remain fixed at one value.

The auditory feedback schemes included providing an alarm alone versus also providing sonification, providing sonification of a system variable alone versus also providing sonification of a reference value, and providing sonifications with different mathematic relationships between input and output variables. The latter manipulation appears to be novel in terms of research on sonification.

The results of this experiment may to varying degrees be generalizable to displays for the same system parameter (tilt angle) for different vehicles (such as aircraft or particularly manoeuvrable watercraft or spacecraft), for different vehicle parameters (such as speed, or such as pitch or roll individually rather than combined as “off-vertical” angle), and for entirely different systems (such as industrial physical, chemical, or biological process systems where, for example, continuous monitoring of temperature would be beneficial).

The research also investigated the effectiveness of auditory feedback in environments where other interface components already employ the sense of hearing, particularly the fairly common circumstance of the human being exposed to voice communications. This exploration of auditory displays in the presence of voice, particularly the combination of sonification with voice, is expected to be another contribution to the literature. Results manipulation may be generalizable to the design of displays for other situations involving possible masking of audio display feedback by speech, especially those involving driving tasks or tasks of similar complexity.

Chapter 2
Background

1 Human Sense Differences and Mental Processing

The primacy of visual displays can be partially explained by the fact that the human sense of sight can carry information at a higher rate than can the other senses. Information theory is one way of identifying such capacity: quantifying information transmission based on the number of possible states of discrete transmissions, the probabilities of those states, and the number or rate of incidences of transmission (Shannon & Weaver, 1949). The resolution of the human eye and the number of detectable colour and brightness combinations, for example, are both factors in the number of possible transmission states, and the time required to register these data is a factor in the rate of transmission. The channel capacity of hearing is related to the number of differentiable possible sounds and the minimum amount of time for which each sound must be present in order to be detected (i.e. change detection time span). The same is true for touch, smell, taste, and other sensations with respect to the number of identifiably different configurations of input into these senses as well as associated change detection time spans.

The way probabilities of different input states factor into information rate reckoning can be illustrated by the fact that discovering the outcomes of fair coin tosses is more informative than discovering the outcomes of tosses of a biased coin. More uncertainty is present in the case of the unbiased coin, thus more uncertainty is resolved upon learning the outcomes, and more information is conveyed (information can be viewed as the reduction of uncertainty). There are natural probabilities of different transmission states in any given environment, but for display purposes probabilities matter less, as a display can be designed with any set of transmission state likelihoods desired. (Again, choosing for all transmission states to be equally likely allows for maximum information flow.)

The channel capacity of human vision is on the order of 10 Mb/s (10 million bits per second) (Koch et al., 2006). Lehrl and Fischer (1985, p. 154) estimated this same value, and gave those of other senses as 1 Mb/s for hearing, 400 kb/s for touch, 5 kb/s for thermoreception, 1 kb/s for proprioception, 20 b/s for smell, and 13 b/s for taste. These widely ranging output capacities, illustrated in Figure 1 (widely-ranging enough to make visual representation difficult), suggest an order of preference for engaging the different senses with interface components, and that after vision, hearing and touch should be preferred.

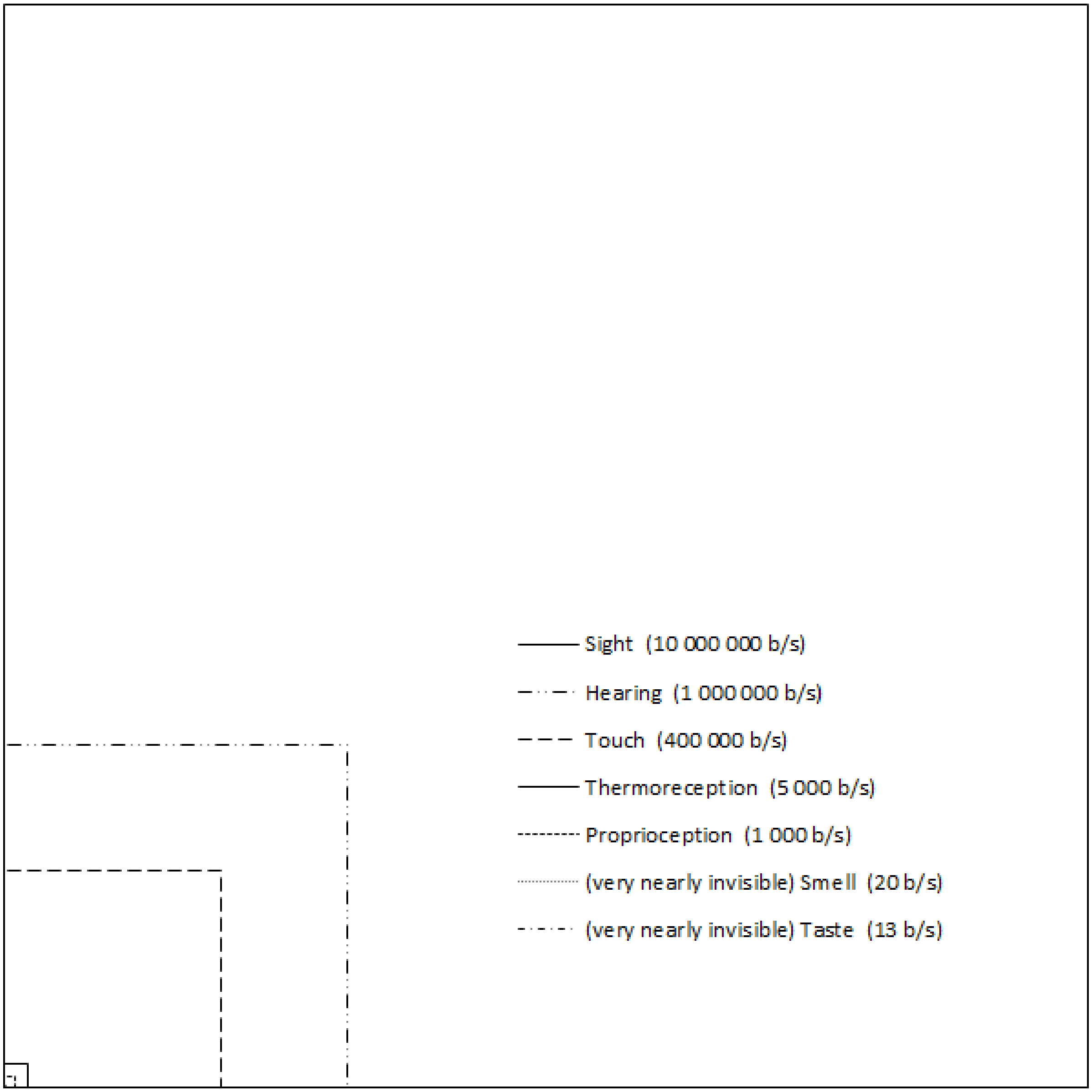


Figure 1 – Channel capacities of the human senses (maximum output to the brain) (indicated by area; areas overlap; smell and taste areas are that of a period ('.'))

Processing by the brain following raw input from the senses is another limiting factor for conveying information. For any task requiring non-trivial responses (i.e., not simply reflexes), a series of neurological structures and capabilities of the brain are required. The first of these resources is a short-term sensory store for each sensory system (Wickens & Hollands, 1999). Handling of sense information then involves varying amounts of attention, long-term memory, working memory (or short-term memory), and cognition, and ends with response selection and response execution. Figure 2, adapted from Wickens and Hollands (1999, p. 11), depicts this process.

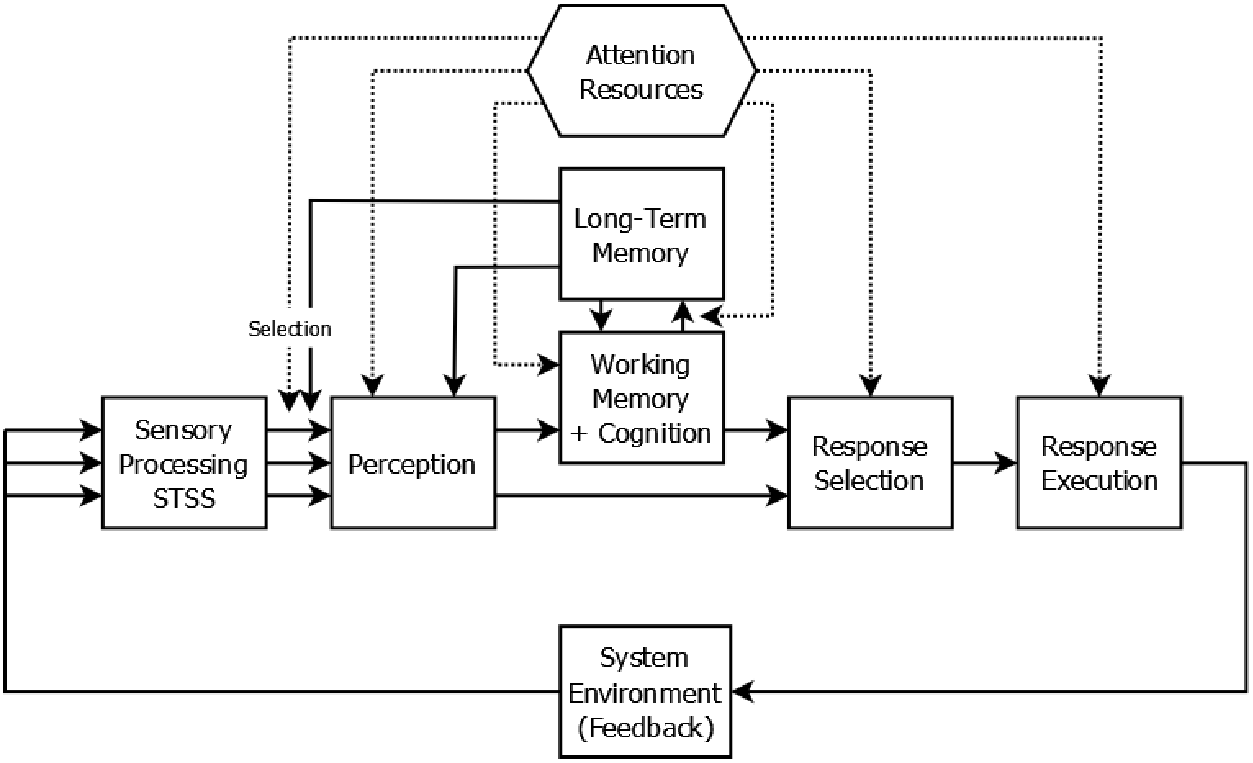


Figure 2 – Full processing of human sense information, for tasks requiring perception (adapted from Wickens and Hollands (1999))

The short-term sensory store processes most or all of the information output by the senses, but only a small proportion of sensory information proceeds past this point and is subject to conscious perception (Lachter, Forster, & Ruthruff, 2004). This fact and the complexity of the processing mean that the ultimate channel capacity available when engaging a sense does not follow directly from the sensory output channel capacity.

Another obstacle to straightforward determination of the usefulness of exercising a certain sense for a display element is that displays can be designed to convey system parameters via many different “dimensions” of a sense (e.g., for sound, intensity and frequency). Between senses there are different numbers of dimensions, and each dimension has a unique information-carrying capacity. There are guidelines, however, for deciding on which senses to target with display elements, such as provided by Wickens' (1984) proposed structure of processing resources, Multiple Resource Theory. One of the conclusions to be made from this theory is that visual versus auditory modalities use some different processing resources, and this is supportive of the idea of the usefulness of visual-auditory multi-modal displays in particular.

In addition to having high basic channel capacity and dedicated processing resources in the brain, hearing is also, as a practical matter, easier to feed information than are the other senses. Hearing involves a transmission medium, air, which is omnipresent around humans, meaning that sound generation equipment need not be co-located with the listener. This property is also true of smell, but hearing is carried much more quickly and degrades less rapidly and more predictably with distance. Some touch interfaces achieve spacing using jets of air, but the transmission occurs over only very small distances. Thermoreception, like vision, does not require a medium at all, but its already low channel capacity would be lowered further with distance without some way to transmit heat energy with high focus. Touch, proprioception, and taste require contact, which presents engineering challenges, especially for mobile applications.

Practicality related to the generation of sound for displays is another strong point. Sound generation equipment (players of recorded sound, synthesizers, speakers) is extremely widespread due primarily to its use for the reproduction of human speech and music, and human skill and equipment for generating new sounds (play musical instruments and use recording devices, program computers or synthesizers) is also common. Ability to generate signals detectable by the senses other than hearing and sight is more rare. Furthermore, while sound generation systems, once built, require only a modest power supply and little need for upkeep, displays engaging other senses (again, excluding vision) would generate higher continuing costs for both operation and maintenance. For engaging proprioception, viz. for full-motion flight simulators and the like, much higher power supply as well as mechanical maintenance requirements exist as these systems call for the manipulation of a large mass (the human body and any interface elements that must not move relative to the user). For the olfactory and gustatory senses, ongoing requirements would be either a continuing supply of the chemically complex substances involved in smell and taste or a supply of simpler materials with which these substances could be created on demand.

A final property of hearing making it a good complement to vision in a multi-modal display is that a human need not be in any particular orientation to receive sound signals. Sound is less directional than light, which, in addition to making the positioning of sound sources even more flexible than those of light, also suits auditory displays for tasks of continuous monitoring (Kramer et al., 2010). The use of multiple visual elements in a display almost always involves shifting gaze between them, reducing vision of any elements outside the central, high-resolution useful field of view (UFOV) of the eye, and completely eliminating vision of any elements outside the visual periphery. Hearing perception of sound, like that of light or any other, can be modulated by changes in attention, but barring noise, hearing of any auditory interface is always unbroken.

2 Auditory and Multi-Modal Displays

The most basic use of the audio modality in a display is an alert or alarm. An *alert* consists of an audio signal of pre-determined length triggered by a specific kind of change in a measured variable or combination of variables of the system. An alert is typically a short sound generated at a point in time after which any reaction by the receiver will take place (Spain & Bliss, 2008). One example of an alert is a sound recording playing upon the receipt, by a computer, of an asynchronous message such as an e-mail. An *alarm* consists of an audio signal also initiated by a change in some system variable or variables, but which persists until the reversal of this change (or until it is deliberately silenced) (Bliss & Gilson, 1998). In this case, any reaction on the part of the receiver is likely to be necessary while the alarm continues to sound. In fact, silencing of the alarm can be a second cue to the receiver, often meaning that the conditions for generating the alarm are no longer present. One example of an alarm is the tone emitted by some passenger vehicles when the ignition key is in place and the driver’s door is open. By design, the alarm will persist until one of these conditions is no longer satisfied.

Alerts and alarms, as defined above, both convey only binary information, manifesting one of only two values or sets of values for the system variables involved. Multiple alerts and alarms (or multiple sets of triggering conditions for an alert or alarm, which essentially constitute multiple alerts or alarms), such as individual alarms for patient heart rate being above an upper safety limit and below a lower safety limit, can be used together in the display component for a single variable or set of variables to overcome limitation of binary display component values, but this strategy has drawbacks. One drawback is the possible increase in the negative impact of false positives. With more alarms or alerts, there are more possible types of false positives (one for each alarm-triggering condition or condition set), and it may require more effort to determine whether a false positive has indeed been encountered. Also, multiple alarms or alerts could be falsely triggered at once. (If the display allows no more than one sound to be present at a time, this is something closer to a sonification, though a true sonification would *always* have one sound present.) The extra susceptibility of such displays to false positives is especially problematic with alarms (vice alerts) as they typically demand rapid reaction. The same argument applies where alarms are not actually false positives but happen not to be beneficial at that particular time, again because alarms compel swift reaction. Dynamically adjusting threshold values for triggering alarms, however, has proven useful for ameliorating this latter situation for complex systems and their auditory displays (Otero, Félix, Barro, & Palacios, 2009).

Another problem arising with the use of multiple alerts or alarms is that having a listener properly associate these with what they represent becomes more difficult as the number grows. A strategy for countering this problem for sets of alarms is to design them to be very heterogeneous, as this strategy increases not only discriminability but also learnability (Edworthy, Hellier, Titchener, Naweed, & Roels, 2011). On the other hand, when multiple alerts or alarms are used, especially when applying the design precept of high heterogeneity, a perceived ordinal relationship between them may be difficult to produce, and they may thus be particularly ill-suited to conveying system states that lie on a continuum.

“Sonification” is a term for conveying information using non-speech sound, where datum relations are reflected in perceived relations in an acoustic signal (Kramer et al., 2010). The requirement for perceived relations in the acoustic signal suggests more than two levels being sonified (as in alerts and alarms), such that a continuum can be formed. The Geiger counter, which conveys information about radiation in-situ using a simple clicking sound repeated with varying temporal spacing, is commonly given as an example of a successful application of sonification.

With sonification, at least one auditory dimension of a base sound is manipulated, where an auditory dimension is defined as the perceptual product of a physical characteristic of the sound (Hermann & Hunt, 2011). The main perceptual attributes of a sound are generally acknowledged to be pitch (related to frequency), loudness (related to amplitude or intensity), sound localization (related to source direction and distance), and timbre. Timbre is a catch-all term with a negative definition, according to which it serves as a label for the aggregate of all perceptual attributes not otherwise defined (Walker & Kramer, 2004). Some research has drawn, however, on larger inventories of attributes, including “brightness”, “resonance”, “reverberation”, “vibrato”, “tremolo”, and a different case of “timbre”, again defined as the sum of all attributes not otherwise specified (Watson & Sanderson, 2007).

Auditory dimensions, with their associated ordinal scales, can be engineered to perceptually match ordinal system parameters underlying them. Such a design, where there is an inherent agreement between the representation and what is represented, is sometimes referred to as *analogical representation* (Barrass & Kramer, 1999). This type of representation, in turn, is set on a classification spectrum opposite *symbolic representation* (which would describe the combinations of alarms mentioned above), over which it has evident advantages such as reduced cognitive load (Kramer, 1994).

There have been successful applications of sonification in multi-modal displays and even in settings with much attentional demand, such as sonifying respiration in a busy medical setting (Watson & Sanderson, 2004). There are also examples of sonifications that are helpful in some respects but harmful in others. Harm may come about in the presence of alerts or alarms, which sonifications may mask, and in the presence of other system elements requiring monitoring and divided attention, away from which sonifications may draw attentional resources (Donmez, Cummings, & Graham, 2009).

The choice of auditory dimensions in sonification designs could be based solely on the sensitivity of the human auditory system to changes in each dimension, as noted in the psychoacoustic literature (e.g., Olson, 1967) where these dimensions are considered in isolation, but doing so would leave out some important practical information. Using multiple dimensions at once will affect the sensitivity to each, for instance, and the number of discriminable steps along the different dimensions must also be considered. Furthermore, psychoacoustic experiments are normally carried out with less distraction and less noise, and using less harmonically complex sounds than would be the case in sonification applications (Anderson &

Sanderson, 2009). All of these facts should also be taken into consideration while designing any sonification.

Pitch is a natural first choice of auditory dimension for sonification, as humans can distinguish very many levels of it, it can be paired with a set high loudness (by setting a high amplitude), and it can be paired with a set highly perceptible timbre (Walker & Kramer, 2004). Using loudness has the weakness of lower levels of it not being usable in the presence of background noise. Using sound localization has a practical drawback rooted in the fact that it partially relies on subconsciously learned effects of the different sound altering properties of the head and ears on sounds arriving from different directions. Because determination of sound source direction relies on processing in the brain which in turn relies on detecting these sound alterations, if sound locale is to be simulated well, the sound must be processed differently for each individual listener in accordance with a profile of her or his head and ear shapes called a Head Related Transfer Function (HRTF) (Wenzel, Arruda, Kistler, & Wightman, 1993). At this time, HRTFs remain costly to produce. Timbre includes some aspects of sound that have been successfully harnessed for sonification, but these aspects are more difficult than pitch to modulate with simple sound production tools.

A sonification conveying even only one system parameter can employ more than one auditory dimension, and such redundancy in coding does confer benefits, but it may cause the individual weaknesses of the different auditory dimensions to apply in new ways. For example, the constraint on the levels of loudness (due to noise) may also constrain pitch. Also, modulation of two dimensions may produce a perceptual mismatch if one dimension appears to vary much more than does the other. Furthermore, there may be interactions between specific sound dimensions. For example, it has been found that pitch and loudness are not orthogonal, but that changing one can affect the other (Neuhoff, Wayand, & Kramer, 2002).

Auditory graphs are the sonification equivalent to visual graphs, and thus are a particular case of sonification in which the sonified data are not real-time but rather recorded (or entirely fabricated) (Davison & Walker, 2007). Findings related to auditory graphs provide useful insights for the design of sonification of real-time data. Research on auditory graphs led to the conclusion that a sonification display as described so far is akin to a visual graph lacking any axes, tick marks, or legends, and suffers for this lack of context (Walker & Nees, 2005).

Performance in judging values in auditory graphs was shown to benefit from a dynamic reference sonification (sonification of a constant) which alternated between representing a particular value in the upper region at any time when the main sonification was increasing, and one in the lower region when it was decreasing (Smith & Walker, 2002). The sonification of a constant must of course match, in the representative dimension, the sonification of that value as a system parameter. It can be made different in other dimensions, however, such as using a different timbre in a sonification based on pitch, or by presenting it only intermittently (amplitude varying with time) so as not to impede perception of the main sonification at all times. The dynamic reference sonification mentioned above could, in fact, be viewed as two reference sonifications presented only intermittently.

Fielding a sonification in an environment with speech communications presents two potential conflicts. There could be competition for processing resources in the brain, and there could be simple masking of the sonification sounds by the speech sounds and vice versa. If the speech communication is between humans, concern over reception of speech (masking and processing) can often be somewhat relaxed. Speakers tend to expect a verbal response no matter what their message, and would therefore notice the lack of one and take appropriate measures. This natural requirement for reception acknowledgement means that the listener can, as far as possible, safely concentrate on the sonification and ignore speech audio. The important aspects of the conflicts mentioned, therefore, can be combatted by specifically testing the impact of *ambient* speech audio on sonification.

There are guidelines for choosing the frequency or frequencies for auditory display for use in the absence of masking sound (e.g., Walker & Kramer, 2004). For contexts with speech sound, sonification frequencies could also be chosen in light of the power spectral density of human speech, perhaps specifically for that as rendered by the voice transmission and sound output system (e.g., noting that a telephone connection can involve a low-pass filter). Doing so, however, would still not account for processing of sound in the brain, and it is known that even unattended speech is automatically processed by the brain and cognitively loads the hearer (Ueda, Nakajima, Doumoto, Ellermeier, & Kattner, 2013). Also, choosing frequencies without regard to the specific application would also not account for the frequency and duration of speech communication episodes varying between domains, nor would it account for the quality of sonification perception required in the task for which the sonification is to be designed. These facts, of course, support application-specific display design and testing.

3 Application Domain

This thesis deals, in general, with the usefulness of various configurations of an auditory display component in the context of a larger, primarily visual, multi-modal display, as well as in the context of a larger stimulus environment that includes ambient speech. The specific application for the display is the task of operation or tele-operation of a land rover over uneven ground. The vehicle parameter chosen to be conveyed by the auditory portion of the display was the tilt angle of the rover away from upright, which has a maximum safe limit according to the design specifications of some vehicles.

The literature review undertaken before devising this study did not reveal prior work with a similar variable (one calling for continuous monitoring) linked to the dynamics of driving and sonified in conjunction with vocal ambient noise. It did, however, reveal a reduction in task completion times associated with the use of a gravity-referenced main visual display rather than a dedicated, supplementary visual display for conveying vehicle attitude (Wang, Lewis, & Hughes, 2004). The benefit of including tilt information in a main visual display rather conveying it via an additional, dedicated visual display may relate to a general unsuitability of discrete display elements for rover tilt information, but a single display may be superior only when the alternative is using a *visual* secondary display, or the advantage may be greater over using a visual secondary display than over using an auditory secondary display.

The design specification of the Lunar Exploration Light Rover (LELR), in development by the Robotics and Space Exploration division of MacDonald Dettwiler and Associates (MDA) Ltd., served as the starting point for the design. It calls for the rover being able to drive across slopes with a tilt (roll) angle of 25 degrees without difficulty (specification [EC-LMR-PRF-110]), and further requires that the rover rollover threshold be at least 36.9 degrees (specification [EC-LMR-PRF-064]). Appendix A contains the full text of these specification sections. For the sake of simplicity, a single threshold of 30 degrees was chosen as the safety limit for the human subject experiment that was conducted as part of this research effort. Details about this experiment are discussed in the upcoming chapters.

Note that the application domain of tele-operation of a *Lunar* rover, specifically, introduces the special interface design problem of considerable control and feedback lag due to the distance of 1.3 light-seconds separating the Earth and its moon. This distance and the time required for electro-magnetic control and feedback signals to cross it adds 2.6 seconds to any lag present in the system for other reasons (control and sensor response lag, and transmitter signal processing lag), which would be the majority of the lag in most systems and could make tele-operation more difficult.

The problem of transmission delay was not modelled in this experiment, however, due to the technical difficulty of doing so along with all of the other system design manipulation required for the experiment, and in an effort to generate results applicable to a much wider range of vehicles than tele-operated Lunar rovers alone. Also, the effects of lag in tele-operated Lunar rovers have been successfully reduced with purpose-built displays, with performance increases of greater than half of the difference between unassisted operation and no-lag operation being achieved (Matheson et al., in press). Any performance effects of the auditory tilt feedback displays under study were expected to apply at least somewhat to situations with lag on the order of seconds, and to be at least somewhat additive with the positive effects of anti-lag displays.

The parameter chosen for the auditory display was not roll angle, but instead tilt angle of the normal of the plane of the vehicle away from vertical in any direction, where vertical is normal to the gravitational field, and where, for example, a vehicle with all tires at the same altitude would have a tilt angle of zero degrees. If only roll angle were conveyed to the driver, a rapid change in direction could result in a rapid change in roll angle (given disparate initial roll and pitch angles). Allowing this kind of rapid change would undermine capability of the display to guard against the driver breaching the safety limit. By using the off-vertical tilt angle (in any direction), which reflects the roll and pitch angles, changes in value will always be as abrupt or as gradual as changes in the angle of the terrain crossed by a vehicle, which constitutes better predictability and possibly reliability of sonification for the driver. Also, any vehicle is likely to have both a maximum safe roll angle and a maximum safe pitch angle (and possibly enumerated limits for combinations of the two at once), thus a safety envelope of angle combinations. This design caters to the simple case where the maximum safe roll and pitch angles are the same, yielding the safety envelope depicted in Figure 3.

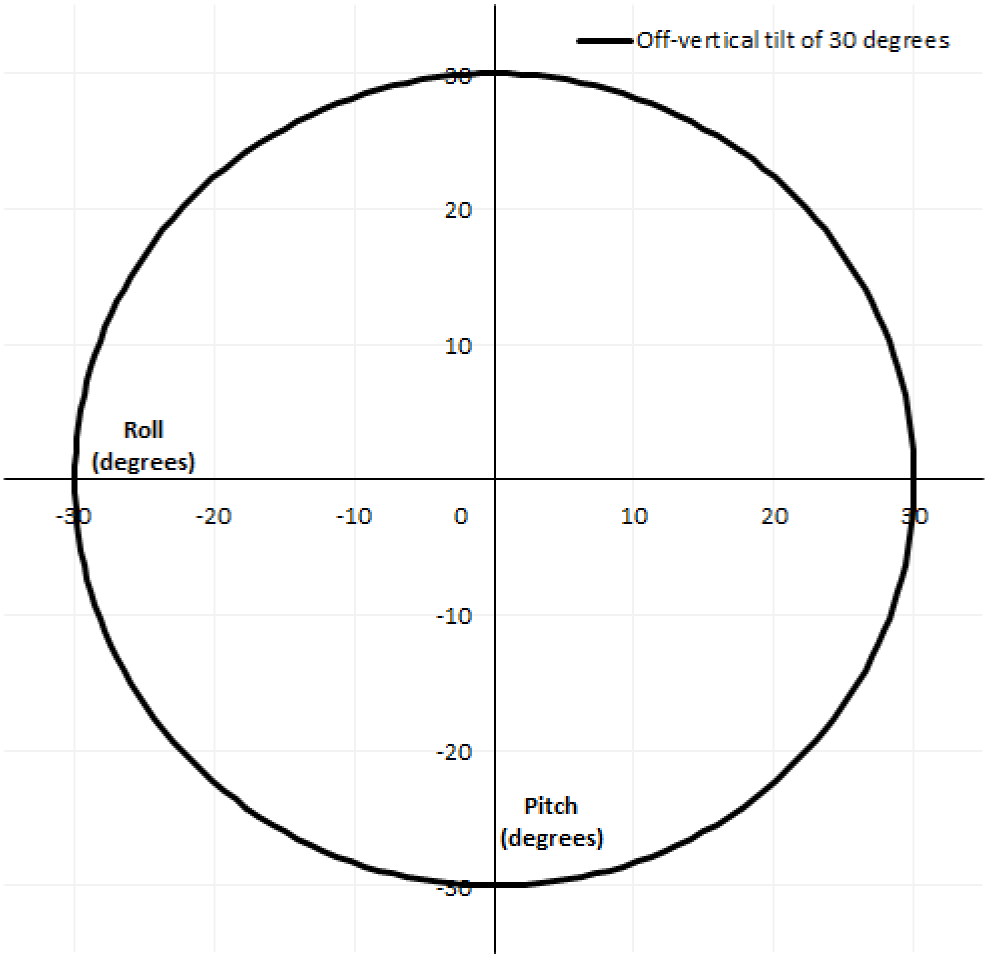


Figure 3 – Vehicle roll-pitch safety envelope

Chapter 3
Display Design

4 Vehicle Simulator

The driving task was realized using a simulator for tele-operation of the CBRN Crime Scene Modeller (C2SM), a sensor platform developed by MDA Ltd. to help crime scene investigators identify and locate chemical, biological, radiological, nuclear, and explosive (CBRNE) threats before introducing humans into a potentially unsafe area. The vehicle simulated was a slow-moving tracked vehicle with the ability to turn in place, which provided extra impetus to give feedback on overall tilt angle rather than simply roll angle.

The C2SM simulator was set to output telemetry including position and orientation in three dimensions, with orientation data included explicit roll and pitch angles sent to the sonification system approximately every 0.25 s.

5 Task

The driving simulator generates auditory alerts for some setting changes, but does not generate any auditory feedback while the driver is directing the vehicle to move. Driving is completely based on visual communication from machine to human, and adding an auditory display made it a multi-modal display. Though vehicle tilt could be ascertained, to some degree, from the visual interface by observing the orientation of the horizon, the angle would be difficult to judge with precision by this method. Moreover the horizon is not always visible, and what ground-sky intersection is visible is not always horizontal. The vehicle would have to be on a featureless plane or on the surface of a featureless sphere in order for the horizon to be a dependable indication of tilt. This was not true in the simulation and is extremely unlikely to be true in a real application, so the only representation of tilt angle to be at all reliable would be the auditory display, and redundancy with the visual display was not a goal of implementing the auditory display. As mentioned above, the fact that hearing processing by humans has some resources not in common with those of vision processing is one reason why presenting information about different system parameters to different senses is potentially advantageous.

To accentuate the importance of the tilt angle without requiring explicit subject responses on judgement of their perceptions of the angle, a driving task was devised in which subjects were incentivized to drive near the tilt angle safety limit but never cross it. Compelling operation near the limit promoted a focus on the precision of the display. It also promoted continuous monitoring, and this was for only a few minutes at a time, during which fatigue was expected not to play a large role. Sonification engages pre-attentive resources to some extent, but having the monitoring taking place in short bursts with strong motivation was expected to reduce concerns about inattention still further, and thus reduce related concerns about display design (possibly inviting inattention through poor signal salience) (Watson, Sanderson, & Russell, 2004).

The task was to drive the rover along a course in which operating near the tilt safety limit (but, again, not exceeding it) was a clearly preferable strategy. Drivers were encouraged to complete the course in as little time as possible, and the terrain and waypoints were chosen such that routes involving roll angles approaching the threshold would be the most time-efficient. The key terrain design element was an obstacle with a slope gradually increasing from zero at its periphery to in excess of 30 degrees at its centre. A terrain rise (a gamut of positive slope angles) of this description would resemble a volcano; a terrain depression (a gamut of negative slope angles) of this kind would resemble a whirlpool. With such an obstacle in the path of a driver, being able to operate with higher tilt angles would enable taking routes closer to the terrain feature's centre and thus shorter and quicker routes.

An operator having better knowledge (quicker, more precise) of the rover's tilt angle means that less safety margin can be used, which amounts to a higher maximum rover tilt angle. Figure 4 is an illustration of the concept showing different routes that could be taken around such terrain features with negative and positive slopes given different maximum rover tilt angles (or better tilt information available to the operator). In the scenario on the left, the rover has a low maximum tilt angle, and is not allowed to exceed the angle represented by the terrain slope contour lines forming medium-sized ellipses, which results in the shortest path from point A to point B being the one shown. In the scenario on the right, the rover has a higher maximum tilt angle, matching that of the smaller ellipses, and a shorter shortest path from A to B.

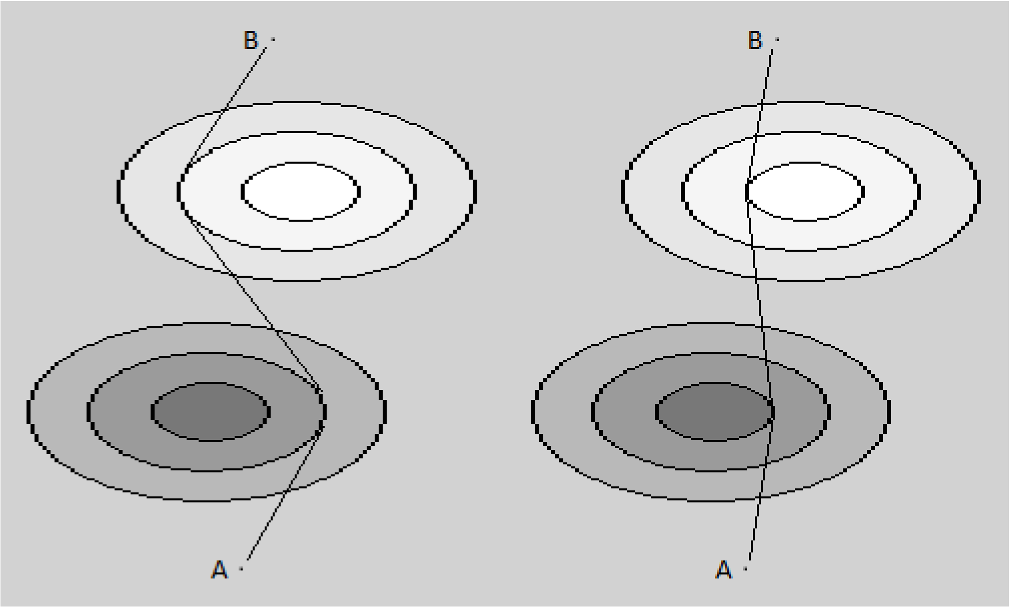


Figure 4 – Constant-slope contour line maps of shortest routes around depression and rise
(Left: Lower maximum tilt angle, longer route from A to B)
(Right: Higher maximum tilt angle, shorter route from A to B)

The experiment used a single design of variable-slope terrain feature with positive slopes. Figure 5 is an orthographic oblique view of the 3-D model of the terrain feature created in Blender 2.66a and used in this experiment. Figure 6 is an orthographic side view of the same feature, demonstrating the relationship between slope and distance from terrain feature centre. The slope increased from two to 46 degrees in discrete steps of two degrees, with these steps equally spaced horizontally along the span from terrain feature edge to its centre. Higher-resolution changes in slope were explored, with the goal of reducing the coarseness of changes in tilt angle resulting from driving over the polygon edges of the 3-D model (intersections between faces, visible in Figure 5). Terrain feature 3-D models with the attendant higher polygon counts could not be used, however, due to limitations in computer processing for the simulator.

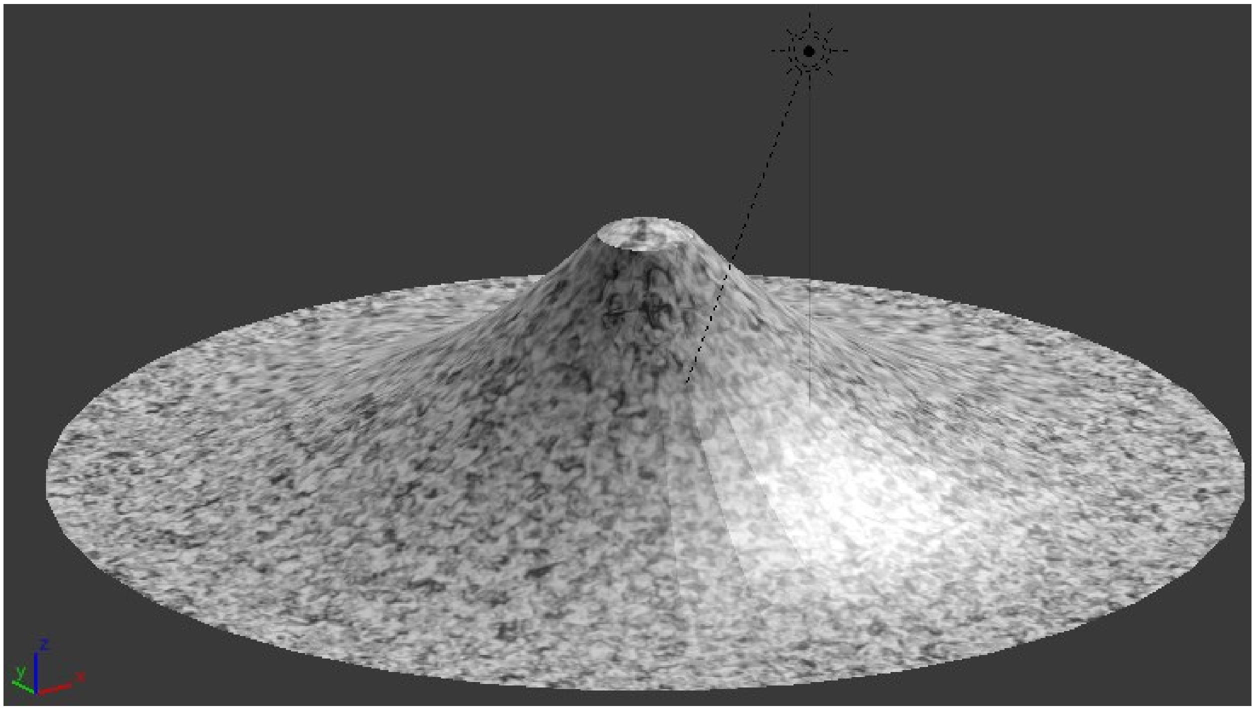


Figure 5 – 3-D model of terrain feature used to encourage driving near tilt safety limit (dashed line unrelated, simply light source axis in 3-D modelling program)

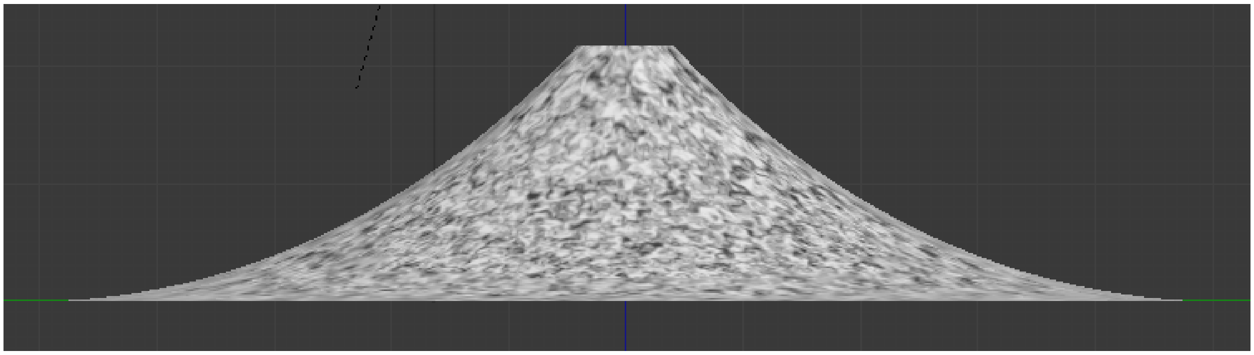


Figure 6 – Side view of terrain feature

The course used in the experiment involved a plane, a single instance of the variable-slope terrain feature, and a gate a short distance from the outer edge of the terrain feature marking both the start and finish lines. To complete the course, drivers had to do a single counter-clockwise lap around the terrain feature. Figure 7 depicts a nearly ideal path.

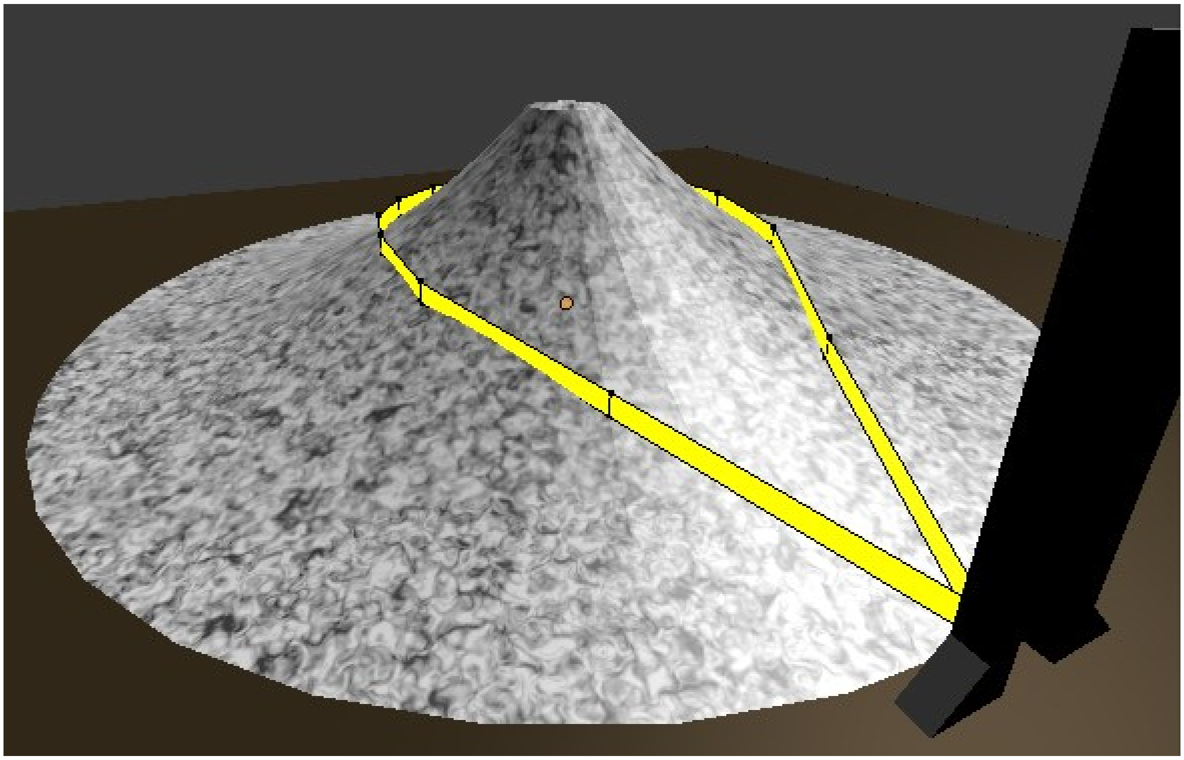


Figure 7 – Task course setting with nearly ideal route shown as yellow ribbon (orange dot unrelated, simply 3-D object anchor point in modelling program)

Ideally subjects would not have traded off accuracy for speed to an extent great enough to mean any of them would cross the roll angle danger threshold, but to make this outcome less likely, a time penalty for operation beyond the threshold was communicated to subjects. Such a penalty being made known to subjects was expected to prevent them from crossing the roll threshold accidentally (due to speed-accuracy trade-off) or deliberately ('cutting corners'). It was expected to allow more meaningful comparisons of task performance scores between participants, as it was expected to reduce the variance in speed-accuracy trade-off decisions between subjects. No deliberate incentives (e.g., monetary) were tied to performance.

The penalty constructed was for time spans spent in excess of the tilt safety limit to be given extra weight in the calculation of the total time span required to complete the course, and a specific penalty weight factor of three (time spans above the safety limit counting triply toward course completion time) was chosen. This factor was expected to be high enough to discourage participants from paying too little attention to accuracy or from flouting the danger threshold and taking unacceptable routes, but also low enough not to penalize too harshly short, inadvertent excursions above the danger threshold.

Ideally raw course completion time spans and time spans in excess of the tilt angle safety limit were to form the basis of results investigation. In that case, the penalty arrangement would simply have served as a tool to moderate the actions of the study participants. Course completion time spans adjusted as per the penalty scheme communicated to the subjects was also seen as a possible outcome variable for results investigation, especially if there was large variance in speed-accuracy trade-off strategies.

6 Environment (Masking by Voice Communication)

This experiment was devised to test the suitability of various designs of auditory display for the application domain as described above, but consideration of sound masking by human voice communication was added. This consideration of voice communication was deemed important based on the supposition that any vehicle operated (especially tele-operated) over uneven ground with close attention paid to a specific tilt safety angle is likely to be part of a team effort, and such efforts are likely to involve human voice communication in some respect. The focus was on the impact of voice communication on perception of the auditory display, and not the reverse, so the auditory displays tested were to be considered suitable if they positively impacted task performance in the presence of speech, and even more so if they promoted performance equal to that in the absence of speech. A review of the literature did not reveal any relevant research on this topic.

Some experimental conditions included background noise in the form of radio chatter between aircraft pilots and air traffic controllers in an aerodrome tower. The radio chatter was seen as a good proxy for the kind of structured voice communication that might be present in the application domain. The characteristic quality of voice transmitted over aviation radio channels was also seen as a potential match with the application domain, where operators might be connected via radio relay or an intercom with similar fidelity. The experiment was meant to judge only the impact of sound masking and any unavoidable processing, in the brain, of speech, rather than any further processing related to deliberate attempts to understand the voice communication. Accordingly there was no requirement for subjects to pay attention to the speech.

7 Auditory Display

The auditory display system was a pair of headphones fed by a purpose-built Java program acting on telemetry information received from the simulator. The program also gathered the performance measurements for the experiment. The program consisted of two class files, the source code of which appear in Appendix B.

10 variations of auditory display design were implemented in the program and eight of these were used in the experiment, all of which included an alarm triggered by the tilt angle being in excess of the safety limit (and remaining activated as long as this condition persisted). Most designs also included sonification of the tilt angle with the frequency of a pure tone. Two different versions of this sonification, with different transfer functions between values of tilt angle and values of frequency, were created. A (reference) sonification of the tilt angle safety value was also developed.

More details on these different display elements and on the overall experimental conditions are discussed in the following sections, but the auditory display elements just described were combined into five distinct arrangements and thus five auditory display conditions: alarm-only, alarm plus sonification of tilt angle with a linear relationship between pitch and tilt angle, alarm plus sonification with a quadratic relationship between pitch and tilt angle, and each of the two conditions just described with the additional sonification of the tilt angle safety limit constant (reference sonification).

7.1 Auditory Display Input

As discussed previously, the system input parameter for the auditory display was the off-vertical tilt angle of the rover. Tilt angle was calculated using the angle summation formula for orthogonal angle pairs (Eq. 1, given subscripts for this particular use) acting on the roll angle and tilt angle telemetry output by the simulator.

Eq 1. $$\Theta_{tilt} = \operatorname{acos}(\cos(\Theta_{roll}) \cdot \cos(\Theta_{pitch}))$$

The tilt angle of the rover is the angle around whichever axis on the x-y (horizontal) plane would yield the measurement of the greatest off-vertical angle. Otherwise stated, it gives the angle around whichever axis can (at that time) fully capture the off-vertical tilt as a single rotation.

7.2 Auditory Display Output

All conditions of the experiment included an alarm. This alarm was designed to be shrill and eminently noticeable in all conditions of the experiment, providing unambiguous direction to drivers to correct the tilt angle immediately once it had exceeded the safety limit. Giving drivers feedback at tilt angles below the safety limit was left solely to the purview of the other auditory display elements.

Since vehicle tilt angle values form an ordinal scale (actually a continuous ratio variable), sonification was seen as the proper way to represent values of it below the safety limit. Representation with a series of distinct alerts or alarms for discrete values of tilt angle would have been less suitable. For ease of implementation of the many display elements required for the experiment, the basic sound chosen was a pure sinusoidal tone. Use of pure tones is common in psychophysical research (e.g., Dai & Micheyl, 2011) but less so in sonification research (Kramer et al., 2010).

Also for ease of implementation, and following successful past sonifications (e.g., Watson & Sanderson, 2004), pitch was chosen as the output auditory dimension, and thus frequency was the sound property modulated.

Modulation of loudness was also implemented in some early, exploratory versions of the display, with higher loudness indicating higher tilt angle, but this approach was abandoned. Changes of loudness could inherently interfere with frequency perception, as noted previously, and modulating loudness raised a design dilemma. There was a question of whether sonification of a reference value should be accorded the loudness level corresponding to that value in the main sonification (i.e., should be fixed at that of the real-time system parameter for that value), or whether the loudness of the reference value should be made to always match that of the main sonification (i.e., should be variable). Using a fixed loudness level avoided the need to make a decision between these options.

Elaborating further on the design dilemma, if the reference sonification were to *hold* at the high loudness, there could be drawbacks to the reference sonification being almost always louder than the main sonification. Also, it would mean that the reference sonification would serve as a reference of the frequency but not the loudness corresponding to the safety limit as represented by the datum sonification. If the reference sonification were to *vary* in loudness, it would mean that the reference sonification would not always be present (since loudness would drop to zero for low or zero values of the real-time variable, unless loudness were made to vary over less than its full range), potentially being missing for long stretches of time, which would undermine its role as a reference. None of these points prove that the sonification would be less successful if it incorporated loudness, but judgement of loudness, whether absolute (based on memory of previous values) or relative (with the presence of multiple values), was not expected to make a worthwhile contribution to the effectiveness of the display.

The first design decision about the sonification frequencies was that the range would be situated within a single octave. The main reason to stay within one octave was that the concept of pitch contains two perceptual attributes: the monotonic dimension *pitch height* and the circular dimension *pitch class* or *pitch chroma* (Deutsch, Dooley, & Henthorn, 2008). Pitch height is the basic perception of frequency, whereas pitch chroma relates to detection of harmonic relationship between frequencies (as in a musical scale), with results such as a perceived sameness of sounds exactly one octave apart (Schomaker et al., 1995). Spreading the sonification frequency range across more than one octave could increase discriminability of pitch height, but at the expense of ambiguous coding in pitch chroma.

Given a set frequency span (expressed as a ratio, a one-octave span), the next design decision with respect to sonification frequencies was the choice of specific frequency range. The frequency range of human audition is widely taken to be 20 to 20 000 Hz, but auditory display designers usually restrict displays to the range of 200 to 5 000 Hz, as this is the region with best human sensitivity to pitch (Walker & Kramer, 2004). A consideration especial to sonification within a single octave is that pitch chroma is a perceptual attribute of sound only in the range of 50 to 4 000 Hz (Schomaker et al., 1995). The frequency range selected for sonification output was 440 to 880 Hz, which is near the centre of the usual range for auditory displays and the centre of the range in which pitch chroma is detectable (with these ranges viewed on a logarithmic scale, in line with human perception of pitch) as well as being bounded by well-known specific musical frequencies known as A4 (or A440) and A5 (or A880).

Another advantage of the frequency range selected was that in this range loudness could be expected to be relatively constant throughout. Human perception of sound intensity (loudness) varies with frequency, and the design decision to use a single octave rather than a wider pitch range as the sonification basis kept this undesirable variation in loudness low. Furthermore, the dependence of loudness on frequency is complex, with less correlation in some frequency ranges than in others, and the choice of sonification frequency range was nearly optimal for minimizing this dependence.

In general, sound at a lower frequency requires a higher sound pressure level (SPL) in order to be perceived at a given loudness, and the differences in required SPL are greater among low frequencies than among higher frequencies. The importance of SPL in determining loudness decreases for increasing frequency up to roughly 1 000 Hz, above which the importance of SPL rises and falls twice with rising frequency. Looking at pitch and the variable influence of SPL through the entire range of human hearing, the octave in which loudness depends least on frequency is roughly 500 to 1 000 Hz. This range can be identified from *equal-loudness curves*, also known as Fletcher-Munson curves, which depict the relationships between particular loudness levels for humans and their causative combinations of sound frequency and SPL (Hermann & Hunt, 2011).

Some sound production devices may modify output amplitude according to frequency in an effort to normalize loudness across large frequency ranges, but a small frequency range guards against any remaining arbitrary variation in loudness, and selection of a frequency range on a fairly horizontal part of the equal-loudness curve guards against loudness variation even further. It was seen as an important design feature of the sonification for it to have nearly constant loudness naturally, given that testing the effectiveness of sonification when masked by other sounds (speech) was part of the experiment.

The alarm frequency or frequencies were chosen by similar reasoning as those for sonification, with some added constraints. The alarm was designed to be clearly more urgent than sonification output. In accordance with research on promoting perception of urgency, the alarm incorporated multiple frequencies, higher frequencies, a rapid pulsing character, no inter-pulse interval, higher intensity, and (relying on variation of loudness with pitch) variable intensity (Giang et al., 2010). Spectrally the alarm was a sawtooth pattern with continuous linear sweep from 1 584 to 1 936 Hz. These sweeps were without inter-pulse interval, as noted, and occurred with a frequency of 4 Hz.

A sonification implementation using the MIDI (Musical Instrument Digital Interface) protocol was explored, as production of sound using MIDI is convenient, requiring only that an instrument and musical note be requested to generate a sound. However, the protocol only yields frequencies other than those of notes in the musical scale with some difficulty, via a process known as pitch-bending (Moussa, 2006). This process, in turn, is based on pitch adjustment amounts rooted in the musical scale, and does not allow for frequencies to be specified exactly without fairly complex calculations. Moreover, the smallest frequency changes producible by the MIDI protocol, assuming use of the equal temperance scale (standard notes, which are equally spaced on a logarithmic scale), were determined not to be small enough. PCM (Pulse-Code Modulation) sound representation, which describes sound through a very rapid succession of recorded or fabricated sound pressure levels (amplitude readings), was used instead (Oliver, Pierce, & Shannon, 1948). Even though PCM is more difficult to use for generating new sounds, there are some programming tools to facilitate doing so, one of which is the JSyn library for the Java programming language (Burk, 1998), and the particular problem of selecting frequencies outside the musical scale can be easier with PCM than with MIDI.

Humans are known to be able to distinguish between frequencies 3.6 Hz apart within the octave of 1,000 to 2,000 Hz (Olson, 1967). Minimum distinguishable frequency difference decreases along a logarithmic scale along with the definition of an octave (i.e., it is proportionately lower for the octave chosen for the sonification), so the comparison with MIDI output precision can be made in this range. One octave on an equal-temperament scale contains 12 semi-tones (13 if counting both ends), which are frequencies equally spaced on a logarithmic scale (Loy, 2007). The formula for the frequency of the different semitones is thus Eq. 2, with integer i in range [0,12] (inclusive) corresponding to the semitones:

Eq 2.
$$f = (1\ 000\ \text{Hz}) \cdot 2^{i/12}, \quad i \in [0,12]$$

The smallest spacing is between the semitones corresponding to $i = 0$ and $i = 1$. These semitones are $f_0 = 1\ 000$ Hz and $f_1 \approx 1\ 059$ Hz, respectively, with a span more than 16 times too great to be perceptually continuous (59 Hz vs 3.6 Hz). There was no specific expectation of the fineness of frequency gradation required in the sonification to enable maximum performance, but it was decided that the chance of it being too coarse should be avoided. The sonification was thus implemented with sound on a continuous frequency scale. This was of course not exactly continuous, as the computer-generated sound was based on digitally quantized and periodically sampled sound, but with 16 bits (2^{16} or 32 768 levels) of quantization for the sound pressure levels and a sampling rate in the tens of kilohertz, the representation of frequencies was fine enough to appear continuous to a human listener (Burk, 1998).

The term "mapping" is used with regard to auditory interfaces to refer to a set of associations between system parameters and auditory dimensions, but here it will be used to refer to a lower level of association: the specific transfer function relating levels of vehicle tilt to levels of sonification frequency. Having set the sonification frequencies for the highest and lowest values of rover tilt angle (terminating at zero degrees and at the safety limit), mapping frequencies to the intermediate values was necessary.

The starting point for sonification of one dimension with another is to design for a perceptually linear increase in the output variable given a linear increase of the input variable. Since there is a logarithmic transformation from frequency of a sound to the pitch (pitch height) perceived by humans, this design would mean a transfer function of Eq. 3, where the system parameter is x, the sonification frequency is f, and z, j, and k are constants.

Eq 3. $$f = j \cdot z^{k \cdot x}$$

Since the frequency range was chosen to be one octave, the top of the normal sonification scale for this experiment had a frequency twice that of the bottom of the scale, and the exponential base z was set to two. According to the design decision to have the angles sonified within the normal sonification range be zero through 30 degrees, k was set to 1/30 and x was changed to Θ, to be expressed in degrees. According to the design decision to set the sonification range to 440 to 880 Hz, j was set to 440 Hz.

Eq. 4 is the equation for the mapping, or the transfer function, which results in a linear relationship between rover tilt angle and pitch. f_{lin} designates the sonification frequency in this mapping, which was one of two mappings used in the experiment and is referred to hereafter as the linear mapping.

Eq 4. $$f_{lin} = (440\ \text{Hz}) \cdot 2^{\Theta/30}$$

Given that 440 Hz is perceived as an A note numbered 4 (‘A4’) and 880 Hz is perceived as an A note numbered 5 (‘A5’), 27.5 Hz must correspond with a numbering of 0, and we can express pitch (p) as a function of frequency using ‘A-number’ units according to Eq. 5. Note that this equation is strictly for pitch height if referring to a frequency range greater than one octave, but for the single-octave range employed in this experiment it also reflects pitch chroma.

Eq 5. $$p = \log_2 (f / 27.5\text{ Hz})$$

Substituting the right hand side of Eq. 4 for f in Eq. 5 and working through Eq. 6, Eq. 7, and Eq. 8, we see that the Eq. 4 mapping does indeed cause pitch to be a linear function of tilt angle.

Eq 6. $$p = \log_2 ([(440\text{ Hz}) \cdot 2^{\Theta/30}] / 27.5\text{ Hz})$$

Eq 7. $$p = \Theta/30 \cdot \log_2 (440\text{ Hz} / 27.5\text{ Hz})$$

Eq 8. $$p = c \cdot \Theta / 7.5$$

For the purposes of the task in this experiment, participant judgement of lower values of rover tilt angle scale were less important for performance of the task than was judgement of higher values, so it was hypothesized that more precision in the perception or judgement of higher values would promote increased task performance. Starting with the first sonification mapping designed but departing from the linear relationship between pitch height and tilt angle in favour of progressively higher display resolution for higher tilt angles, other mappings were explored.

Eq. 9 is the general form of the transfer function arrived at for a mapping which allowed more perceptual variation in pitch for changes between higher values of a system parameter. k^2 is used in place of an unmodified constant to allow direct comparison with Eq. 3. Within the two equations for pitch sonification of the same parameter of the same system for the same frequency range, all of the constant (j, k, z) and variable (f, x) values would be shared.

Eq 9. $$f = j \cdot z^{k^2 \cdot x^2}$$

Eq. 10 is Eq. 9 modified for the proper frequency range for this experiment, and describes a quadratic relationship between rover tilt angle and pitch. f_{quad} designates the sonification frequency in this mapping, which was the other of the two mappings used in this experiment and is referred to hereafter as the quadratic mapping.

Eq 10. $$f_{quad} = (440\text{ Hz}) \cdot 2^{\Theta^2/900}$$

Substituting the right hand side of Eq. 10 (the quadratic transfer function) for *f* in Eq. 5 and working through Eq. 11, Eq. 12, and Eq. 13, we see that the Eq. 4 mapping does indeed cause pitch to be a quadratic function of tilt angle.

Eq 11. $$p_h = \log_2 \left([(440\text{ Hz}) \cdot 2^{\Theta^2/900}] / 27.5\text{ Hz} \right)$$

Eq 12. $$p_h = \Theta^2/900 \cdot \log_2 (440\text{ Hz} / 27.5\text{ Hz})$$

Eq 13. $$p_h = \Theta^2 / 225$$

For the minimum and maximum system parameters, the quadratic mapping yields, of course, the same frequencies as does the linear mapping ($1 \cdot j$ and $2 \cdot j$, respectively). Because Θ is raised to a power greater than one in the quadratic mapping, however, a larger swath of that same allocated frequency range will be used for higher values of rover tilt angle. This discrepancy can be seen by noting that whereas the linear mapping renders the median rover tilt angle of 15 degrees as $\sqrt{2} \cdot j$, which is roughly 41% of the space between minimum and maximum frequencies (leaving 59% of the frequency space for the upper half of rover tilt angles), the quadratic mapping renders a tilt angle of 15 degrees as $\sqrt{4} \cdot j$, which is roughly 19% of the space (leaving 81% of it for the upper half of angles).

Figure 8 illustrates the point about the impact of transfer function on resolution further, showing the relationships between sonification frequency and rover tilt angle for the (perceptually) linear mapping and (perceptually) quadratic mapping from this experiment, as well as a mathematically linear mapping (not perceptually linear; not used in this experiment) for comparison.

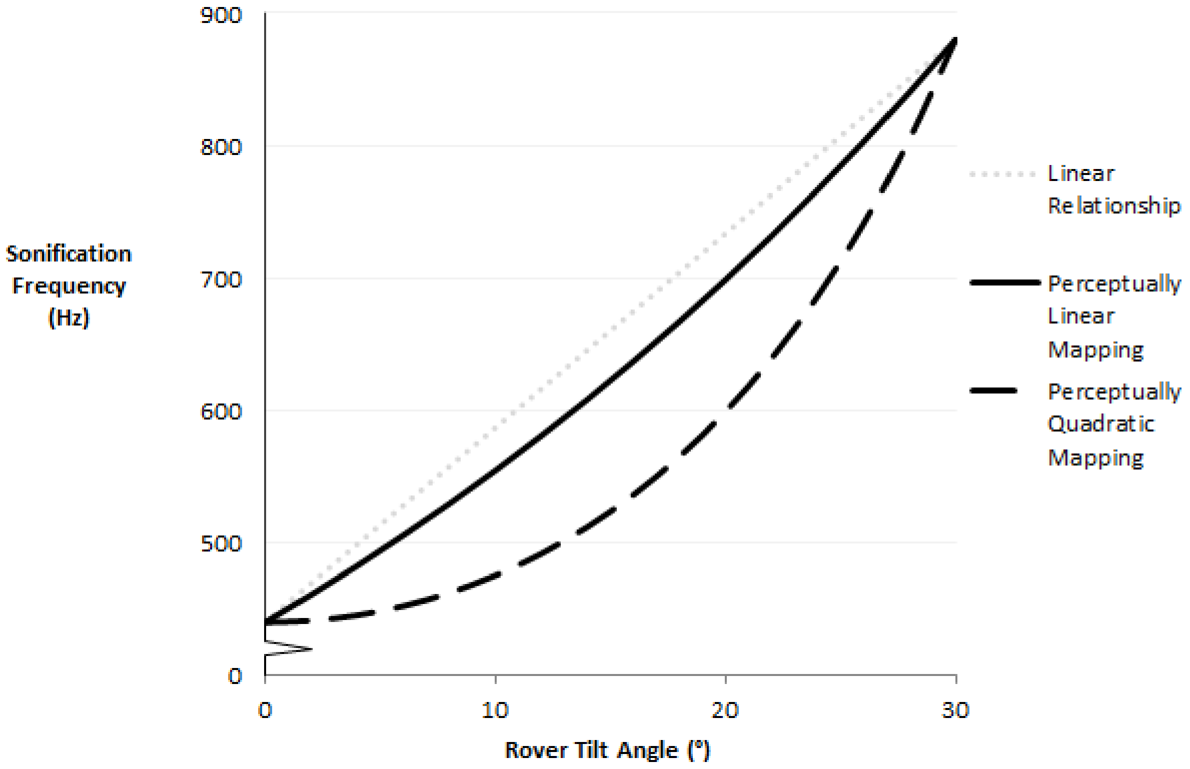


Figure 8 – Transfer functions, or "mappings", used in this experiment (perceptually linear mapping designed to uniformly represent angle differences; perceptually quadratic mapping designed to heighten pitch resolution for high angles)

The differences in the resolution of pitch (the psychological quantity) are less dramatic than the differences in the resolution of frequency (the physical quantity), due to the relationship between frequency and pitch being logarithmic. Figure 9 illustrates the relationships between pitch and rover tilt angle for the linear mapping and the quadratic mapping, where pitch is the pitch that is expected to arise in the operator following from the frequency used, but which is subject to the individual differences in perception on the part of the operator). This mapping visualization directly exhibits the modifications to pitch resolution brought about by the quadratic mapping with respect to the linear mapping.

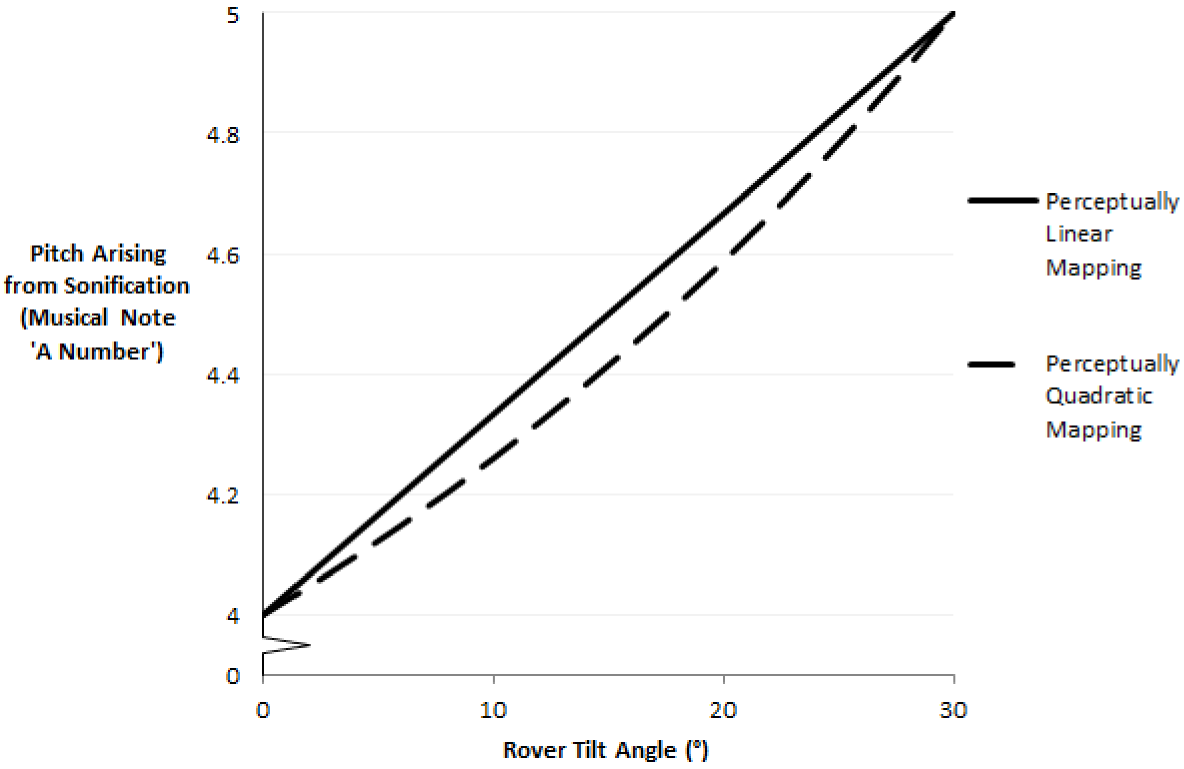


Figure 9 – Relationship, for each mapping, between tilt angle and pitch perceived

In addition to sonification of the real-time data, a sonification of the tilt angle safety limit, which can be called a reference sonification or a reference tone, was also designed. It was designed to use the same output dimension as the datum sonification (sonification of real-time system data), i.e., use a set pitch of 880 Hz, allowing subjects to use relative judgement rather than absolute judgement to determine tilt angle. This sonification, however, was made to differ from sonification of a live 30-degree datum in one respect. The reference sonification involved presenting sound intermittently, half of the time and with a frequency of 0.5 Hz. Thus, each second with the reference tone was followed by a second without. The intermittency was designed to allow subjects to distinguish between the reference and the datum sonifications. The short period of the intermittency was chosen so as to allow very little time to pass during which the comparison aid would not be available.

Chapter 4
Method

8 Manipulated Variables and Hypotheses

This was an experiment to test the effects of different sonification design elements, including sonification of a reference value as well as different mappings between sonification input and output variables, in both the presence and absence of voice communication background noise. The auditory displays involving sonification were tested against the display of a simple alarm.

8.1 Voice Communications versus No Voice Communications

Given that operation of a rover may take place in an environment with interaction of the operator with other humans by voice (in person or by radio, intercom, or other voice link), one goal of this study was to test the suitability of auditory displays such as those with sonification in situations with sound masking by voice communications. Some conditions in the experiment thus involved ambient voice communications, while others did not.

8.1.1 Hypothesis 1

A hypothesis related to this experimental manipulation was that there would be degraded task performance in the conditions with voice communications versus those without, both overall and for individual auditory display designs or types of designs. This hypothesis was based on the expectation that sound masking of the auditory displays will compromise subjects' perception of them.

8.2 Sonification and Alarm versus Alarm only

Another goal of this experiment was to assess the impact of sonification on performance at a manual control task rewarding operation approaching but not crossing a limit. The comparison was done with respect to performance with an auditory alarm rather than with no auditory display at all. There were two reasons for this factor level choice: the simulator used in the experiment does not include any reliable or precise visual indication of rover tilt angle, and it is reasonable to assume that relevant systems would, by default, employ an auditory alarm, as it is the bare minimum auditory display that could be used to convey information about tilt angle. All conditions of this experiment thus included an alarm (at the rover tilt safety limit), while some also included sonification of rover tilt data.

8.2.1 Hypothesis 2

A hypothesis related to this experimental manipulation was that there would be increased task performance in the conditions with sonification versus those with an auditory alarm only. This hypothesis was based on the simple premise of there having been previous instances of successful application of sonification, including, specifically, modulation of frequency in displays for aiding complex tasks (e.g., Watson & Sanderson, 2004).

8.2.2 Hypothesis 3

A hypothesis related to both this manipulation and that of the presence of background voice communication was that within those conditions with voice communication exposure, sonification would (in spite the background noise) provide a benefit to task performance over the use of an auditory alarm only. This hypothesis was predicated on the previous hypothesis being true, and was further based on the simple fact that any sound masking of the auditory displays will not be complete (*some* portion of each display element, and therefore some portion of any sonification, would come through in any case), meaning that any positive effect of sonification in this case could be reduced but should not be completely eliminated by noise.

8.3 Reference Value and Data versus Data-Only Sonification

Another goal of this experiment was to test whether sonification of a safety limit important to the task at hand promotes better task performance. All experimental conditions involving the reference sonification included sonification of the main data stream (rover tilt angle) of course, since a reference sonification would have no use otherwise. The experiment thus included some conditions with a sonified reference value and sonified data, and some with sonification of data alone. Previous research on sonification of even fairly arbitrary variable values (such as recent variable minima and maxima) showed increased performance, so it was expected that reference sonification of a key value (tilt angle safety limit) would provide the same benefit if not more (Smith & Walker, 2002).

8.3.1 Hypothesis 4

A hypothesis related to this experimental manipulation was that performance will be better with a sonified reference value. This hypothesis was based on evidence that humans perform better when comparing a value against another immediately observable value rather than against a memorized value (relative judgement versus absolute judgement) (Wickens & Hollands, 1999).

8.4 Perceptually Linear versus Quadratic Sonification Mapping

An apparently novel concept in this experiment is the sonification mapping involving deliberate departure from a very compatible linear perceptual relationship between sonification input and output values. The quadratic mapping was created in light of the design choice to limit sonification to a narrow output dimension range, and because of the larger application domain property of one end of the range of sonified system parameter values being far more important than the other. It was a goal of this experiment to test whether, in this particular context, a perceptually quadratic sonification mapping, designed to increase display resolution for more important system parameter values, would promote better task performance than would a perceptually linear mapping. The experiment thus included some conditions with each of these two mappings.

8.4.1 Hypothesis 5

A hypothesis related to this experimental manipulation was that the perceptually quadratic mapping would promote better performance than would the perceptually linear mapping.

9 Experimental Conditions – Summary of Soundscapes

The combination of a particular auditory display design with a particular level of experimentally manipulated sound environment (voice or no voice) is hereafter referred to as a *soundscape*. Table 1 shows the full list of possible soundscapes and thus the full list of possible experimental conditions.

Table 1 – All soundscapes – List of all potential experimental conditions

Condition	Voice	Auditory Feedback	Values Sonified	Datum Sonification Frequency Mapping
1	voice	alarm	-	-
2	voice	alarm, sonification	data	linear
3	voice	alarm, sonification	data	quadratic
4	voice	alarm, sonification	data, reference	linear
5	voice	alarm, sonification	data, reference	quadratic
6	no voice	alarm	-	-
7	no voice	alarm, sonification	data	linear
8	no voice	alarm, sonification	data	quadratic
9	no voice	alarm, sonification	data, reference	linear
10	no voice	alarm, sonification	data, reference	quadratic

In the interest of practicality in the recruitment of subjects for and in the running of the experiment, as well as to facilitate counterbalancing of subjects to counter learning effects, not all soundscapes were used. Table 2 shows the list of all conditions of soundscape, with those not used in the experiment shown on a grey background.

Table 2 – Experiment soundscapes – List of experimental conditions (white background)

Condition	Voice	Auditory Feedback	Values Sonified	Datum Sonification Frequency Mapping
1	voice	alarm	-	-
2	voice	alarm, sonification	data	linear
3	voice	alarm, sonification	data	quadratic
4	voice	alarm, sonification	data, reference	linear
5	voice	alarm, sonification	data, reference	quadratic
6	no voice	alarm	-	-
7	no voice	alarm, sonification	data	linear
8	no voice	alarm, sonification	data	quadratic
9	no voice	alarm, sonification	data, reference	linear
10	no voice	alarm, sonification	data, reference	quadratic

10 Condition Ordering

Counterbalanced ordering of conditions was employed in order to reduce the impact of learning effects on experimental results. There were many experimental conditions, and employing the number of participants necessary for a fully counterbalanced set of sessions was impractical, so incomplete counterbalancing was employed to address first-order learning effects. A balanced Latin square was used to determine the condition sequences to be used for the first half of the experimental subjects. A modified version of the first balanced Latin square was used for the remainder of the subjects. Appendix C contains the balanced Latin squares determining the experimental condition sequence for the study subjects.

11 Experimental Subjects and Procedure

This was a within-subject experiment with 16 University of Toronto students from the Department of Mechanical and Industrial Engineering. There were 5 females and 11 males, and their mean age was 26.4, with a standard deviation of 3.7. The study was approved by the University of Toronto Research Ethics Board.

Subjects were given an explanation of the experiment as a test of a feedback system for driving a rover, and were then asked to fill out a background questionnaire. The subjects are not known to have had any experience with the type of audio interface used in the experiment, nor any hearing impairments. Appendix D contains this questionnaire as administered during the experiment. It also includes an extra question (described below, within this section) which was administered several days after the experiment.

Each subject was given an explanation of the rover as a relatively fragile vehicle necessitating a limit on tilt angle. Each subject was also given an explanation of the simulator controls, which involved the arrow keys of the laptop on which the simulator and sonification programs ran, and was then given a practice run of the full extent of the course with the alarm-only soundscape (Experimental Condition 6). During this practice run each subject was directed to drive high enough up the terrain feature so as to trigger the alarm, in order to familiarize herself or himself with its sound. Figure 10 shows the visual interface of the C2SM simulator. Figure 11 shows the wider experimental setting.

Figure 10 – C2SM sensor platform simulator with experiment course loaded

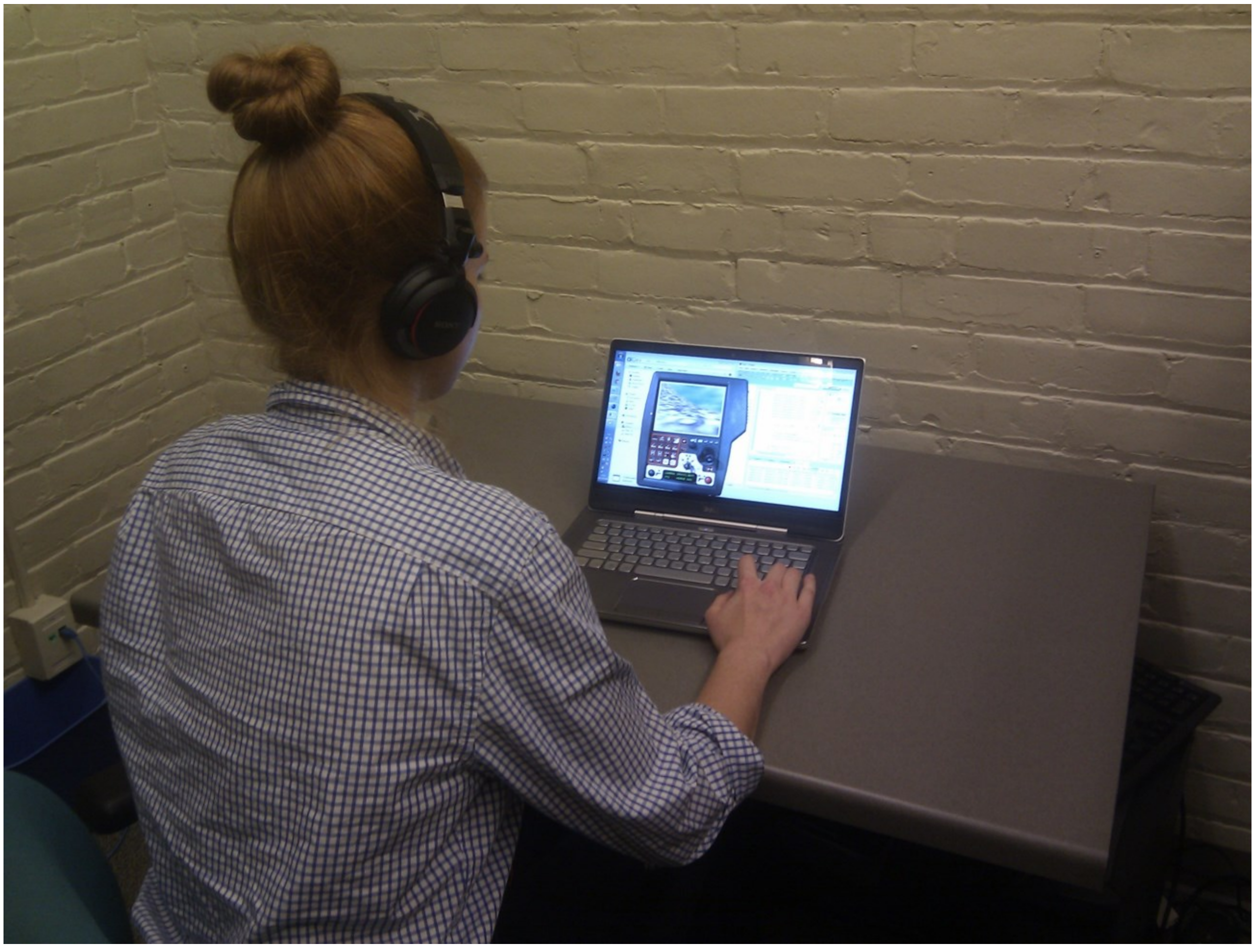


Figure 11 – Experimental subject driving through course with auditory display feedback

After the practice run and before the first experimental condition, each subject was given instructions on filling out two feedback questionnaires. One questionnaire was a modified version of the NASA Task Load Index (NASA TLX), which focusses on difficulties posed by the task (Hart & Staveland, 1988). The other was the System Usability Scale (SUS), which focusses on assessment of the user interface (Brooke, 1996). Appendix D contains copies of these questionnaires as administered. Participants completed these questionnaires after each condition.

Before each experimental condition, each subject received a brief describing the soundscape to which she or he was about to be exposed. Then the simulator and sonification program were started. The alarm was artificially triggered at the beginning of each run as a reminder of the sound (and as a verification that it was enabled), and all other soundscape elements were immediately and consistently audible. As soon as the subject was ready, she or he began driving. A single run of the course took approximately four minutes to complete.

The sonification program also served as a telemetry processing program, recording, for each driving run, the time of the course start and finish and calculating the course duration time as well as the cumulative duration of excursions outside the rover tilt safety envelope (i.e., all times when the alarm was activated). A log file containing a detailed recording of position and orientation telemetry was generated by the driving simulator during each run as well.

During the course of the experimentation, one subject generated strange data on a majority of runs, possibly indicating a lack of understanding or acceptance of the rover tilt safety limit, and possibly even a suspicion that the limit was fictitious and the task was not as described. A replacement subject was sought out immediately. This added subject was given the same condition sequence as the rejected subject. All experimental data reported in this thesis, including subject sample size, age, and gender, omit the data on the rejected subject and include data on the added subject.

After completing all driving runs, two subjects each made independent, unprompted comments to do with music and how musical ability on their part might have given them an advantage in making use of the sonification. Prompted by these comments, after the formal experiment was complete, an added background question was sent to all subjects, and responses were received from each one of them. The question was “How would you rate your musical ability (based on music performance, music appreciation, and music education), on a scale from 1 to 7?” followed by the visual aid “scale: (low)1, 2, 3, 4, 5, 6, 7(high)”. The question is also included as part of Appendix D. The subject self-assessments of musical ability were expected to be useful as a covariate in the data analysis, based not just on the comments by the two subjects, but also by literature on musical ability allowing better performance of tasks involving the use of sonifications with system data mapped to pitch (Neuhoff, Knight, & Wayand, 2002).

Chapter 5
Results and Data Analysis

12 General Analysis Procedure

A model with a two-level categorical term for the presence or absence of voice, a five-level categorical term for the auditory display design, an interaction term, and the covariates mentioned previously was created for each of the outcome variables. Planned contrasts were the main source of experimental results, however. Since not all conditions of soundscape were used in the experiment (see Table 2), data analysis involved the use of planned contrasts conducted on two separate but overlapping subsets of the data in order to facilitate the investigation of the impact of individual display design elements (datum sonification presence, reference sonification presence, transfer function). Datum Subset 1 was all data except those for conditions with sonification of data but no reference sonification (Table 3). Datum Subset 2 was data only for conditions in which voice communication background noise was present (Table 4).

Table 3 – Datum Subset 1 conditions (white background)

(all conditions except those with sonification of data but no reference sonification)

Condition	Voice	Auditory Feedback	Values Sonified	Datum Sonification Frequency Mapping
1	voice	alarm	-	-
2	voice	alarm, sonification	data	linear
3	voice	alarm, sonification	data	quadratic
4	voice	alarm, sonification	data, reference	linear
5	voice	alarm, sonification	data, reference	quadratic
6	no voice	alarm	-	-
7	no voice	alarm, sonification	data	linear
8	no voice	alarm, sonification	data	quadratic
9	no voice	alarm, sonification	data, reference	linear
10	no voice	alarm, sonification	data, reference	quadratic

Table 4 – Datum Subset 2 conditions (white background)

(conditions with background voice communication)

Condition	Voice	Auditory Feedback	Values Sonified	Datum Sonification Frequency Mapping
1	voice	alarm	-	-
2	voice	alarm, sonification	data	linear
3	voice	alarm, sonification	data	quadratic
4	voice	alarm, sonification	data, reference	linear
5	voice	alarm, sonification	data, reference	quadratic
6	no voice	alarm	-	-
7	no voice	alarm, sonification	data	linear
8	no voice	alarm, sonification	data	quadratic
9	no voice	alarm, sonification	data, reference	linear
10	no voice	alarm, sonification	data, reference	quadratic

Data were analyzed using the mixed linear model framework and the SAS “MIXED” procedure with a compound symmetry variance-covariance matrix structure to account for the within-subject (repeated measures) experimental design. Contrasts were performed using the “ESTIMATE” command statement. The code can be found in Appendix E.

Analysis was conducted with some covariates. Despite some counterbalancing in presentation of the experimental conditions, counterbalancing was incomplete and there was still some effect of order (position of the condition in the sequence). Since the incomplete counterbalancing used concerned only balancing frequencies of pairs of conditions, and thus guarded only against first-order learning effects, the effect of position is likely due to learning effects of second order and beyond, where learning from one condition carries forward further than to just the next condition in the sequence and affects performance in these later conditions. The effect of order could also involve fatigue. In any case, data analysis was conducted with order as a covariate. Subject gender and self-rating of musical ability were also used as covariates.

Appendix F contains relevant data analysis output not included in this chapter. The Studentized residual plots shown were used to assess model adequacy in terms of homoscedasticity and of normality of the residuals.

A summary of the more informative statistical results follows the more detailed exposition directly below.

13 Outcome Variables

Many outcome variables were recorded in this experiment. Variables based directly on rover telemetry were the cumulative time span above rover tilt safety limit, the time span to complete course, and the time span to complete course adjusted with penalty of twice the cumulative time span above safety limit. Variables based on user assessments were five individual response items from the NASA TLX (Task Load Index) questionnaire and the overall rating based on the SUS (System Usability Score) questionnaire. Note that user ratings of the auditory displays were performed for each condition, meaning feedback was received on most display designs both for with and without background voice communications.

13.1 Cumulative Time Span above Safety Limit

The first outcome variable for which the effects of the manipulated variables were assessed was the cumulative time span spent above the rover tilt safety limit (measured in milliseconds). This is the sum of all time spans during which the alarm was activated, and represents the accuracy component of the speed-accuracy trade-off inherent in this task. It is a good indicator of the amount of vehicle damage or other problems that would be caused in a real system as a function of time spent at hazardous or detrimental tilt angles. It is also the clearest indicator of the success or failure of the auditory feedback system given that it reflects the ability of operators to avoid being above the safety threshold. Figure 12 shows the data on this variable.

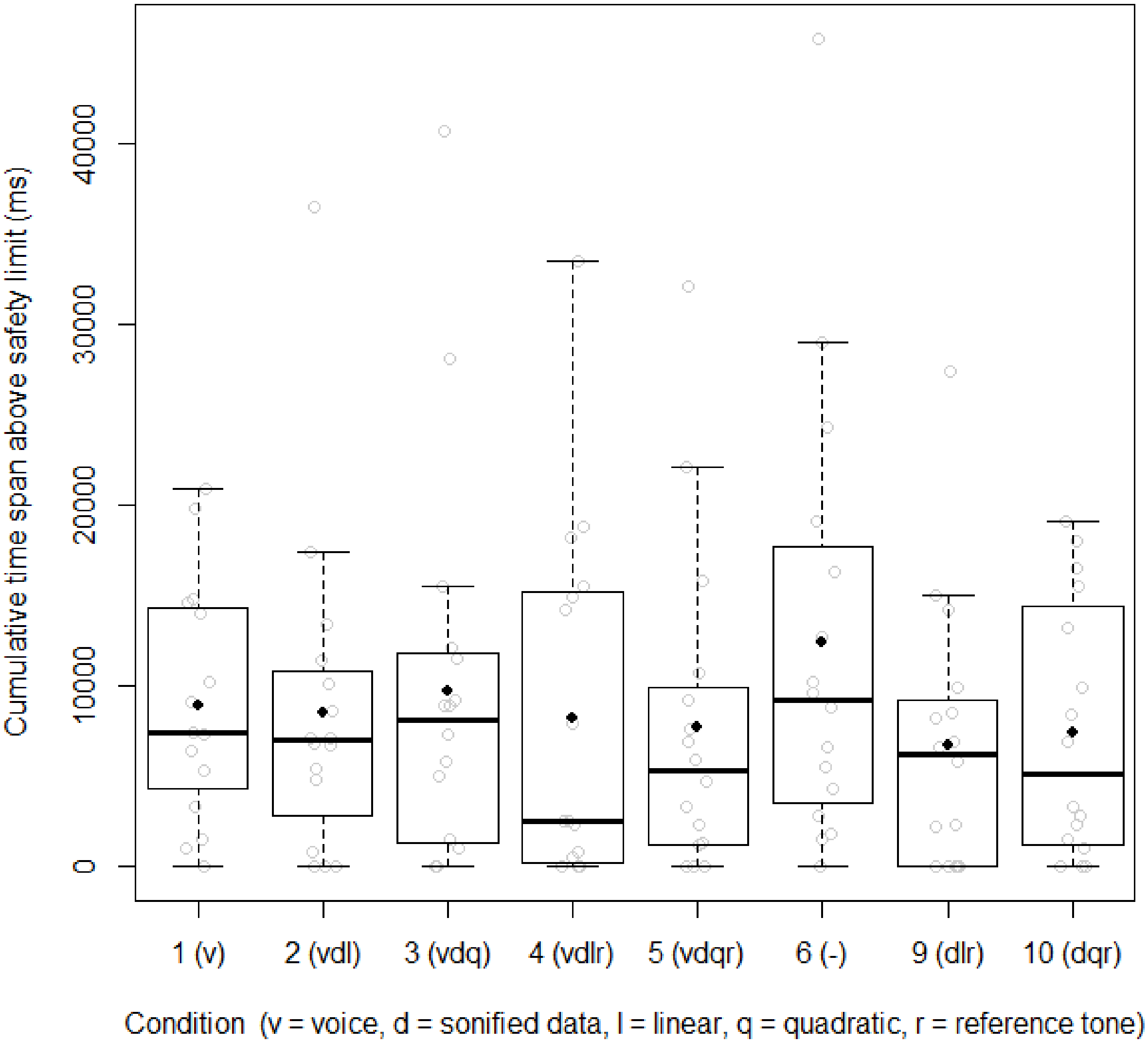


Figure 12 – Condition vs. cumulative time span above safety limit (ms)
(dots at means; whisker ends at most extreme data within 1.5 times interquartile range)

No significant effect of voice was found (*F*(1, 15) = 0.21, *p* = 0.66), no significant effect of display design was found (*F*(4, 60) = 1.77), *p* = 0.15), and no significant interaction effect was found (*F*(2, 30) = 1.61, *p* = 0.22). The effects of the covariate order and the covariate musical ability were significant (*F*(7, 98) = 3.44, *p* = 0.002; *F*(5, 9) = 4.08, *p* = 0.03), but no significant effect of the covariate gender was found (*F*(1, 9) = 1.62, *p* = 0.24).

Within Datum Subset 1 (which, again, was all data except those for conditions with sonification of data but no reference sonification), four planned contrasts were conducted.

Cumulative time span above safety limit was found to be significantly different between conditions with no sonification and those with both datum and reference sonification ($t(30) = 2.41$, $p = 0.02$). Participants spent an average of 3.17 seconds less above the safety limit when they were provided with datum and reference sonification compared to when the only auditory feedback was the alarm. Further, the same comparison was also statistically significant when the analysis focussed on conditions without background voice ($t(30) = 2.88$, $p = 0.007$). When there was no background voice communication, participants spent an average of 5.37 seconds less above the safety limit when they were provided with datum and reference sonification versus when the only auditory feedback was the alarm.

Cumulative time span above safety limit was found not to be significantly different between sonifications with the perceptually linear mapping versus sonifications with the perceptually quadratic mapping ($t(30) = -0.06$, $p = 0.95$). Similarly, within conditions without background voice communication, there was no significant difference between perceptually linear mapping and perceptually quadratic mapping ($t(30) = -0.33$, $p = 0.74$).

Within Datum Subset 2 (which, again, included data only for conditions in which voice communication background noise was present), four additional planned contrasts were conducted. All of these contrasts were among conditions with background voice communication, and there were no statistically significant findings: no sonification versus sonification of data only ($t(53) = 0.13$, $p = 0.90$); sonification of data only versus datum and reference sonification ($t(53) = 0.97$, $p = 0.34$); no sonification versus datum and reference sonification ($t(53) = -0.66$, $p = 0.51$); perceptually linear mapping versus perceptually quadratic mapping ($t(53) = 0.13$, $p = 0.78$).

13.2 Time Span to Complete Course

Analysis was also done based on the time span required to complete the course. This represents the speed component of the speed-accuracy trade-off inherent in this task, and as such is less of an indicator of the performance of the different auditory displays. Figure 13 shows the data on this variable.

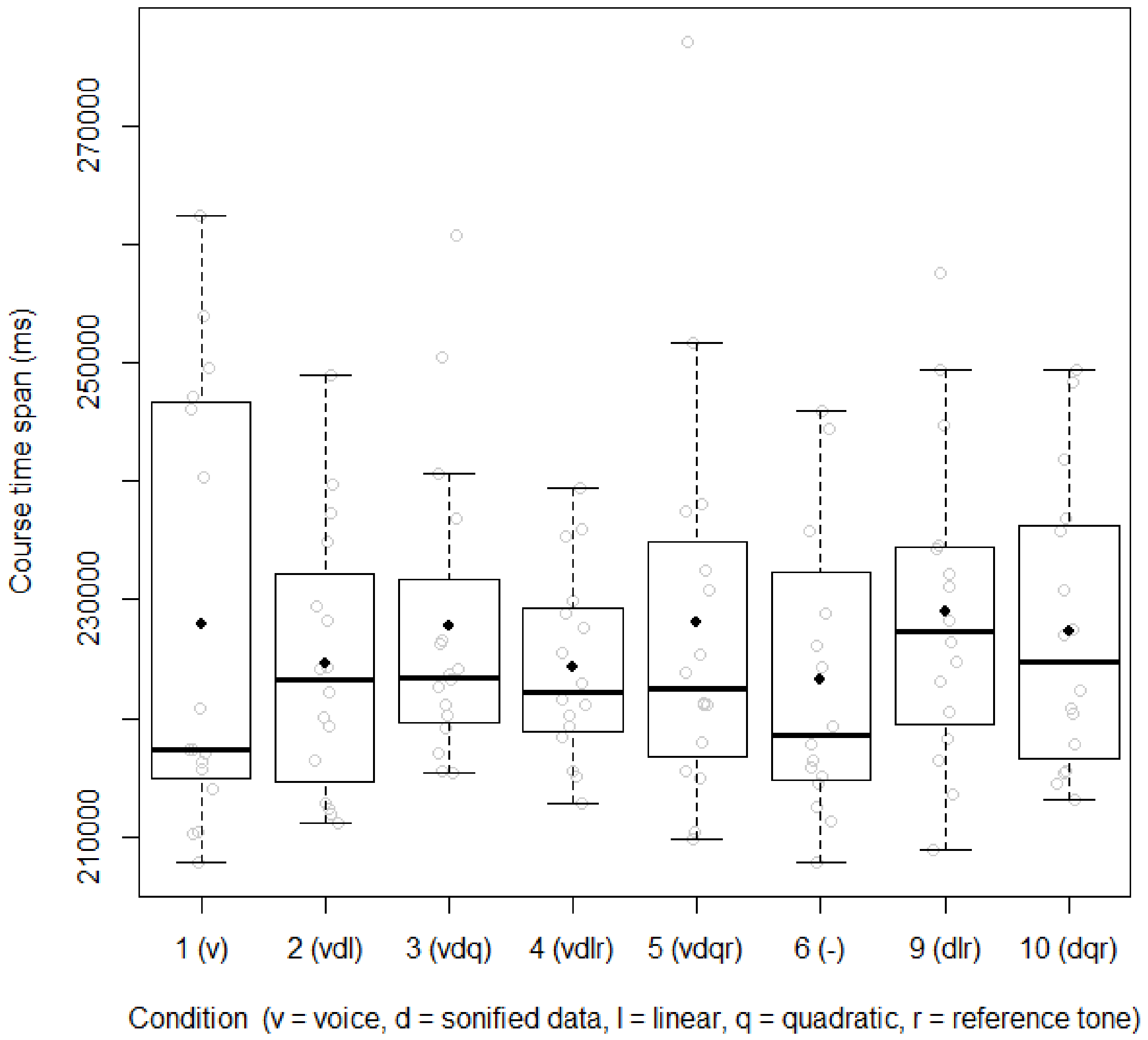


Figure 13 – Condition vs. course completion time span (ms)
(dots at means; whisker ends at most extreme data within 1.5 times interquartile range)
(vertical axis not grounded to zero)

No significant effect of voice was found (*F*(1, 15) = 0.01, *p* = 0.91), no significant effect of display design was found (*F*(4, 60) = 0.44, *p* = 0.78), and a no significant interaction effect was found (*F*(2, 30) = 0.16, *p* = 0.85). No significant effect of the covariate order was found (*F*(7, 98) = 1.52, *p* = 0.17), but the effects of the covariate musical ability and the covariate gender were significant ($F(5, 9) = 4.20$, $p = 0.03$; $F(1, 9) = 8.96$, $p = 0.02$). Female subjects took on average 12.66 seconds more to complete the course.

Within Datum Subset 1 (again all data except those for conditions with sonification of data but no reference sonification), the same four planned contrasts as for other outcome variables were conducted. No significant differences were found.

In particular, course completion time span was found not to be significantly different between conditions with no sonification and those with both datum and reference sonification ($t(30) = -0.77$, $p = 0.45$). Similarly, within conditions without background voice communication, there was no significant difference between no sonification and datum plus reference sonification ($t(30) = -1.67$, $p = 0.11$).

In addition, course completion time span was found not to be significantly different between sonifications with the perceptually linear mapping and sonifications with the perceptually quadratic mapping ($t(30) = -0.43$, $p = 0.67$). Similarly, within conditions without background voice communication, there was no significant difference between perceptually linear mapping and perceptually quadratic mapping ($t(30) = 0.48$, $p = 0.63$).

Within Datum Subset 2 (again data only for conditions in which voice communication background noise was present), the same four additional planned contrasts were conducted. All of these contrasts were among conditions with background voice communication, and there were no statistically significant findings: no sonification versus sonification of data only ($t(53) = -0.49$, $p = 0.62$); sonification of data only versus datum and reference sonification ($t(53) = -0.01$, $p = 0.99$); no sonification versus datum and reference sonification ($t(53) = -0.49$, $p = 0.63$); perceptually linear mapping versus perceptually quadratic mapping ($t(53) = -1.19$, $p = 0.24$).

13.3 Course Completion Time Span Plus Penalty

Analysis was also done based on the quantity subjects were told would represent performance in the task. This quantity was the course completion time span adjusted by adding a penalty of twice the cumulative time span above the safety limit. Insofar as the specific trade-off expressed to subjects reflects the real-world cost-benefit between speed and accuracy, adjusted course completion time span, like (raw) course completion time span, is a good indicator of the effectiveness of the different auditory displays. Figure 14 shows the data on this variable.

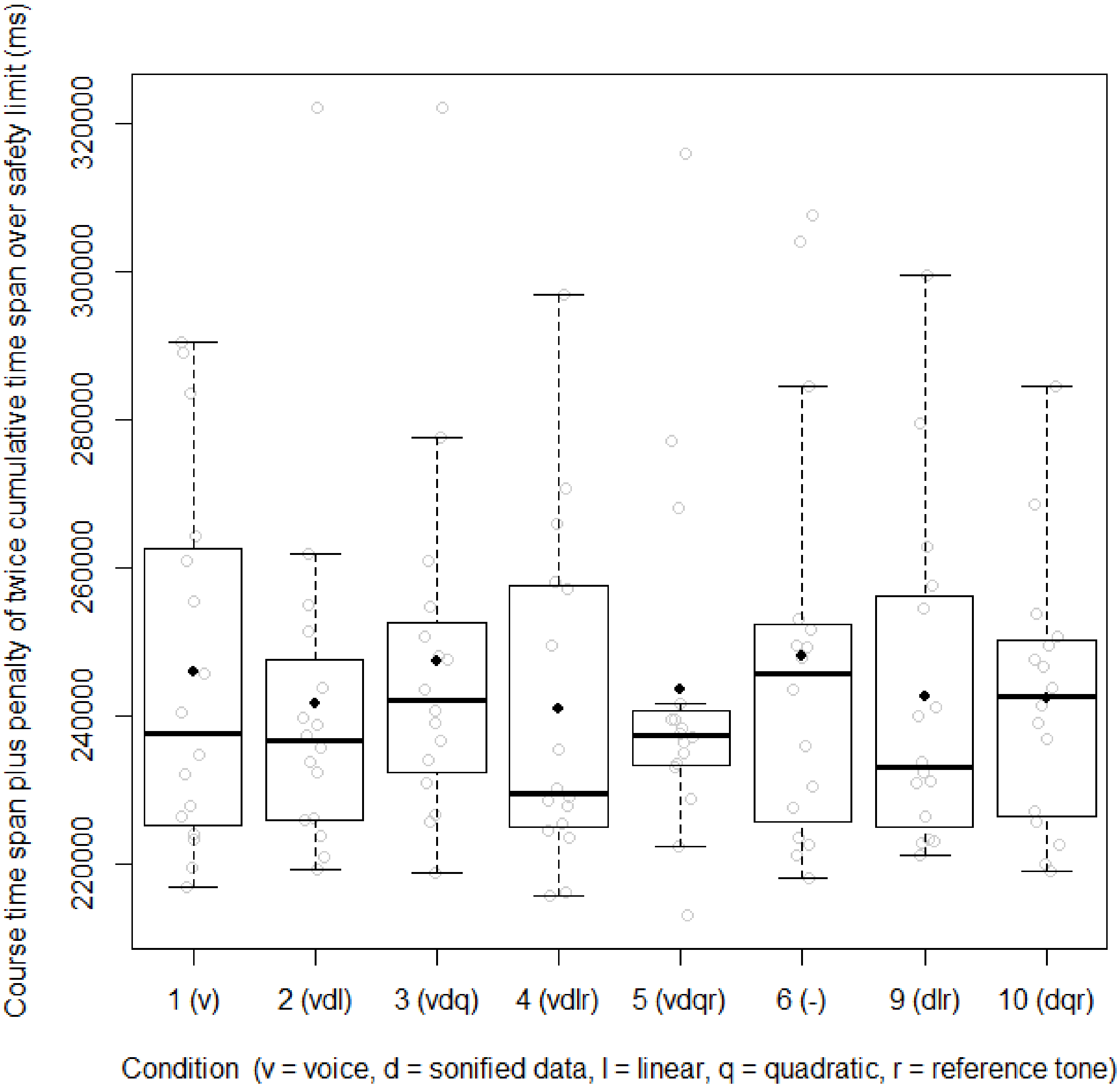


Figure 14 – Condition vs. course completion time plus penalty of twice alarm time (ms) (dots at means; whisker ends at most extreme data within 1.5 times interquartile range) (vertical axis not grounded to zero)

No significant effect of voice was found (*F*(1, 15) = 0.10, *p* = 0.76), no significant effect of display design was found (*F*(4, 60) = 1.12, *p* = 0.35), and no significant interaction effect was found ($F(2, 30) = 0.16$, $p = 0.85$). The effects of the covariate order, the covariate musical ability, and the covariate gender were all significant ($F(7, 98) = 4.09$, $p = 0.0006$; $F(5, 9) = 11.48$, $p = 0.001$; $F(1, 9) = 12.71$, $p = 0.006$). Female subjects had, on average, an adjusted course completion time that was 20.72 seconds higher than males had.

Within Datum Subset 1 (again all data except those for conditions with sonification of data but no reference sonification), the same four planned contrasts as for other outcome variables were conducted. There were no significant findings.

In particular, adjusted course completion time span was found not to be significantly different between conditions with no sonification and those with both datum and reference sonification ($t(30) = 1.63$, $p = 0.11$). Similarly, within conditions without background voice communication, there was no significant difference between no sonification and datum plus reference sonification ($t(30) = 1.42$, $p = 0.17$).

In addition, adjusted course completion time span was found not to be significantly different between sonifications with the perceptually linear mapping versus those with the perceptually quadratic mapping ($t(30) = -0.36$, $p = 0.72$). Similarly, within conditions without background voice communication, there was no significant difference between perceptually linear mapping and perceptually quadratic mapping ($t(30) = 0.05$, $p = 0.96$).

Within Datum Subset 2 (again data only for conditions in which voice communication background noise was present), the same four additional planned contrasts as for other outcome variables were conducted. All of these contrasts were among conditions with background voice communication, and there were no statistically significant findings: no sonification versus sonification of data only ($t(53) = -0.39$, $p = 0.70$); sonification of data only versus datum and reference sonification ($t(53) = 0.81$, $p = 0.42$); no sonification versus datum and reference sonification ($t(53) = -1.06$, $p = 0.30$); perceptually linear mapping versus perceptually quadratic mapping ($t(53) = -1.45$, $p = 0.15$).

This last result, though non-significant, is the only indication of the experimental manipulations related to mapping transfer function having had any effect. If anything at all can be concluded from the fact that it approaches significance for this outcome variable (course completion time adjusted with penalty for time above safety limit) it may be that the transfer function affects the apparent suddenness with which the limit is crossed and therefore the time required to reverse up-slope momentum and return below the limit.

13.4 User Rating of System Conditions – NASA Task Load Index

A subset of questions from the NASA TLX (no ‘Physical Demand’) was administered for each condition. Figures 15 to 19 present the raw results (not weighted as per the true NASA TLX).

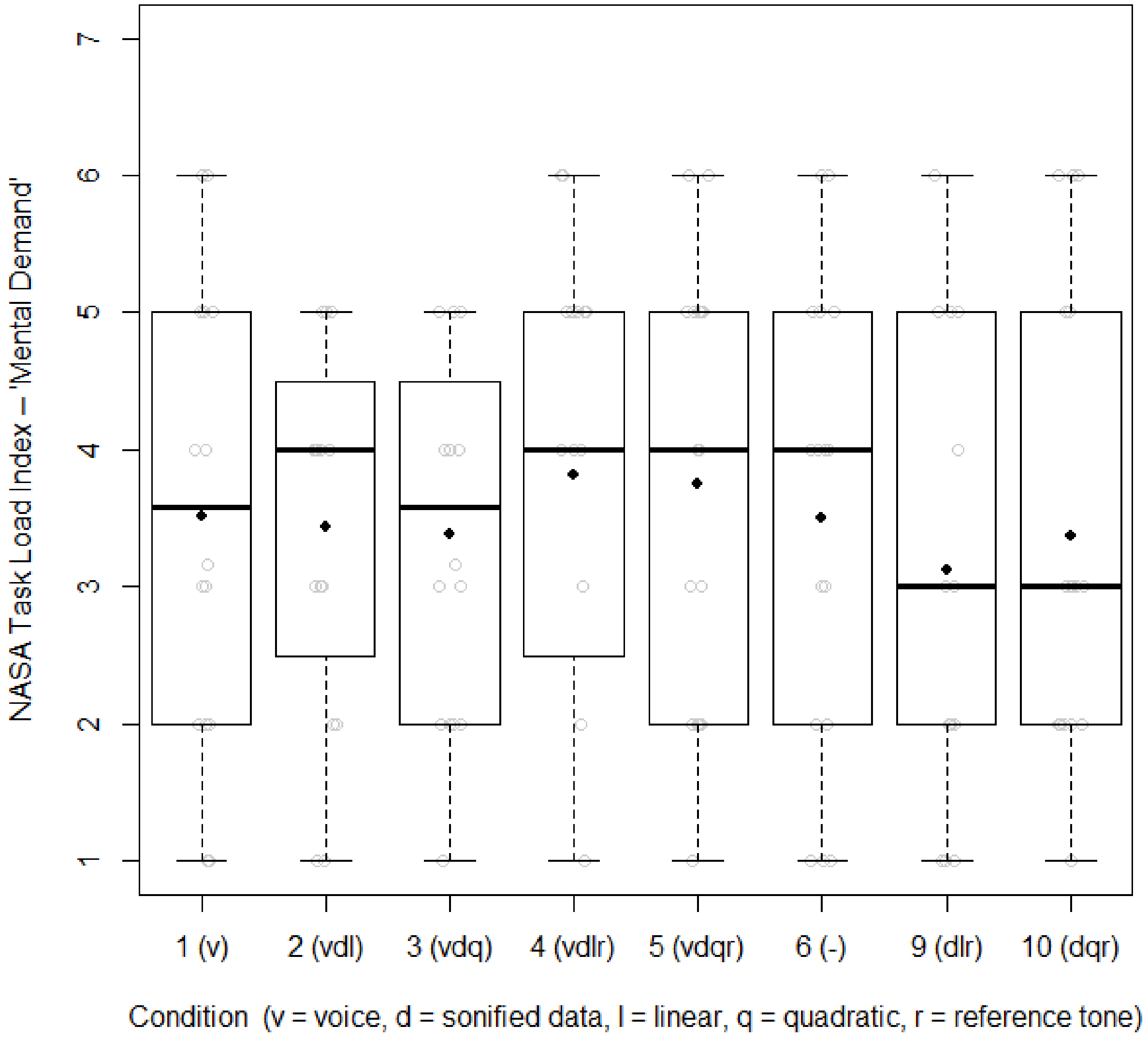


Figure 15 – NASA TLX responses for ‘Mental Demand’ item (1 = low, 7 = high)
(dots at means; whisker ends at most extreme data within 1.5 times interquartile range)

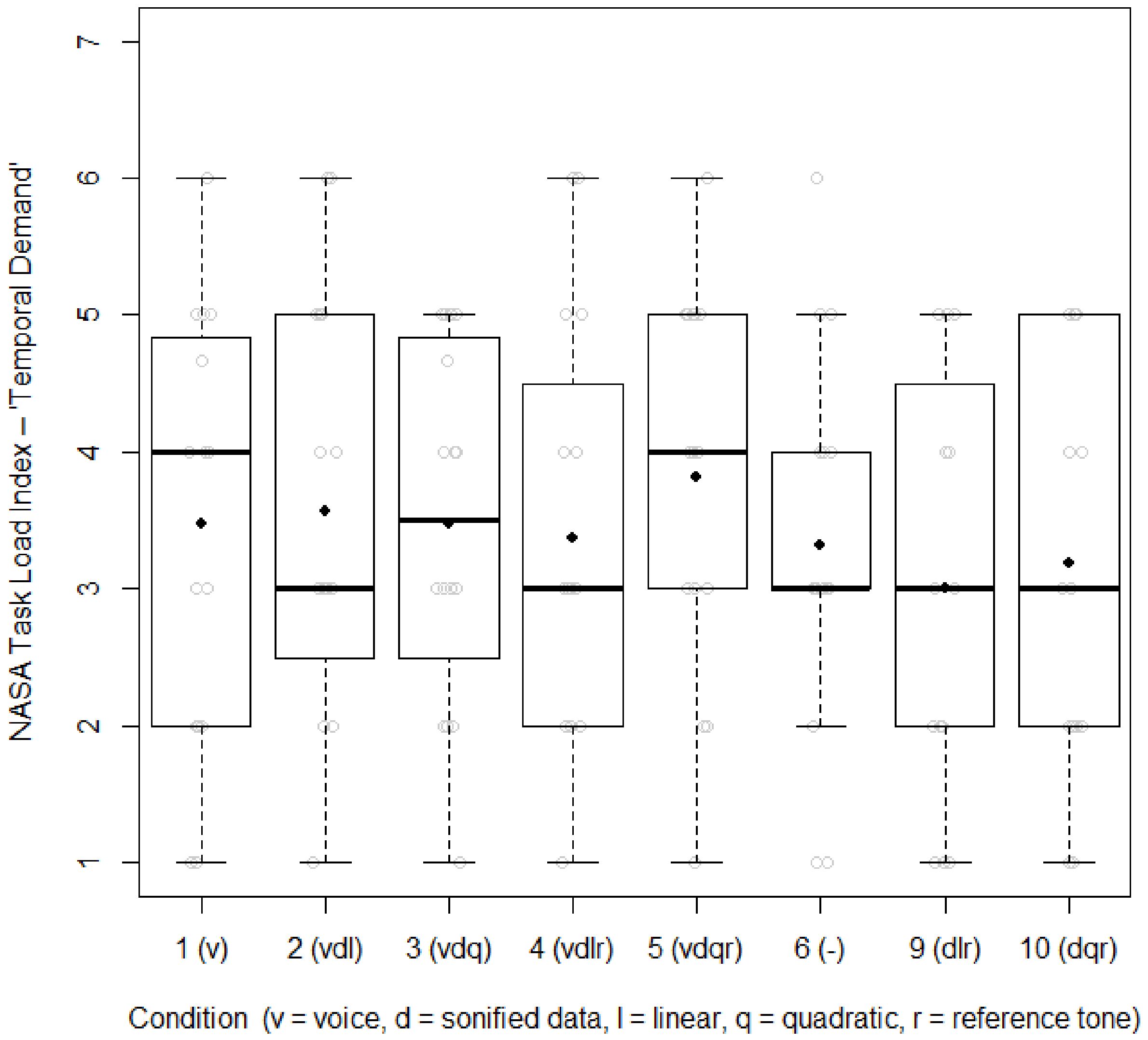


Figure 16 – NASA TLX responses for 'Temporal Demand' (1 = low, 7 = high)

(dots at means; whisker ends at most extreme data within 1.5 times interquartile range)

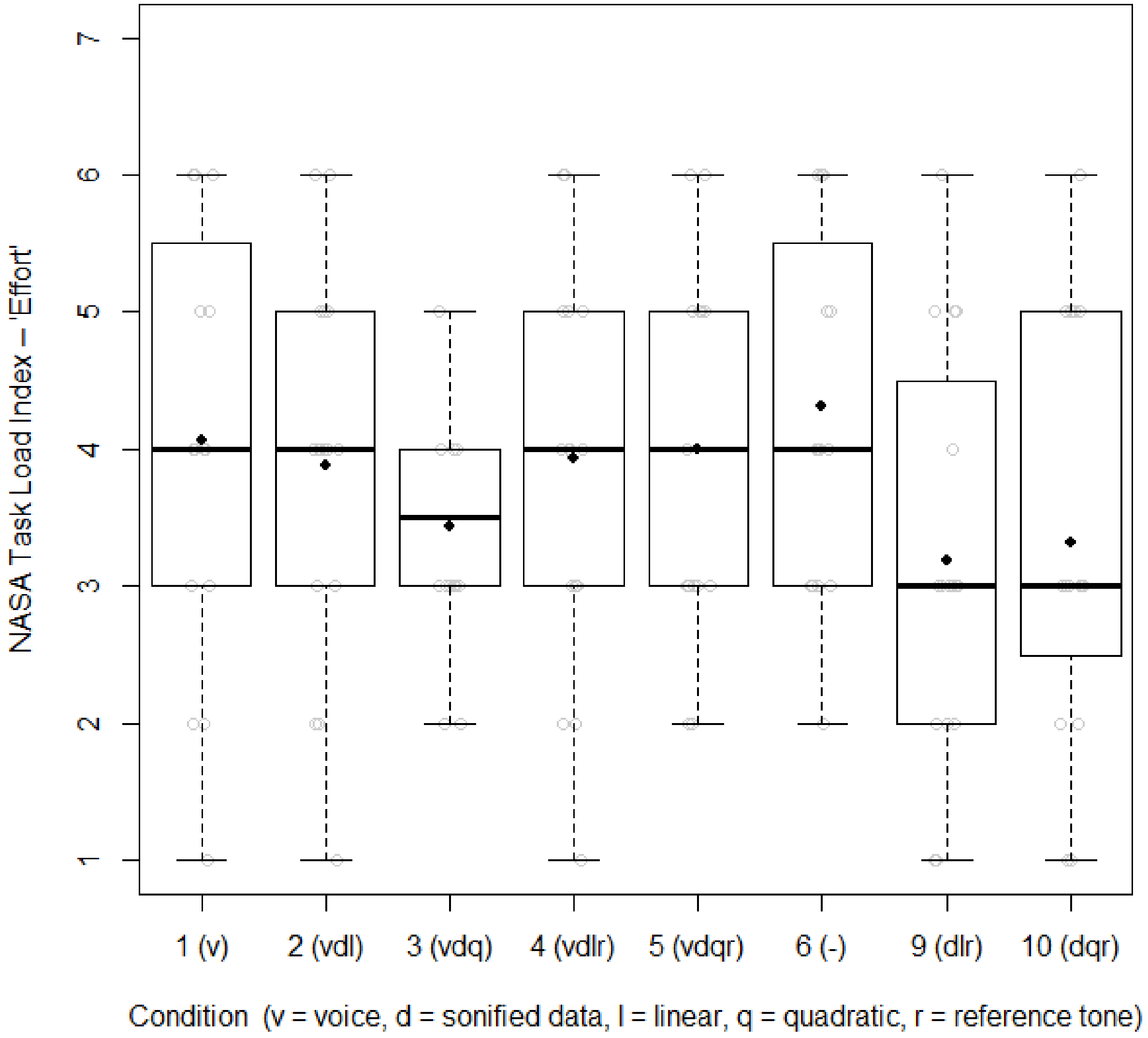


Figure 17 – NASA TLX responses for item 'Effort' (1 = low, 7 = high)

(dots at means; whisker ends at most extreme data within 1.5 times interquartile range)

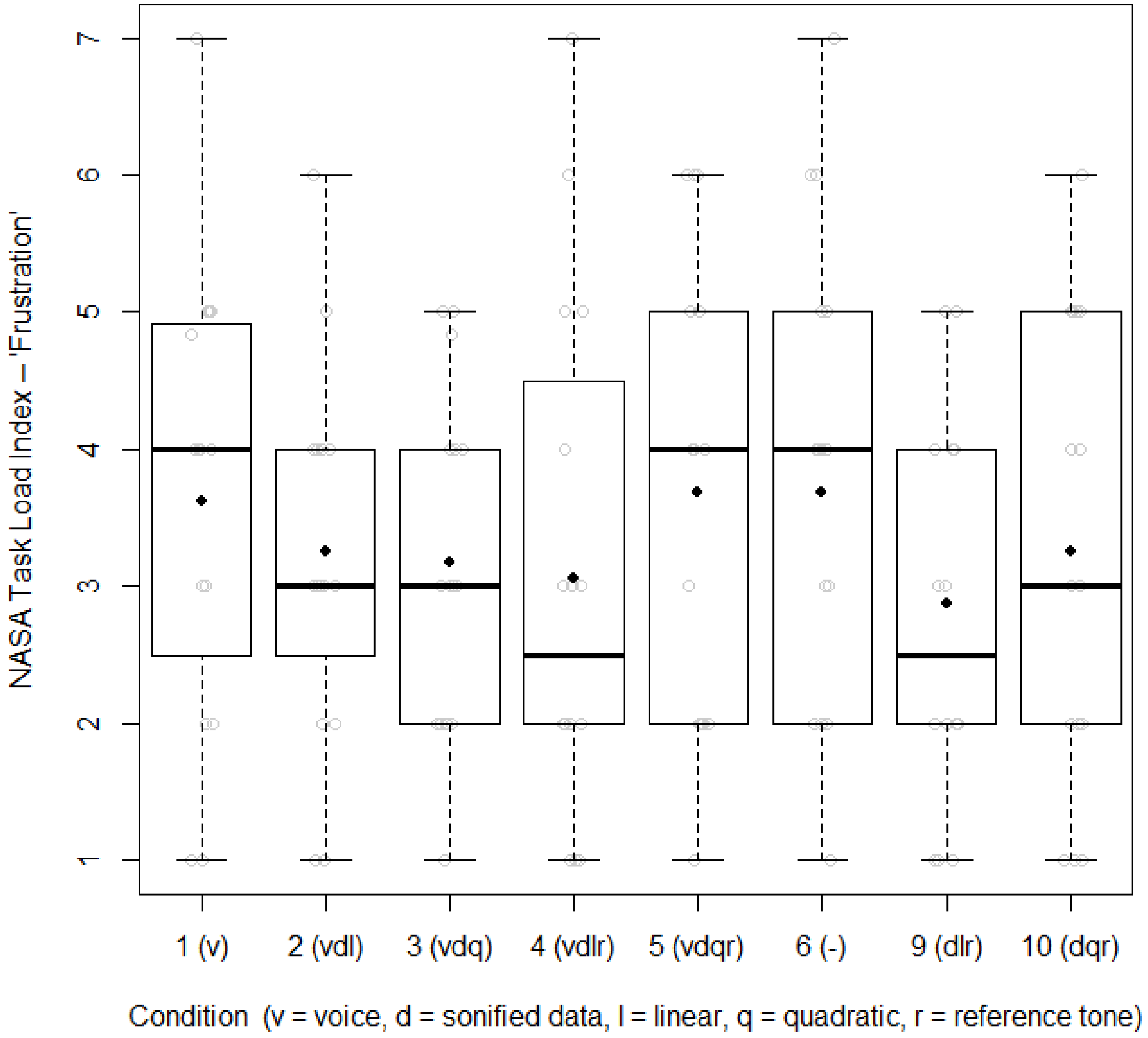


Figure 18 – NASA TLX responses for item 'Frustration' (1 = low, 7 = high)

(dots at means; whisker ends at most extreme data within 1.5 times interquartile range)

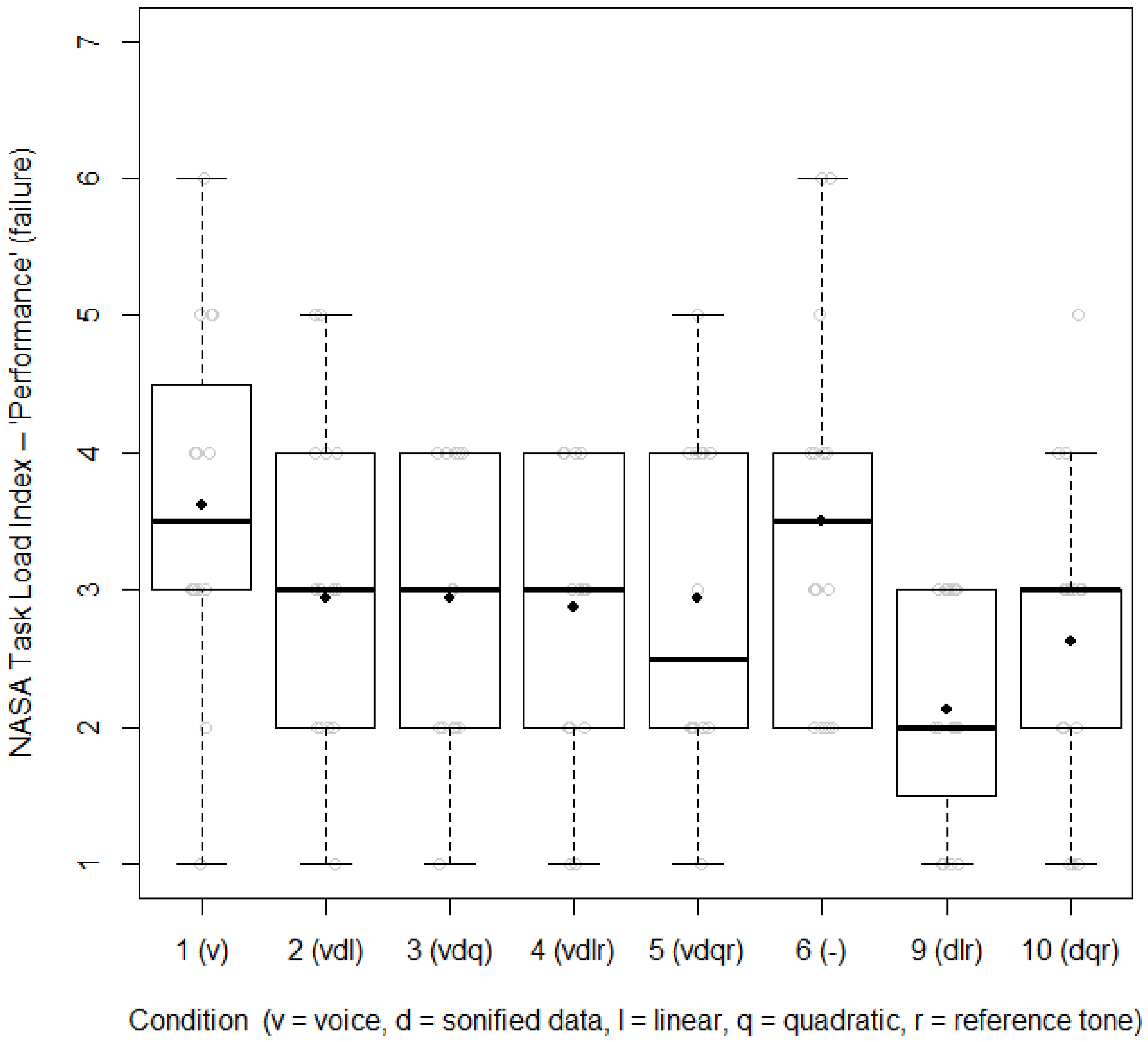


Figure 19 – NASA TLX responses for item 'Performance' (1 = perfect, 7 = failure) (dots at means; whisker ends at most extreme data within 1.5 times interquartile range)

Inspection of the figures reveals no clear differences between the soundscapes in terms of any NASA TLX response items. Statistical analysis would need to be conducted to confirm the lack of significant differences, but these basic results are reported simply as background information on the soundscapes.

13.5 User Rating of System – System Usability Scale

The SUS questionnaire was administered for each experimental condition. The results are presented in Figure 20.

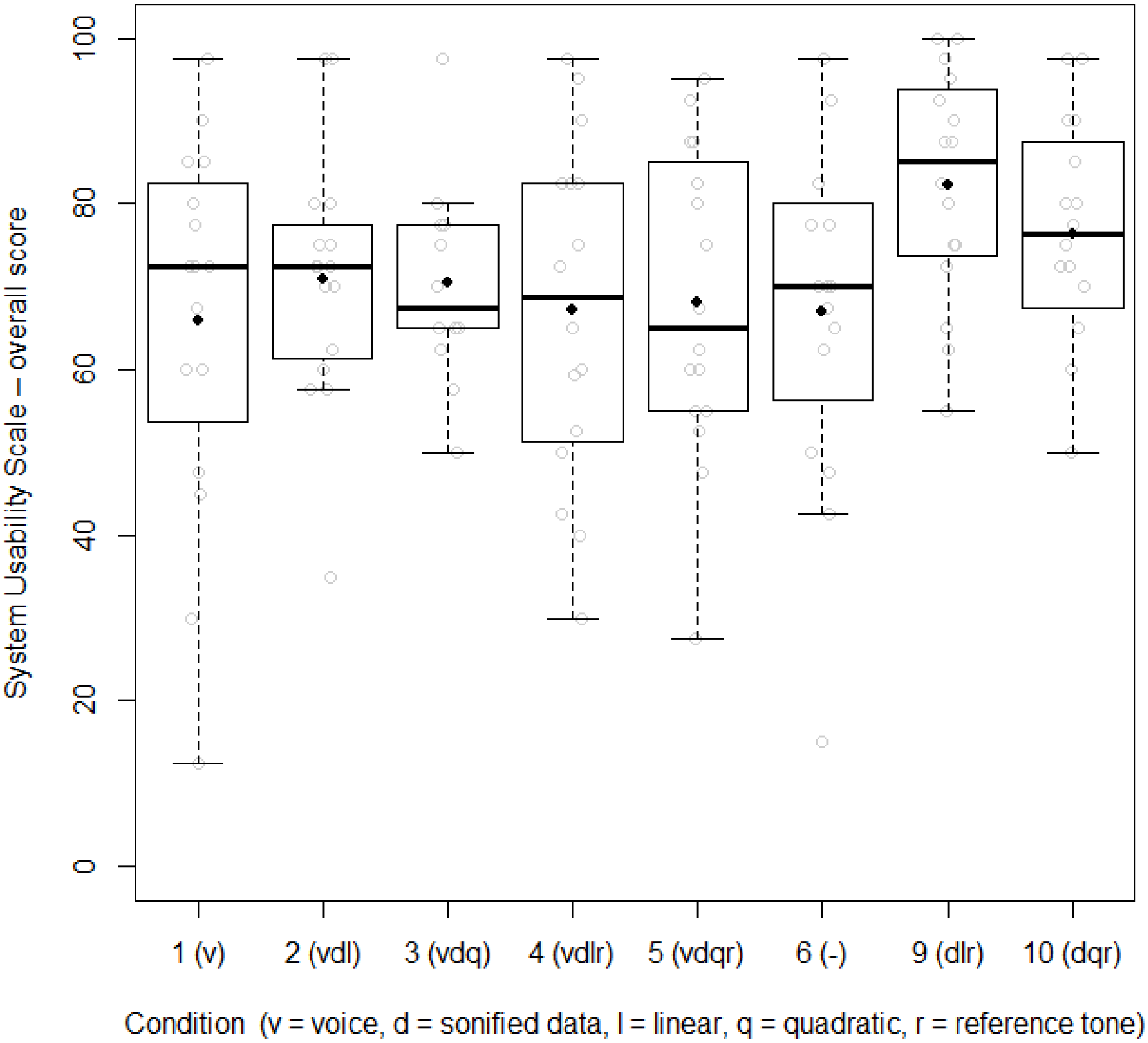


Figure 20 – SUS overall score (higher is better)

(dots at means; whisker ends at most extreme data within 1.5 times interquartile range)

Inspection of the figure reveals no clear differences between the soundscapes in terms of SUS score, though there does appear to have been a slight subject preference for the condition with no background voice communication, the alarm, the datum sonification with the linear transfer function, and the reference sonification. Statistical analysis would be needed to confirm this preference, but these results are presented simply as background information on the soundscapes.

14 Results Summary

Results of the planned contrasts, upon which investigation of the individual display design elements are based, are summarized in Table 5.

Table 5 – Experimental results from planned contrasts

Cumulative time span above safety limit	overall	background voice present	background voice absent
no sonification vs. datum and reference sonification	$t(30) = 2.41$, $p = 0.02$ *	$t(53) = -0.66$, $p = 0.51$	$t(30) = 2.88$, $p = 0.007$ **
no sonification vs. datum sonification only		$t(53) = 0.13$, $p = 0.90$	
datum sonification only vs. datum and reference sonification		$t(53) = 0.97$, $p = 0.34$	
perceptually linear vs. perceptually quadratic mapping	$t(30) = -0.06$, $p = 0.95$	$t(53) = 0.13$, $p = 0.78$	$t(30) = -0.33$, $p = 0.74$

Time span to complete course	overall	background voice present	background voice absent
no sonification vs. datum and reference sonification	$t(30) = -0.77$, $p = 0.45$	$t(53) = -0.49$, $p = 0.63$	$t(30) = 1.67$, $p = 0.11$
no sonification vs. datum sonification only		$t(53) = -0.49$, $p = 0.62$	
datum sonification only vs. datum and reference sonification		$t(53) = -0.01$, $p = 0.99$	
perceptually linear vs. perceptually quadratic mapping	$t(30) = -0.43$, $p = 0.67$	$t(53) = -1.19$, $p = 0.24$	$t(30) = 0.48$, $p = 0.63$

Course completion time span plus alarm penalty	overall	background voice present	background voice absent
no sonification vs. datum and reference sonification	$t(30) = -1.63$, $p = 0.11$	$t(53) = -1.06$, $p = 0.30$	$t(30) = 1.42$, $p = 0.17$
no sonification vs. datum sonification only		$t(53) = -0.39$, $p = 0.70$	
datum sonification only vs. datum and reference sonification		$t(53) = 0.81$, $p = 0.42$	
perceptually linear vs. perceptually quadratic mapping	$t(30) = -0.36$, $p = 0.72$	$t(53) = -1.45$, $p = 0.15$	$t(30) = 0.05$, $p = 0.96$

* $p < 0.05$; ** $p < 0.01$; *** $p < 0.001$

all tests two-tailed, despite any directional hypotheses

The only significant effect found related to the experimental manipulations was that combined sonification of data and the safety limit (along with the alarm) promoted better performance than did the alarm alone in most circumstances. This effect was not apparent for when background voice communication was present, but it was true when voice was absent, and it was true in the general case (the full set of related experimental conditions).

With regard to the experimental hypotheses, Hypothesis 1, that there would be degraded task performance in the conditions with voice communication, is not supported by the F-tests on the main effect of voice. The hypothesis does seem to be weakly supported for the case where auditory displays including sonification of both data and reference value are contrasted with alarm-only displays, however. This conclusion is drawn from the fact that the sonification displays promoted better performance than did the alarm-only displays when background voice communication was absent and in the general case of background voice (voice absent and present), but *not* when voice was present.

Hypothesis 2, that sonification (of system data alone) would promote increased task performance, meaning that sonification in general is suitable for the task in this experiment, is not supported by the F-tests involving the display design term. Conversely, the significant difference in cumulative time span above safety limit between the alarm-only display and the datum and reference sonification display (for the general case) does provide some support. The hypothesis was meant to refer to the effectiveness of sonification of system data only, and this hypothesis remains unproven here, but to attribute the success of the displays with two sonifications entirely to the reference tone seems unreasonable. For the hypothesis to be false, the reference tone would have to provide value on its own, which is not sensible, or it would need to be *required* in order to derive *any* benefit from the sonification of system data, which seems very unlikely.

Hypothesis 3, that sonification (of system data alone) would promote increased task performance even considering only situations with background voice communication, has no support. The contrast between the alarm-only display and the datum and reference sonification display within the set of conditions with exposure to background voice communication showed a non-significant effect.

Hypothesis 4, that reference value sonification would promote better performance, is not supported by the F-tests including the display design term. Similar to the situation for Hypothesis 2, however, support for Hypothesis 4 can be seen in the fact that auditory displays including sonification of system data and reference value yielded positive effects on performance versus alarm only displays. The case is much weaker here, however, since it would be reasonable for the benefit of the dual-sonification display designs over the alarm only designs to be have been due to the datum sonification alone, with the reference sonification contributing nothing. Still, it seems likely that both contributed.

Hypothesis 5, that system variable sonification with a perceptually quadratic mapping would promote performance better than that with a perceptually linear mapping, is also unsupported. Again, in each model the display design term had no significant effect. Also, all planned contrasts between the different transfer functions revealed no significant difference.

Chapter 6
Discussion and Future Work

Most notable among the results is the fact that sonification had a significant effect on cumulative time span above rover tilt angle safety limit when there was no background voice communication (and for the entire subset of data in which that comparison was made), while it did not have a significant effect when there was background speech. This finding suggests, though it was not borne out by a significant related interaction term in the model, that background voice does potentially lower the effectiveness of sonification. Indeed, the interaction term may only have been missed due to the statistical power of the experiment not being great enough, at least in the case of the outcome variable cumulative time span above safety limit. This possible interaction would seem to endorse sonification for applications such as those in this experiment when there is no, or possibly just *less*, background voice communication. This conclusion should be studied further, however. It is also possible that a positive effect of sonification was not found with background voice due to a lack of statistical power.

Ideally the effects of sonification schemes should have been compared with those of analogous visual feedback mechanisms which could serve as controls. This possibility was explored for the present experiment, but it was considered too difficult to implement within the experimental apparatus. The suitability of sonification (along with the alarm) compared with the suitability of a similar visual display component for this driving task, is left to future comparison studies of the two modalities for this or similar applications. It may be able to be inferred with reference to any existing literature on comparisons of sonification with visual feedback for other tasks and environments, but there would still be benefit to application-specific testing. Of course, in contexts where the visual modality is known to be near capacity, the case for use of sonification is automatically stronger. Regardless of what levels of performance might be associated with visual modality display options, this experiment demonstrated a positive effect of the sonifications designed for the task, as least when “sonifications” is taken to include representation of both the system variable and a reference value.

Another possible counter to the conclusion that sonification should be implemented in displays for applications such as that for this experiment (though potentially limited to those with no background speech) is the fact that the alarm could possibly have been used more effectively. Future work could focus on determining better use of the alarm or multiple alarms. By leaving some tolerance between an alarm and the safety limit rather than co-locating it, excursions beyond the safety limit will certainly be able to be reduced without resorting to sonification, which has drawbacks. Though it is clear that accuracy could be improved, however, experimentation could determine what levels of speed and accuracy together could be promoted by the use of an alarm or several alarms in this fashion.

Experimentation could reveal the relationship between the alarm trigger value's distance from the safety limit and task performance for a given task. There is likely to be a best distance at which to "place" one alarm (or alert), and there are also likely to be best distances at which to place multiple alarms or alerts as well. Task performance with these well-tuned alarm-based auditory displays would be the best basis of comparison for performance with different sonification designs, for a more realistic impression of the value of sonification versus more simple auditory displays.

Supporting the notion of sonification being suitable for the rover driving task, and suggesting room for further gains is the fact that the sound design of the sonification was very simplistic in terms of timbre. The sonification was successful despite being a pure-tone changing only in pitch, and that leaves much room for exploration of superior sonification designs. In particular, sonification based on a sound with a very different timbre could be very differently affected by the presence of background voice communication. The same could be true of a sonification made to vary in more than one dimension, or one simply occupying a different frequency band (perhaps wider, perhaps more deliberately separated from the more interfering aspects of human speech sound).

Aside from the general prospective usefulness of sonification, the variables manipulated in this experiment could be approached differently for potentially different outcomes. The majority of experimental manipulations had no significant effect for most outcome variables, and though this could mean that some or all of the auditory interface manipulations performed are not worthwhile for this and similar applications, there are other possible reasons for this, and many things that could be attempted in future experiments to increase the power to detect related effects.

Using a different timbre in the sound on which the sonification is based may provide benefits other than just to the ability of the sonification to be effective in the presence of voice communication. A given timbre could be more effective simply by being more salient and garnering more attention, or through less clear effects such as being less tiring to listen to than are pure tones or sounds with worse timbres.

The effectiveness of reference sonification may also be able to be improved by implementing more levels of reference. Multiple levels can be conveyed without ever needing to display more than one at any given time. Using a dynamic reference tone (sonifying, depending on data trend, the upper or lower value) helped effectiveness in the experimentation by Smith & Walker with auditory graphs (2002). Indeed, in that study a single reference tone provided no significant benefit, though in the task in that experiment there was need for precision throughout the full range of sonified values. For the task in the current experiment, it was expected that since there was only need for precision near a single value, a single reference tone would suffice. Having no fewer than two levels of reference may confer advantage even for a task with a single, key value. This may be due to two levels providing a better sense of scale by reducing the amount of memory of past mapping values needed to make precise measurements from the sonification.

Reference sonification might also be more likely to produce significant effects if the subjects used are restricted to those with musical ability. One experimental subject mentioned the ability to perceive interference patterns between sounds of similar frequency as being instrumental in being able to approach but not exceed the safety limit, which would mean better use of the reference tone. Musical ability was included in the model only as a covariate, with no related interaction terms, so while it was shown to have a significant effect, the effect appears to have been due to a unique musical ability rating by a subject with particularly poor performance. Also, musical ability having an effect (or performance) in isolation does not mean that musical ability affects display effectiveness. It is left as future work to look at the interaction effect between musical ability and auditory display design on performance. In any case, the display users could be trained on or selected for prior ability to notice the particular phenomenon of interference patterns, rather than being trained on or selected for ability in the broader area of music.

To improve the effectiveness of the sonification in general, but at the expense of the effectiveness of the reference sonification in particular, more training on the system could be provided to subjects before experimentation. Smith and Walker (2005) found that training and reference tones appear to have positive but not additive effects on performance.

The variables tested here may of course simply have modest effects, in which case research using a more controlled task and a larger sample size may be necessary to measure these effects.

Still more avenues for future work arise if the failed hypotheses of this experiment prove true with more statistical power or under different circumstances. Should a non-linear transfer function be found to promote better performance than does a linear mapping in some future context (likely one with an important boundary value or range of values), further work could be done to test whether the non-linear mapping also makes the sonification more resilient against any negative effects of background voice communication. This could be the case if frequency judgement is made more difficult in the presence of masking by speech, since adding extra resolution for more important values of a system variable should provide value by making any errors in frequency judgement less important.

In the context of a future sonification for which a reference sonification is also shown to improve performance (or indeed simply revisiting existing sonification designs for which that is true), testing the interaction between transfer function and reference sonification presence or absence could be worthwhile. Reference sonification could prove to be less effective when used in combination with a quadratic datum sonification mapping, for example. The design goal behind providing higher-resolution display of higher system variable values was to make absolute judgement easier, and less challenging absolute judgement should reduce the benefit of using reference sonification to shift to relative judgement. The shift may be valuable regardless, or the relationship of reference tones with novel mappings may be akin to that of reference tones with training. That is, they may have positive but not additive effects on performance.

A related consideration is that the novel mapping may just not have been enough of a departure from the standard linear mapping. The mappings appear quite different in terms of the relationship between rover tilt angle and sonification frequency (as shown in Figure 8), but in terms of the better indicator of the psychological output from the mappings, the relationship between rover tilt angle with expected pitch, the two mappings appear much more subtly different (as shown in Figure 9). This suggests that pitch-based auditory displays with perceptually cubic or even quartic, quintic, higher-order, or various bold non-polynomial mappings should be studied.

Chapter 7
Conclusion

This experiment provided support for the use of sonification of the system parameter of rover tilt angle for increasing performance in a task of operating the rover in the presence of limits on maximum safe tilt angle and a task where operating near these limits is advantageous. It did not provide support for the use of sonification for this purpose in the context of background noise, however. The finding of a sonification display being useful for a rover tilt angle in this case may be generalizable to other vehicle operation tasks and other tasks with similar levels of need for continuous control, and to tasks with similar levels of engagement of vision and hearing..

The design techniques used in this experiment may be specifically useful in the wide range of tasks involving any intent to approach but not exceed some threshold. These tasks include any task where the limits of a system are to be pushed, but where there is a cascade effect, feedback loop that must not be triggered, or where there is any negative outcome that could arise from a change in a system variable value across some threshold. In addition to examples of such variables given earlier, another would be temperature in situations where phase change of a material could be triggered.

As discussed previously, the timbre of the base sound of the sonification and the frequency range of the sonification could be carefully selected to allow it to be more easily attended to in the presence of human speech. Doing so and possibly also mapping the system variable to more than just frequency would lead to more effective sonifications. With regard to improvements in the theory of sonification, research on auditory graphs has already covered ground on the effectiveness of reference sonification (Smith & Walker, 2002), but it may be worthwhile for the research presented here on the effectiveness of different sonification mapping transfer functions to be continued with other auditory displays, especially if these mappings vary display resolution more strongly with system variable value. Testing the relative usefulness of such mappings could probably also benefit from more of a laboratory setting, paired with a better basic sonification design and with a task for which changes in performance are more easily detectable and less subject to variance due to outside factors.

References

Anderson, J. E., & Sanderson, P. (2009). Sonification design for complex work domains: Dimensions and distractors. *Journal of Experimental Psychology: Applied*, *15*(3), 183–198. doi:10.1037/a0016329

Barrass, S., & Kramer, G. (1999). Using sonification. *Multimedia Systems*, *7*(1), 23–31.

Bliss, J. P., & Gilson, R. D. (1998). Emergency signal failure: Implications and recommendations. *Ergonomics*, *41*(1), 57–72.

Brooke, J. (1996). SUS – A quick and dirty usability scale. *Usability evaluation in industry*, *189*, 194.

Burk, P. (1998). JSyn–a real-time synthesis API for Java. In *Proceedings of the 1998 International Computer Music Conference* (pp. 252–255).

Chou, E., Lim, J., Brant, R., Ford, S., & Ansermino, J. M. (2008). Accuracy of detecting changes in auditory heart rate in a simulated operating room environment*. *Anaesthesia*, *63*(11), 1181–1186. doi:10.1111/j.1365-2044.2008.05629.x

Dai, H., & Micheyl, C. (2011). Psychometric functions for pure-tone frequency discrimination. *The Journal of the Acoustical Society of America*, *130*(1), 263. doi:10.1121/1.3598448

Davison, B. K., & Walker, B. N. (2007). Sonification Sandbox reconstruction: Software standard for auditory graphs. In *ICAD 07-Thirteenth Annual Conference on Auditory Display*.

Deutsch, D., Dooley, K., & Henthorn, T. (2008). Pitch circularity from tones comprising full harmonic series. *The Journal of the Acoustical Society of America*, *124*(1), 589. doi:10.1121/1.2931957

Donmez, B., Cummings, M. L., & Graham, H. D. (2009). Auditory decision aiding in supervisory control of multiple unmanned aerial vehicles. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *51*(5), 718–729. doi:10.1177/0018720809347106

Edworthy, J., Hellier, E., Titchener, K., Naweed, A., & Roels, R. (2011). Heterogeneity in auditory alarm sets makes them easier to learn. *International Journal of Industrial Ergonomics*, *41*(2), 136–146. doi:10.1016/j.ergon.2010.12.004

Flowers, J. H. (2005). Thirteen years of reflection on auditory graphing: Promises, pitfalls, and potential new directions. *Faculty Publications, Department of Psychology*, 430.

Giang, W., Santhakumaran, S., Masnavi, E., Glussich, D., Kline, J., Chui, F., … Zelek, J. (2010). *Multimodal interfaces: Literature review of ecological interface design, multimodal perception and attention, and intelligent adaptive multimodal interfaces*. DTIC Document.

Hart, S. G., & Staveland, L. E. (1988). Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. *Human mental workload*, *1*(3), 139–183.

Hermann, T., & Hunt, A. (2011). *The sonification handbook*. Berlin: Logos Verlag.

Koch, K., McLean, J., Segev, R., Freed, M. A., Berry, M. J., Balasubramanian, V., & Sterling, P. (2006). How much the eye tells the brain. *Current Biology*, *16*(14), 1428–1434. doi:10.1016/j.cub.2006.05.056

Kramer, G. (1994). *Auditory display : sonification, audification, and auditory interfaces*. Reading, Mass.: Addison-Wesley.

Kramer, G., Walker, B., Bonebright, T., Cook, P., Flowers, J., Miner, N., & Neuhoff, J. (2010). Sonification report: Status of the field and research agenda. *Faculty Publications, Department of Psychology*.

Lachter, J., Forster, K. I., & Ruthruff, E. (2004). Forty-five years after broadbent (1958): Still no identification without attention. *Psychological Review*, *111*(4), 880–913. doi:10.1037/0033-295X.111.4.880

Lehrl, S., & Fischer, B. (1985). Der maximale zentrale Informationsfluss bei Küpfmüller und Frank: beträgt er 50 bit/s oder 16 bit/s? *Grundlagenstudien aus Kybernetik und Geistenswissenschaft / Humankybernetik*, *26*, 147–154.

Loy, G. (2007). *Musimathics, Volume 1: The Mathematical Foundations of Music*. Cambridge, MA, USA: MIT Press.

Matheson, A., Donmez, B., Rehmatullah, F., Jasiobedzki, P., Ng, H.-K., Panwar, V., & Li, M. (in press). The effects of predictive displays on performance in driving tasks with multi-second latency: Aiding tele-operation of lunar rovers. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*.

Moussa, A. S. (2006). Perception-based microtuning over MIDI networks. *MultiMedia, IEEE*, *13*(1), 56–64.

Neuhoff, J. G., Knight, R., & Wayand, J. (2002). Pitch change, sonification, and musical expertise: Which way is up. In *Proceedings of the International Conference on Auditory Display* (pp. 351–356).

Neuhoff, J. G., Wayand, J., & Kramer, G. (2002). Pitch and loudness interact in auditory displays: Can the data get lost in the map? *Journal of Experimental Psychology: Applied*, *8*(1), 17–25. doi:10.1037//1076-898X.8.1.17

Oliver, B. M., Pierce, J. R., & Shannon, C. E. (1948). The philosophy of PCM. *Proceedings of the IRE*, *36*(11), 1324–1331.

Olson, H. F. (1967). *Music, Physics and Engineering*. Mineola, NY, USA: Dover Publications, Incorporated.

Otero, A., Félix, P., Barro, S., & Palacios, F. (2009). Addressing the flaws of current critical alarms: A fuzzy constraint satisfaction approach. *Artificial Intelligence in Medicine*, *47*(3), 219–238. doi:10.1016/j.artmed.2009.08.002

Schomaker, L., Nijtmans, J., Camurri, A., Lavagetto, F., Morasso, P., Robert-Ribes, J., … RUB, J. B. (1995). A taxonomy of multimodal interaction in the human information processing system. *ESPRIT BRA No. 8579, Multimodal Integration for Advanced Multimedia Interfaces (MIAMI)*.

Shannon, C. E., & Weaver, W. (1949). *The Mathematical Theory of Communication.* Urbana, University of Illinois Press.

Smith, D. R., & Walker, B. N. (2002). Tick-marks, axes, and labels: The effects of adding context to auditory graphs. In *International Conference on Auditory Display, Kyoto, Japan*.

Smith, Daniel R., & Walker, B. N. (2005). Effects of auditory context cues and training on performance of a point estimation sonification task. *Applied Cognitive Psychology*, *19*(8), 1065–1087. doi:10.1002/acp.1146

Spain, R. D., & Bliss, J. P. (2008). The effect of sonification display pulse rate and reliability on operator trust and perceived workload during a simulated patient monitoring task. *Ergonomics*, *51*(9), 1320–1337. doi:10.1080/00140130802120234

Ueda, K., Nakajima, Y., Doumoto, K., Ellermeier, W., & Kattner, F. (2013). Disruptive effect of unattended noise-vocoded speech on recall of visually presented digits: Interaction between the number of frequency bands and languages. In *ICA 2013 Montreal* (Vol. 19, p. 060168). Montreal, Canada: ASA. doi:10.1121/1.4800480

Walker, B. N., & Kramer, G. (2004). Ecological psychoacoustics and auditory displays: Hearing, grouping, and meaning making. *Ecological psychoacoustics*, 150–175.

Walker, B. N., & Nees, M. A. (2005). An agenda for research and development of multimodal graphs. In *International Conference on Auditory Display (ICAD2005), Limerick, Ireland*.

Wang, J., Lewis, M., & Hughes, S. (2004). Gravity-referenced attitude display for teleoperation of mobile robots. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* (Vol. 48, pp. 2662–2666).

Watson, M. O., & Sanderson, P. M. (2004). Sonification supports eyes-free respiratory monitoring and task time-sharing. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *46*(3), 497–517.

Watson, M. O., & Sanderson, P. M. (2007). Designing for attention with sound: Challenges and extensions to ecological interface design. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *49*(2), 331–346.

Watson, M., Sanderson, P., & John Russell, W. (2004). Tailoring reveals information requirements: the case of anaesthesia alarms. *Interacting with Computers*, *16*(2), 271–293. doi:10.1016/j.intcom.2003.12.002

Watson, M., & Sanderson, P. M. (2001). Intelligibility of sonifications for respiratory monitoring in anesthesia. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* (Vol. 45, pp. 1293–1297).

Wenzel, E. M., Arruda, M., Kistler, D. J., & Wightman, F. L. (1993). Localization using nonindividualized head-related transfer functions. *The Journal of the Acoustical Society of America*, *94*(1), 111–123. doi:10.1121/1.407089

Wickens, C. D. (1984). Processing resources in attention. In *Varieties of Attention*. New York, NY, USA: Academic Press.

Wickens, C. D., & Hollands, J. G. (1999). *Engineering Psychology and Human Performance* (Third Edition.). Upper Saddle River, NJ, USA: Prentice-Hall.

Appendix A – LELR Design Specification Extract

[EC-LMR-PRF-110]

Side Slope: The Lunar Exploration Light Rover shall drive across slopes with a tilt angle of 25 degrees. Performance may be reduced (e.g. speed) but safety (human and equipment) must not be compromised.

Comment: Note that this requirement is not intended to be applied for low-friction surfaces such as concrete, which would require rubber for traction.

Rationale: Given a standard vehicle layout, the transverse stability is likely to be lower than the longitudinal stability, so the requirement is lowered compared to the maximum gradient. The Rollover requirement covers both cases for safety.

1. Performance: There shall be no stalling, slipping, overheating, upsetting or hesitation.

2. No Leaks: There shall be no leaks of fuel, lubricants or coolants.

3. Shear limit: The requirement shall be met on slopes where the angle of internal shearing resistance exceeds 36°.

[EC-LMR-PRF-064]

Rollover Threshold: The rollover threshold of the Lunar Exploration Light Rover shall be at least 36.9° (0.75 g) when measured in accordance with SAE J2180. Equipment and Payloads will be in a stowed configuration for travelling, but their Center of Mass shall be permitted to be at least as high as the center of the Cargo volume, and anywhere horizontally on the surface of the interface plates controlled by RD-2.

Rationale: This will provide an adequate margin between the operation-on-side-slopes capability of 25° specified previously, and actually rolling over.

Appendix B – Auditory Display and Simulator Telemetry Program

The Java program for generating all auditory display components, collecting certain rover telemetry data, and generating other descriptive statistics about rover driving consisted of one large, primary class and one small, secondary class. The source code for both classes appears below.

Auditory Display and Simulator Telemetry Program – Primary Java Class Source Code

```
import java.lang.*;
import java.io.*;
import java.net.*;
import java.util.Hashtable;
import javax.sound.midi.*;
import java.util.concurrent.Callable;
import java.util.concurrent.ExecutionException;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import java.util.concurrent.Future;
import java.util.Scanner;
import java.lang.Math;
/**-----imports for PCM sound (using JSyn synthesizer)-----*/
/**---Note that importing x.* does not import x.y.* !!!----*/
/**----------------------------------------by Adrian-----*/
import com.jsyn.JSyn;
import com.jsyn.Synthesizer;
import com.jsyn.unitgen.*;
/**-----import for C2SM log file processing-----*/
import javax.swing.JFileChooser;
import javax.swing.filechooser.FileNameExtensionFilter;
import javax.swing.text.DefaultEditorKit.InsertBreakAction;
import javax.swing.JFrame;
import java.io.FileReader;
//import ??? ...JDialog;
// Will Eclipse automatically add an import for com.jsyn.unitgen.Add here?
// If not, I may need to do it.
// (added later)  Eclipse added the import statement after I clicked on the
//                  error message on the line referring to com...Add and Add, and
//                  selected the import option from among (4) different options.

/*****************************************************
 * Client.java            Author: Bill (Mufan) Li        *
 *                         Modified by: Faizan         *
```

```
 *                         Date: September 7th, 2012    *
 *                                                     *
 *         This program is to be used with simulator    *
 * C2SMScoutGUI.exe file. Taking in a stream of        *
 * data from the simulator, this program will be    *
 * responsible to create corresponding MIDI sounds    *
 * from the Java Synthesizer library.                *
 *                                                     *
 * Key Variables:                                    *
 * numbers - hash table storing data for multi-        *
 *              thread communications                    *
 * instrument - selects from MIDI sound library        *
 * data_num - decides which variable to use as        *
 *               the input (ie. speed, tilt)            *
 *         * see end of file for list of variables        *
 * data_max - the maximum float value expect in        *
 *               the input variable                    *
 * data_min - the minimum...                        *
 * step_num - the number of different notes played    *
 *               in the output (ie. value of 24 will    *
 *               play 24 different notes from min to    *
 *               max)                                    *
 ****************************************************

 ****************************************************
 * Command format:                                    *
 * - Instrument change "i XXX"                        *
 *         - 'i' being the English character 'i'        *
 *         - "XXX" being any integer from 1-234        *
 *     *see list of instruments at the end of file        *
 *                                                     *
 * - Data change "d data# step# max min"            *
 *         - 'd' being the English character 'd'        *
 *         - "data#" being an integer 0-9 to select    *
 *            the input variable (ie. velocity)        *
 *         - "step#" gives the number of steps            *
 *         - "max" and "min" are data_max and data_min    *
 *     *see list of variables at the end of file        *
 ****************************************************/

class Client {

    //constants (all ending in "_C" for easy searches)

    // A penalty factor of 3 means that time spent above the threshold will be
    // **counted a total of 3 times**, or, otherwise stated, **2 extra times**
    static int PENALTY_FACTOR_C = 3;

    // in radians (set to pi/6 for 30 degrees) -- Adrian
    static double INCLINATION_THRESHOLD_C = java.lang.Math.PI / 6;
    //static double INCLINATION_THRESHOLD_C = java.lang.Math.PI / 30;
        // 2nd line to test (6 degrees).
        // deactivate 2nd line and activate 1st unless testing!


    static float LOWEST_FREQUENCY_C = (float) 440;
```

```
static float OCTAVES_SPANNED_C = (float) 1;
        // if 1 octave is spanned,
        // then the highest frequency is = 2 * lowestFrequencyCONSTANT
/* This variable is no longer required.  Woot smooth mapping
static int frequencyStepsCONSTANT = 97;
*/

//other class variables (updatable by multiple callings of a method)
//(Can a 2nd calling of a method "pick up where the last one left off"
// if I used a method variable?  I do not think so.  ...unless 'static'?)
// Do I really have to type "(float)" when I already have "float"?  silly.
static float secondsOverMaxInclination = (float) 0; //TODO use


// needed for my kludge of a debugging technique
// (needed because I don't know how to debug in Eclipse)
static String progLocJustPreAnyCrash = "";

static boolean firstTimeGettingTimeStamp = true;
static String firstTimeStampHopefullySameAsLogStamp = "";


// for data collection from log files after experiment
static String sourceProgramOrLog =
        "not yet set to C2SM 'program' or 'log' file";



public static void main(String args[]) {
    //constants (in my own invented notation, to avoid underscores (why?))
    //Is it impossible to use a class variable in a method?!
    //Apparently yes, when it involves a static method (including main),
    //so let's try to put this outside the 'main' method.
    //float lowestFrequencyCONSTANT = (float) 100;

    //set up variables for synthesizer
    int instrument = 0;
    int tempinst = instrument;
    int note = -1;        // Actions on 'freq' for PCM sound will parallel
    float freq = -1;//f?    // ..those on 'note' for MIDI sound.  --Adrian
    float volSonification = 0.2f;//f?..yes //vary w 'freq'? (linear maybe)
    float volAlarm = (float) 0.5;//need "f" or "(float)" **if decimal**
    int timbre = 0;
    int force = 100;

    //set up variables of type double
    double inclinRad_temp_d = -1;
    double inclinRad_last_d = -1; // needed to find threshold crossings
    double rollRad_temp_d = -1;
    double pitchRad_temp_d = -1;
    double a = 0;
    double b = 0;
    double alarmCentreFreq = 0;
    double alarmWaverFreq = 0;
    double refCentreFreq = 0;
    double refOnOffFreq = 0;
```

```
    //set up variables for main
    float inclinRad_temp_f= -1;
    //float temp1 = -1;  //no longer required (maybe never was)
    //float temp2 = -1;  //no longer required (maybe never was)
    String simDataLine = "";
    String [] simDataArray = new String[10];
    String [] modeDescriptArray = new String[10];
    String [] modeVoiceDataRefPercexp = new String[10];
    String tempSimDatum = "";
    float last = -1;
    String inst = "0";

    //int data_num = 4; // 9 = speed, 6 = roll
    int data_num_pitch = 4; // actually a constant...should move and add _C
    int data_num_roll = 6; // actually a constant...should move and add _C

    //needed to track overall time spent above inclination threshold
    int data_num_timestamp = 0; // actually a constant...should move and _C
    boolean aboveThresholdLastLoop = false;
    int aboveThresholdForayStartTimePoint_ms = 0;
    int aboveThresholdForayFinishTimePoint_ms = -1;
    int aboveThresholdCumulativeTimeSpan_ms = 0;

    //used to get course completion time span (easier than using text logs!)
    int data_num_z_distance = 3; // actually a constant...should move and add _C
    int courseStartTimePoint_ms = 0;
    int courseFinishTimePoint_ms = -1;
    double zDist_temp_d = 1000000;
    double zDist_last_d = 1000001;

    // in milliseconds since beginning of day
    // (ASSUMPTION:  simulator will not be run over midnight!)
    int simDataTimePoint_ms_into_today = 0;  //TODO use

    float data_max = 6;
    float data_min = 0;
    int step_num = 24;

    /**---variables (objects?) for PCM sound (using JSyn synthesizer)---*/
    /**----------------------------------------------------by Adrian---*/
    /*com.jsyn.unitgen.SineOscillator myOsc;    // a "unit"
    com.jsyn.unitgen.LineOut myOut;            // a "unit"
    */    //commented out here because of
        //"cannot make a static reference to the non-static method
        //______() from the type Client" (and "cannot be resolved") errors
        //What is going on?!  What does 'static' mean?  Can I change
        //the main method to get rid of 'static' and would that fix things??
//    com.jsyn.Synthesizer synthPCMSonification = JSyn.createSynthesizer();
//    com.jsyn.Synthesizer synthPCMAlarm = JSyn.createSynthesizer();
                                // "an instance of Synthesizer"
    com.jsyn.Synthesizer synthPCM = JSyn.createSynthesizer();
                                // an instance of Synthesizer
    com.jsyn.unitgen.SineOscillator oscData = new SineOscillator();
```

```
//    PulseOscillator oscRefOnOffEnvelope = new PulseOscillator();
//    SineOscillator oscRefOnOffEnvelope = new SineOscillator();
    SquareOscillator oscRefOnOffEnvelope = new SquareOscillator();
    SineOscillator oscRefComplete = new SineOscillator();


//    SineOscillator oscAlarmWaverEnvelope = new SineOscillator();
    SawtoothOscillator oscAlarmWaverEnvelope = new SawtoothOscillator();
//    TriangleOscillator oscAlarmWaverEnvelope = new TriangleOscillator();
//    SquareOscillator oscAlarmWaverEnvelope = new SquareOscillator();
//    ImpulseOscillator oscAlarmWaverEnvelope = new ImpulseOscillator();
//    PulseOscillator oscAlarmWaverEnvelope = new PulseOscillator();
//    Latch oscAlarmWaverEnvelope = new Latch();
//    FunctionOscillator oscAlarmWaverEnvelope = new FunctionOscillator();

    SineOscillator oscAlarmComplete = new SineOscillator();
                            // "a unit"
    com.jsyn.unitgen.LineOut oscsLineOut = new LineOut();
                            // "a unit"

    /* This code block moved until after PCM sound start, so can test that.
    // Implement a prompt to determine the sonification
    // mode for the program
    Scanner user_input = new Scanner (System.in);
    int mode;
    System.out.print("Select the sonification mode: ");
    mode = user_input.nextInt();
    */

    // Test to print the command prompt
    //System.out.println(mode);

    //buffer for input
    // variable name changed from "br" to "bufferedReader"
    // here and in SimpleThreads.java for readability --Adrian
    BufferedReader bufferedReader
            = new BufferedReader(new InputStreamReader(System.in));

    //set up hash table
    Hashtable numbers = new Hashtable();
    numbers.put("inst", new Integer(instrument));

    //numbers.put("num", new Integer(data_num));
    numbers.put("indexPitch", new Integer(data_num_pitch));
    numbers.put("indexRoll", new Integer(data_num_roll));

    numbers.put("max", new Float(data_max));
    numbers.put("min", new Float(data_min));
    numbers.put("step", new Integer(step_num));

    //for timing tests
    long start = 0;
    long stop = 0;

    /***********************/
    /** setup MIDI synthesizer **/
```

```
        /**********************/
        javax.sound.midi.Synthesizer synthMIDI = null;
        try {
            synthMIDI = javax.sound.midi.MidiSystem.getSynthesizer();
            synthMIDI.open();
        }
        catch (Exception e) {
            System.out.println(e);
        }

        Soundbank soundbank = synthMIDI.getDefaultSoundbank();
        Instrument[] instr = soundbank.getInstruments();
        synthMIDI.loadInstrument(instr[instrument]);
        MidiChannel[] mc = synthMIDI.getChannels();
        /**--added to see what the different MIDI channels are -- Adrian--*/
        System.out.println("The MIDI channel array length is: " + mc.length);
        System.out.println("Midi channel [4] is: " + mc[4]);

        mc[4].programChange(0, instrument);
        //finish setup synthesizer

        progLocJustPreAnyCrash = "finished setting up MIDI synthesizer";

        /**-----------------------------------------------------*/
        /**-----start-up for PCM sound (using JSyn synthesizer)-----*/
        /**-------------------------------------------by Adrian-----*/

        // startSynthesisEngine();    // generates errors
        /** from gutted method startSynthesisEngine()
         *  ...not sure why needed to move*/
    //    synthPCMSonification.start();
    //    synthPCMAlarm.start();
        synthPCM.start();

        // buildUnitGenerators();    // "cannot make static reference
        /** from gutted method buildUnitGenerators()
         *  ...not sure why needed to move*/
        //synthPCM.add(myOsc = new SineOscillator());
    //    synthPCMSonification.add(oscData);
//        synthPCMAlarm.add(oscAlarmWaverEnvelope); //TODO:  need this line??
    //    synthPCMAlarm.add(oscAlarmComplete);
        synthPCM.add(oscData);
        synthPCM.add(oscAlarmWaverEnvelope); //TODO:  need this line??
        synthPCM.add(oscAlarmComplete);
        synthPCM.add(oscRefOnOffEnvelope);
        synthPCM.add(oscRefComplete);


        //synthPCM.add(myOut = new LineOut());
    //    synthPCMSonification.add(oscsLineOut);
    //    synthPCMAlarm.add(oscsLineOut);
        synthPCM.add(oscsLineOut);

        oscData.frequency.set(LOWEST_FREQUENCY_C);  // 440 Hz  //err "cannot be
resolved"
```

```
/*sample JSyn code:
    AddUnit  freqAdder  = new AddUnit();
    sineOsc1.output.connect( freqAdder.inputA );
            // pass through adder
    freqAdder.output.connect( sineOsc2.frequency );
            // control second oscillator freq
    freqAdder.inputB.set( 500.0 );
            // add constant that will center us at 500 Hz
    sineOsc1.amplitude.set( 100.0 );
            // reduce offset to +/- 100 Hz
    //Thus the frequency of sineOsc2 will be sineOsc1.output plus inputB
  */

//create a frequency adder for a siren-like alarm
com.jsyn.unitgen.Add oscAlarmFreqAdder = new Add(); //used to be AddUnit

//set the alarm centre frequency
alarmCentreFreq = (LOWEST_FREQUENCY_C
                   * Math.pow(2, OCTAVES_SPANNED_C + 1));
                  //This formula centres the alarm one octave
                  //above the threshold's sonification freqency
alarmWaverFreq = alarmCentreFreq / 10;
                 //This sets the waver at one tenth of the centre freq
                 //Unfortunately, the waver appears to need to be the
                 //same amount above and below the centre
                 //(linear, vice perceptually-linear (quadratic))
System.out.println(alarmCentreFreq + "-Hz alarm centre frequency");
oscAlarmFreqAdder.inputB.set(alarmCentreFreq);

//set the alarm waver envelope
        //(alarm will range between centre-waver and centre+waver)
oscAlarmWaverEnvelope.amplitude.set(alarmCentreFreq / 10);
oscAlarmWaverEnvelope.frequency.set(4.0);

//"pass through adder" (??)
oscAlarmWaverEnvelope.output.connect(oscAlarmFreqAdder.inputA);
//(entered this with by starting to type, then hitting [Ctrl]+[Space]!)

//"control the 2nd oscillator frequency" (?)
oscAlarmFreqAdder.output.connect(oscAlarmComplete.frequency);




//create a frequency adder for an intermittent (on-off) reference tone
Add oscRefFreqAdder = new Add();

//set the reference tone centre frequency
refCentreFreq = (LOWEST_FREQUENCY_C
                 * Math.pow(2, OCTAVES_SPANNED_C));
                //This formula centres the reference tone
                //at the threshold's sonification freqency
refOnOffFreq = 0.5;
                 //This sets the on-off frequency
System.out.println(refCentreFreq + "-Hz ref tone centre frequency");
```

```
oscRefFreqAdder.inputB.set(refCentreFreq/2);

//set the alarm waver envelope
        //(alarm will range between centre-waver and centre+waver)
oscRefOnOffEnvelope.amplitude.set(refCentreFreq/2);
oscRefOnOffEnvelope.frequency.set(refOnOffFreq);

//"pass through adder" (??)
oscRefOnOffEnvelope.output.connect(oscRefFreqAdder.inputA);
//(entered this with by starting to type, then hitting [Ctrl]+[Space]!)

//"control the 2nd oscillator frequency" (?)
oscRefFreqAdder.output.connect(oscRefComplete.frequency);





oscData.amplitude.set(0);

//oscRefComplete.amplitude.set(volSonification);
oscRefComplete.amplitude.set(0);

//set alarm volume
//(maybe 0.7 is default?)
//maybe should be 0.5, to avoid clipping from adding??
oscAlarmComplete.amplitude.set(0);





// actually, http://www.softsynth.com/jsyn/docs/javadocs/
// says default amplitude is 0.999969482421875

// connectUnitGenerators();    // to the non-static
/** from gutted method connectUnitGenerators()
 *  ...not sure why needed to move*/
// connect oscillator to both channels of stereo player
oscData.output.connect(0, oscsLineOut.input, 0);
oscData.output.connect(0, oscsLineOut.input, 1);

oscAlarmComplete.output.connect(0, oscsLineOut.input, 0);
oscAlarmComplete.output.connect(0, oscsLineOut.input, 1);

oscRefComplete.output.connect(0, oscsLineOut.input, 0);
oscRefComplete.output.connect(0, oscsLineOut.input, 1);

// startUnitGenerators();    // method ________"
/** from gutted method startUnitGenerators()
 *  ...not sure why needed to move*/
// start execution of units.  JSyn 'pulls' data so the only unit
// you have to start() is the last one, in this case our LineOut
oscsLineOut.start();

progLocJustPreAnyCrash = "finished setting up PCM synthesizer";
```

```
/**-------------------------Audio Feedback Modes------------------------
 * 1. voice;    alerts;                -            -
 * 2. voice;    alerts, soni;        data;        quadratic (basic) //y?
 * 3. voice;    alerts, soni;        data;        transformed            //y?
 * 4. voice;    alerts, soni;        data, ref;    quadratic (basic)
 * 5. voice;    alerts, soni;        data, ref;    transformed
 * 6. no voice;    alerts                -            -
 * 7. no voice; alerts, soni;        data;        quadratic (basic) //y?
 * 8. no voice; alerts, soni;        data;        transformed            //y?
 * 9. no voice; alerts, soni;        data; ref;    quadratic (basic)
 * 0. no voice; alerts, soni;        data; ref;    transformed
 */

modeDescriptArray[1] = "1     voice;  no soni;           ;            ;";
modeDescriptArray[2] = "2     voice;     soni;  data     ;  percepLin;";
modeDescriptArray[3] = "3     voice;     soni;  data     ;  percepExp;";
modeDescriptArray[4] = "4     voice;     soni;  data, ref;  percepLin;";
modeDescriptArray[5] = "5     voice;     soni;  data, ref;  percepExp;";
modeDescriptArray[6] = "6  no voice;  no soni;           ;            ;";
modeDescriptArray[7] = "7  no voice;     soni;  data     ;  percepLin;";
modeDescriptArray[8] = "8  no voice;     soni;  data     ;  percepExp;";
modeDescriptArray[9] = "9  no voice;     soni;  data, ref;  percepLin;";
modeDescriptArray[0] = "10 no voice;     soni;  data, ref;  percepExp;";

modeVoiceDataRefPercexp[1] = "v1,d0,t0,m2";
modeVoiceDataRefPercexp[2] = "v1,d1,t0,m0";
modeVoiceDataRefPercexp[3] = "v1,d1,t0,m1";
modeVoiceDataRefPercexp[4] = "v1,d1,t1,m0";
modeVoiceDataRefPercexp[5] = "v1,d1,t1,m1";
modeVoiceDataRefPercexp[6] = "v0,d0,t0,m2";
modeVoiceDataRefPercexp[7] = "v0,d1,t0,m0";
modeVoiceDataRefPercexp[8] = "v0,d1,t0,m1";
modeVoiceDataRefPercexp[9] = "v0,d1,t1,m0";
modeVoiceDataRefPercexp[0] = "v0,d1,t1,m1";


//Thread.sleep(1000);  // TODO: Find out why this line needed to change
                       //       to the below line to work.
try {
    Thread.sleep(1000);
} catch(InterruptedException ex) {
    Thread.currentThread().interrupt();
}

System.out.println("");
for (int i = 1;  i < 11;  i++) {
    System.out.print(i + ":  ");
    System.out.println(modeDescriptArray[i % 10]);
}

// Implement a prompt to determine the sonification
// mode for the program
Scanner user_input = new Scanner (System.in);
```

```
        //for data collection from log files after experiment
        System.out.print("Read from C2SM 'p'rogram or 'l'og file?");
        sourceProgramOrLog = user_input.nextLine();
        System.out.println(sourceProgramOrLog);

    //    Thread.sleep(2000); // Do I need this to make sure properly processed?

        //if (sourceProgramOrLog == "l"){// this way apparently compares address
        if (sourceProgramOrLog.equals("l")) {


            final JFrame jFrame;
            jFrame = new JFrame();
            jFrame.setVisible(true);
            jFrame.setExtendedState(JFrame.ICONIFIED);
            jFrame.setExtendedState(JFrame.NORMAL);

            simDataArray[1] = simDataArray[2];

            String currentDirectoryFolderPath =
                    "H:\\LELR_roll_sonification_program";
            javax.swing.JFileChooser jFileChooser =
                    new JFileChooser(currentDirectoryFolderPath);
            //jFileChooser.setVisible(true); //defaults to invisible?!?

//this for container custom dialog? -->        jFileChooser.setAlwaysOnTop();

            javax.swing.filechooser.FileNameExtensionFilter fileExtensionFilter
                    = new FileNameExtensionFilter(
                            "comma-separated values and text files",
                            "csv", "txt");
            jFileChooser.setFileFilter(fileExtensionFilter);
//'parent'?    int returnVal = jFileChooser.showOpenDialog(parent);
//notwork    int returnVal = jFileChooser.showOpenDialog(jFileChooser);
            //jFileChooser.showDialog(null, "testing 1--2--3");
            //jFileChooser.requestFocusInWindow();

            //noprint    System.out.println(jFileChooser.requestFocus());
            //System.out.println(jFileChooser.requestFocusInWindow());
            //System.out.println(jFileChooser.requestDefaultFocus());

            //jFileChooser.showOpenDialog(jFileChooser);
            //jFileChooser.requestFocus();
            int returnVal = jFileChooser.showOpenDialog(null);
            System.out.println("returnVal = "
                               + "jFileChooser.showOpenDialog(jFileChooser) = "
                               + returnVal);
            System.out.println("JFileChooser.APPROVE_OPTION = "
                               + JFileChooser.APPROVE_OPTION);
            if (returnVal == JFileChooser.APPROVE_OPTION) {
                System.out.println("You chose to open this file: " +
                        jFileChooser.getSelectedFile().getName());
            } else {
```

```
        System.out.println("No file was selected.");
    }


    System.out.println(JFileChooser.APPROVE_OPTION);
    System.out.println(jFileChooser);


    jFrame.setVisible(false);

    /**
    FileReader fileReader; //Eclipse(Java) insists this be
                           //outside the try-catch block
                           //even though the rest is (must be) inside.
    try { //this try-catch block auto-filled
          //(Eclipse suggested this or "throws exception..."
          // in class line)
        fileReader =
                new FileReader(jFileChooser.getSelectedFile());
    } catch (FileNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    BufferedReader bufferedReaderLogFileReader =
            new BufferedReader(fileReader);
    System.out.println(fileReader);
    System.out.println(bufferedReaderLogFileReader);
    */
}
else {
    System.out.println("sourceProgramOrLog, '" + sourceProgramOrLog
                       + "' judged as not equal to 'l'");
}


String subject;
System.out.print("Select the participant letter: ");
subject = user_input.nextLine();
System.out.println("Participant " + subject + " selected.");

int mode;
System.out.print("Select the sonification mode number: ");
mode = user_input.nextInt();
System.out.println("Soundscape Mode " + modeDescriptArray[mode % 10]
                   + " selected.");


//turn on data tone if applicable
if ((mode != 1) && (mode != 6)) {
    oscData.amplitude.set(volSonification);
}

try {
    Thread.sleep(2000);
} catch(InterruptedException ex) {
    Thread.currentThread().interrupt();
```

```
}

// turn on reference tone if applicable
if ((mode == 4) || (mode == 5) || (mode == 9) || (mode == 10)) {
    oscRefComplete.amplitude.set(volSonification);
}

try {
    Thread.sleep(4000);
} catch(InterruptedException ex) {
    Thread.currentThread().interrupt();
}

// turn on alarm temporarily (demonstration)
oscAlarmComplete.amplitude.set(volAlarm);
try {
    Thread.sleep(1000);
} catch(InterruptedException ex) {
    Thread.currentThread().interrupt();
}
oscAlarmComplete.amplitude.set(0);


/******************/
/** setup socket **/
/******************/
try {
    Socket skt = new Socket("127.0.0.1", 17112);

    BufferedReader in = new BufferedReader(new
            InputStreamReader(skt.getInputStream()));


    /******************/
    /** setup thread **/
    /******************/
    //takes in input of the input buffer and the hash table
    Runnable task = new SimpleThreads(bufferedReader, numbers);
    Thread worker = new Thread(task);


    /***************/
    /** main loop **/
    /***************/
    while (true){

        /*********************************/
        /** check if new thread needed **/
        /*********************************/
        if ( instrument != (int) numbers.get("inst") ||

                //data_num != (int) numbers.get("num") ||
                data_num_pitch != (int) numbers.get("indexPitch") ||
                data_num_roll != (int) numbers.get("indexRoll") ||

                data_max != (float) numbers.get("max") ||
```

```
        data_min != (float) numbers.get("min") ||
        step_num != (int) numbers.get("step"))

{
    instrument = (int) numbers.get("inst");
    instrument = tempinst;
    mc[4].programChange(0, instrument);

    //data_num = (int) numbers.get("num");
    data_num_pitch = (int) numbers.get("indexPitch");
    data_num_roll = (int) numbers.get("indexRoll");
            /* This line had an error (though it did not seem
             * to affect the program):  The argument was
             * "roll", while it was supposed to be "num2".
             * It has now been changed to another name, along
             * with all instances of "num2". --Adrian */

    data_max = (float) numbers.get("max");
    data_min = (float) numbers.get("min");
    step_num = (int) numbers.get("step");

    task = new SimpleThreads(bufferedReader, numbers);
    worker = new Thread(task);
}
/*** ending check ***/


// if there is no thread working, start a new thread
if (!worker.isAlive()) {
    worker.start();
}

// wait here to read new data every 0.25 seconds
/*I think Bill means (above) that the data happens to come in
 * roughly every 0.25 seconds, not that the program waits.
 * --Adrian */
simDataLine = in.readLine();

// started having problems with simDataLine being null, but
// could not figure out.  ...possibly memory problem?
//                          ...or is it my fault, with something
//                             I programmed??
if (simDataLine == null) {
    System.out.println("simDataLine was null.  "
                       + "memory problem?  loop 'continued'..."
                       + "which *actually* means 'skipped'(!)");
}

// comment lines are skipped
if (simDataLine.charAt(0) == '#')
    continue;

//xxxdeactivate or activate (comment in or 'comment out'??) here
System.out.println("simDataLine: " + simDataLine);
        // Read 1 line and output it
//*/
```

```
// split every line into an array of strings
simDataArray = simDataLine.split(" ");


/* //activate-deactivate (comment out and...'comment in'?) here
// I want to see whether the ".trim" used later is necessary
// performing .split on a string with single (or even multiple)
// space delimiter should leave array with no delimiters, yes?
// (added later) Answer:  yes, there are no delimiters or spaces
for (int i = 0; i < simDataArray.length; i++) {
    System.out.print("[" + i + "]" + simDataArray[i]);
}
System.out.println(" end of array");
// */

progLocJustPreAnyCrash = "finished checking for '#' in data";

/**---- Assign last loop's 'inclinRad_temp_d' (inclination)
 * value to 'inclinRad_last_d' (before it actually gets
 * assigned this loop's proper inclinRad_temp_d value).
 * This is needed for determining, later in this loop, whether
 * there is a **transition to** above or within the threshold.
 */
inclinRad_last_d = inclinRad_temp_d;

progLocJustPreAnyCrash = "finished setting new inclin to old";

/**---- Assign last loop's 'z' (inclination)
 * value to 'inclinRad_last_d' (before it actually gets
 * assigned this loop's proper inclinRad_temp_d value).
 * This is needed for determining, soon in this loop, whether
 * there is a **transition to** being in or out of the course.
 */
zDist_last_d = zDist_temp_d;

progLocJustPreAnyCrash = "finished setting new z to old";

/**---- Test to see if starting or finishing course. ----
 * ---- Save course start or course finish time points. ----
 */



if (firstTimeGettingTimeStamp == true) {
    firstTimeStampHopefullySameAsLogStamp =
            simDataArray[0];
    System.out.println(firstTimeStampHopefullySameAsLogStamp);
    firstTimeGettingTimeStamp = false;
}
```

```
// Take the z data
tempSimDatum = simDataArray[data_num_z_distance];

progLocJustPreAnyCrash = "setting tempSimDatum to z distance";
//System.out.println(tempSimDatum);

//convert to double
try {
    progLocJustPreAnyCrash = ("entering z try" + tempSimDatum);
    zDist_temp_d = new Double(tempSimDatum.trim());
    progLocJustPreAnyCrash = "exiting z try";
} catch (NumberFormatException nfe) {
    zDist_temp_d = 1000000;
}
/*/deactivate or activate (comment out or 'comment in'??) here
//see z values
System.out.print("z" + zDist_temp_d + ",");
//*/

if ((zDist_temp_d < 0) && (zDist_last_d > 0)) {
    courseStartTimePoint_ms =
            extract_ms_this_day_from_timestamp(
                    simDataArray[data_num_timestamp]);
    System.out.println();
    System.out.println("course started at "
                       + courseStartTimePoint_ms);
}
if ((zDist_temp_d > 0) && (zDist_last_d < 0)) {
    courseFinishTimePoint_ms =
            extract_ms_this_day_from_timestamp(
                    simDataArray[data_num_timestamp]);
    System.out.println();
    System.out.println("course finished at "
               + courseFinishTimePoint_ms);
}


/*// take only the data wanted
data = tempdata[data_num];

//convert to float
try
{
    temp = new Float(data.trim());
}
catch (NumberFormatException nfe)
{
    temp = -1;
}
*/
```

```
// Take the roll data
tempSimDatum = simDataArray[data_num_roll];

//convert to double
try {
    rollRad_temp_d = new Double(tempSimDatum.trim());
} catch (NumberFormatException nfe) {
    rollRad_temp_d = -1;
}

// Take the pitch data
tempSimDatum = simDataArray[data_num_pitch];

//convert to double
try {
    pitchRad_temp_d = new Double(tempSimDatum.trim());
} catch (NumberFormatException nfe) {
    pitchRad_temp_d = -1;
}


// Changing into absolute values
rollRad_temp_d = Math.abs(rollRad_temp_d);
pitchRad_temp_d = Math.abs(pitchRad_temp_d);

// Calculate the inclination using the equation
// inc = arcos(cos(roll)*cos(pitch))
a = Math.cos(rollRad_temp_d);
b = Math.cos(pitchRad_temp_d);
inclinRad_temp_d = Math.acos(a*b);

/**-------------------------------------------------------------
 * Test if inclination above threshold.
 * ----If yes, test if just transitioned to above.
 * --------If yes, then set aboveThresholdForayStartTimePoint
 * --------and turn on alarm.
 * ----If no, then test if just transitioned to not above.
 * --------If yes, then set aboveThresholdForayFinishTimePoint
 * --------and turn off alarm
 * --------and add this foray's time span to the cumulative one.
 * After all else is done, mark whether above threshold
 * 'last loop' (for reference during test on next loop).
 * -------------------------------------------------------------
 * Note:  alarm "off" is actually sound frequency lowered to
 *        below audibility.  (Otherwise I would have had to
 *        figure out multithreading (+1 Java crash course).)
 * -------------------------------------------------------------
 */

if (inclinRad_temp_d >= INCLINATION_THRESHOLD_C) {
    /*/ deactivate or activate (comment out or 'in'??) code here
    // see inclination, threshold, and whether above last loop
    System.out.println("T, inc" + inclinRad_temp_d + ", thresh"
                       + INCLINATION_THRESHOLD_C
                       + ", above..Last"
                       + aboveThresholdLastLoop);
```

```
    //*/
    if (aboveThresholdLastLoop == false) {
        aboveThresholdForayStartTimePoint_ms =
                extract_ms_this_day_from_timestamp(
                        simDataArray[data_num_timestamp]);
        /*/ deactivate or activate (comment out or 'comment in'?)
        // see newly determined foray start time point
        System.out.print(aboveThresholdForayStartTimePoint_ms
                         + "ms this day at foray start.  ");
        //*/
        oscAlarmComplete.amplitude.set(volAlarm);
        }
    aboveThresholdLastLoop = true;
} else {
    /*/ deactivate or activate (comment out or 'in'??) code here
    // see inclination, threshold, and whether above last loop
    System.out.println("F, inc" + inclinRad_temp_d + ", thresh"
               + INCLINATION_THRESHOLD_C
               + ", above..Last"
               + aboveThresholdLastLoop);
    //*/
    if (aboveThresholdLastLoop == true) {
        aboveThresholdForayFinishTimePoint_ms =
                extract_ms_this_day_from_timestamp(
                        simDataArray[data_num_timestamp]);
        /*/ deactivate or activate (comment out or 'comment in'?)
        // see stored foray start time point
        // see newly determined foray finish time point
        // see old (stored) cumulative time span before adding
        // see new cumulative time span updated w latest foray
        System.out.print(aboveThresholdForayStartTimePoint_ms
                         + "ms this day at foray start.  "
                         + aboveThresholdForayFinishTimePoint_ms
                         + "ms this day at foray finish.  ");
        System.out.println();
        System.out.print(aboveThresholdCumulativeTimeSpan_ms
                           + "ms old cumulative total.");
        //*/
        aboveThresholdCumulativeTimeSpan_ms =
                aboveThresholdCumulativeTimeSpan_ms
                + (aboveThresholdForayFinishTimePoint_ms
                   - aboveThresholdForayStartTimePoint_ms);
        System.out.println(aboveThresholdCumulativeTimeSpan_ms
                   + "ms new cumulative total.");

        oscAlarmComplete.amplitude.set(0);
    }
    aboveThresholdLastLoop = false;
}

// Convert inclinRad_temp_d to float (inclinRad_temp_f)
inclinRad_temp_f= (float) inclinRad_temp_d;

/*/activate-deactivate (comment out and...'comment in'?) here
// Print the inclination
// UNCOMENT THIS IF YOU WANT TO SEE THE INCLINATION
```

```
                // VALUES BEING USED TO GENERATE THE SOUND
                //TODO: See if this produces values I could use instead of logs
                System.out.println("i" + inclinRad_temp_f + ",");
                // */

                // check if it is the same as the last note played
                //Actually, whether the current angle
                // is the same as the last --Adrian
                if (inclinRad_temp_f!= last){
                    if (note != -1){
                        mc[4].noteOff(note);
                        /**Time test code
                        //stop = System.currentTimeMillis();
                        System.out.println("Time: " + (stop - start) + "ms");
                         */
                    }

                    // Finds the new note from the input given
                    // based on the mode in which the program
                    // is currently working

                    if ((mode==2) || (mode==4) || (mode==7) || (mode==9))
                    {
//xMIDI                         note = find_note1(inclinRad_temp_f, numbers);
//xMIDI                         mc[4].noteOn(note, force);

                        freq = find_freq_PCM(inclinRad_temp_f, numbers, mode);
                        oscData.frequency.set(freq);

                        /*// deactivate or activate code here
                        //TODO Vary volume or not (2nd copy of this comment)
                        volSonification =
                              ((freq - find_freq_PCM((float)0, numbers, mode))
                               / (find_freq_PCM((float)INCLINATION_THRESHOLD_C,
                                                numbers, mode)
                                  - find_freq_PCM((float)0, numbers, mode)))
                              * 1;        // <-- maximum volume (1 is highest)
                        oscData.amplitude.set(volSonification);
                        oscRefComplete.amplitude.set(volSonification);
                        //*/
                            //determine proportion of way between
                            // 0-degree sonification frequency and
                            // 30-degree sonification frequency, and set
                            // this proportion as the volume level
                            //question:  IS LINEAR VOLUME CHANGE
                            // PERCEIVED AS LINEAR??
                            //maybe should have volume mapping
                            // always be linear perception scale??
                            //maybe change factor from
                            // 1 to 0.7 to avoid clipping??
                            //maybe should put this code
                            // in one place instead of two?

                        last = inclinRad_temp_f;
                    }
```

```
                    else if ((mode==3) || (mode==5) || (mode==8) || (mode==10))
                    {
//xMIDI                        note = find_note2(inclinRad_temp_f, numbers);
//xMIDI                        mc[4].noteOn(note, force);

                        freq = find_freq_PCM(inclinRad_temp_f, numbers, mode);
                        oscData.frequency.set(freq);

                        /*// deactivate or activate code here
                        //TODO Vary volume or not (2nd copy of this comment)
                        volSonification =
                              ((freq - find_freq_PCM((float)0, numbers, mode))
                               / (find_freq_PCM((float)INCLINATION_THRESHOLD_C,
                                                numbers, mode)
                                  - find_freq_PCM((float)0, numbers, mode)))
                              * 1;        // <-- maximum volume (1 is highest)
                        oscData.amplitude.set(volSonification);
                        oscRefComplete.amplitude.set(volSonification);
                        //*/
                            //determine proportion of way between
                            // 0-degree sonification frequency and
                            // 30-degree sonification frequency, and set
                            // this proportion as the volume level
                            //question:  IS LINEAR VOLUME CHANGE
                            // PERCEIVED AS LINEAR??
                            //maybe should have volume mapping
                            // always be linear perception scale??
                            //maybe change factor from
                            // 1 to 0.7 to avoid clipping??
                            //maybe should put this code
                            // in one place instead of two?

                        last = inclinRad_temp_f;
                    }


/*// deactivate or activate code here
//xMIDI
                    else if (mode == 3)
                    {
                        note = find_note3(inclinRad_temp_f, numbers);
                        mc[4].noteOn(note, force);
                        last = inclinRad_temp_f;
                    }

                    else if (mode == 4)
                    {
                        note = find_note4(inclinRad_temp_f, numbers);
                        mc[4].noteOn(note, force);
                        last = inclinRad_temp_f;
                    }

                    else if (mode == 5)
                    {
                        if (inclinRad_temp_f<= 0.5235){
                            mc[4].noteOff(note, force);
```

```
                            last = inclinRad_temp_f;
                        }
                        else if (inclinRad_temp_f>= 0.5235){
                            note = find_note5(inclinRad_temp_f, numbers);
                            mc[4].noteOn(note, force);
                            last = inclinRad_temp_f;
                        }
                    }

//*/

                    //finds the new note from the input given
                    //note = find_note(temp, numbers);


                    //mc[4].noteOn(note, force);
                    /**Time test code
                    //start = System.currentTimeMillis();
                     */

                    //last = temp;
                } //end if (inclinRad_temp_f != last)

                //finish
                // I do not know why Bill ever expects the string "done"!
                // ...and whether he expects to look for it in the full-line
                // string or an(the, for him) individual datum string!
                // --Adrian
                if (tempSimDatum.equalsIgnoreCase("done")){
                    System.out.println(tempSimDatum); // see if ever happens
                    in.close();
                    mc[4].noteOff(note);
                    break;
                }
            } //end while
        }//end try
        catch(Exception e) {
            System.out.println(e);
            System.out.print("Whoops! It didn't work!\n");
        }

        /**---- from gutted method stop() --Adrian ----*/
    //    if (synthPCMSonification != null) {
    //        synthPCMSonification.stop();
    //        synthPCMSonification = null;
    //    }
    //    if (synthPCMAlarm != null) {
    //        synthPCMAlarm.stop();
    //        synthPCMAlarm = null;
    //    }
        if (synthPCM != null) {
            synthPCM.stop();
            synthPCM = null;
        }

        synthMIDI.close();
```

```
    System.out.println("latest data string: " + simDataLine);
    System.out.println("if program crashed, was shortly after:  "
                       + progLocJustPreAnyCrash);

    //deactivate-activate code here
    // see variables on which course completion time span algorithm relies
    System.out.println("this loop z " + zDist_temp_d
               + ", last loop z" + zDist_last_d);
    System.out.println("course start (ms): " + courseStartTimePoint_ms
                       + "course finish (ms): " + courseFinishTimePoint_ms);
    //*/

    System.out.println("Soundscape Mode " + modeDescriptArray[mode % 10]);

    /**---- Output experimental data (verbose). ----*/
    /*/ deactivate or activate code here
    System.out.print("participant letter, soundscape mode number, ");
    System.out.print("course completion time span, ");
    System.out.print("cumulative time span above threshold, ");
    System.out.print("time span penalty (cumul * penalty), ");
    System.out.println("penalty-adjusted course completion time span");
    System.out.println(subject + "," + modeDescriptArray[mode % 10] + ","
            + (courseFinishTimePoint_ms - courseStartTimePoint_ms)
            + "," + aboveThresholdCumulativeTimeSpan_ms + ","
            + aboveThresholdCumulativeTimeSpan_ms * (PENALTY_FACTOR_C - 1)
            + "," + ((courseFinishTimePoint_ms - courseStartTimePoint_ms)
                     + (aboveThresholdCumulativeTimeSpan_ms
                        * (PENALTY_FACTOR_C - 1))));
    //*/// end deactivate-activate-code block

    /**---- Output experimental data (terse). ----*/
    //*/ deactivate or activate code here
    System.out.print("time stamp, ");
    System.out.print("participant, position (**not prog'd**, manual), ");
    System.out.print("soundscape, ");
    System.out.print("voice(1yes),soniData(1yes),soniThresh(1yes),");
    System.out.println("percepExpon(0lin,1exp,2--), ");
    System.out.print("ms course, ");
    System.out.print("ms @>threshold, ");
//    System.out.print("ms penalty (@>thresh * "+(PENALTY_FACTOR_C-1)+ "), ");
    System.out.println("ms course+penalty(" + (PENALTY_FACTOR_C-1) + "x)");
    System.out.print("num of above-thresh excursions (**not prog'd**), ");
    System.out.print("excursion time span mean (**not prog'd**), ");
    System.out.print("excursion time span standard dev (**not prog'd**), ");
    System.out.print("excursion time span skew (**not prog'd**), ");
    System.out.print("excursion time span kurtosis (**not prog'd**), ");
    System.out.print("excursion time span minimum (**not prog'd**), ");
    System.out.println("excursion time span maximum (**not prog'd**), ");
    System.out.print("num excursions where x>0 (1st ½) (**not prog'd), ");
    System.out.print("num excursions where x<0 (2nd ½) (**not prog'd), ");
    System.out.print("ms @>thresh, where x>0 (1st ½) (**not prog'd), ");
    System.out.print("ms @>thresh, where x<0 (2nd ½) (**not prog'd), ");
    System.out.println(firstTimeStampHopefullySameAsLogStamp + ","
            + subject + ",S" + mode + ",P,"
            + modeVoiceDataRefPercexp[mode % 10]
```

```
            + "," + (courseFinishTimePoint_ms - courseStartTimePoint_ms)
            + "," + aboveThresholdCumulativeTimeSpan_ms
//            + ","+aboveThresholdCumulativeTimeSpan_ms*(PENALTY_FACTOR_C - 1)
            + "," + ((courseFinishTimePoint_ms - courseStartTimePoint_ms)
                     + (aboveThresholdCumulativeTimeSpan_ms
                        * (PENALTY_FACTOR_C - 1)))
            + "[insert num here]"  //TODO (To do calculations for all of
            + "[insert mean here]"  //TODO  these, will need to save details
            + "[insert std dev here]"  //TODO  of each excursion as new
            + "[insert skew here]"  //TODO  element in growing array, and
            + "[insert kurtosis here]"  //TODO  either do ongoing calcs
            + "[insert minimum here]"  //TODO  saved to vars or just
            + "[insert maximum here]"  //TODO do right before this point.)
            + "[ins 1st ½ num here]"  //TODO (These 4 values require checks
            + "[ins 2nd ½ num here]"  //TODO of x values (1st half vs 2nd).
            + "[ins 1st ½ ms here]"  //TODO For straddling cases...
            + "[ins 2nd ½ ms here]"  //TODO look at just start time.)
            ); /*TODO May need to make separate summary print block
                (this block) for when processing log files vice socket,
                since some of these values cannot be determined from
                log files (e.g. participant and soundscape mode).*/

    //*/ end deactivate-activate-code block

}// end main


/*********************/
/** Helper Function **/
/*********************/

private static int prompt(String string) {
    // TODO Auto-generated method stub
    return 0;
}



/**-----------------------------------------------------------------
 * -----method for extracting time point from timestamp string----
 * -----(in milliseconds this day (since beginning of day))-------
 * ------------------------------------------------by Adrian-----
 */

public static int extract_ms_this_day_from_timestamp (String timestamp) {

    int ms_this_day = 0;
    int ms = 0;
    int s = 0;
    int min = 0;
    int h = 0;

    String ms_places = "";
    String s_places = "";
    String min_places = "";
    String h_places = "";
```

```
    h_places = timestamp.substring(9, 11);
    min_places = timestamp.substring(11, 13);
    s_places = timestamp.substring(13, 15);
    ms_places = timestamp.substring(16, 19);

    /* example code trying to copy for int, but not easy!
    try {
        zDist_temp_d = new Double(tempSimDatum.trim());
    } catch (NumberFormatException nfe) {
        zDist_temp_d = 1000000;
    }
    */

    // deactivate or activate (comment out or 'comment in'??) code here
    /*/ my low-Eclipse-knowledge debugging technique.  Move this around:
    System.out.println("Program got to ms extraction method.");
    //*/

    // deactivate or activate (comment out or 'comment in'??) code here
    // part of my better low-Eclipse-knowledge debugging technique:
    progLocJustPreAnyCrash = "finished getting time span substrings";
    //*/

    try {
        h = Integer.parseInt(h_places, 10); // don't need ", 10" (radix)
    } catch (NumberFormatException nfe) {
        h = 1111; // much higher and would burst int size in ms_this_day?
    }
    // not going to bother with try-catch for the others
    min = Integer.parseInt(min_places, 10);
    s = Integer.parseInt(s_places, 10);
    ms = Integer.parseInt(ms_places, 10);

    ms_this_day = (h*60*60*1000) + (min*60*1000) + (s*1000) + ms;

    //System.out.print("timestamp string length = "
    //                   + timestamp.length() + ".  ");

    //xxxdeactivate or activate code (comment out or 'comment in'??) here
    //see date stamp parts, to make sure calculation correct
    System.out.println(h_places + "h, " + min_places + "min, "
                      + s_places + "s, and " + ms_places
                      + "ms since midnight (time zone?).  -->  Calculated "
                      + ms_this_day + "ms this day (today's time in ms).");
    //*/
    return ms_this_day;
}



/**------------------------------------------------------------------*/
/**-----methods for PCM sound (using JSyn synthesizer) set-up-----*/
/**-------------------------------------------------by Adrian-----*/
```

```
/**--Adding "static" to all 5 JSyn methods got program to run, but it
 * still crashed after the "select sonification" stage, throwing the
 * errors "synthPCM cannot be resolved to a variable" and
 * "synthPCM cannot be resolved".
 * ...and of course the "{myOsc,myOut,synthPCM} cannot be resolved"
 * and "[...] cannot be resolved to a variable" warnings persist.
 */

/** moved everything out of 4 of the methods and into the main stream!!
 * (all except the 'stop' one which does not get call initially)
 * ...because maybe that would make it start working!!*/
/*
//private void startSynthesisEngine() {
private static void startSynthesisEngine() {
    //synthPCM = JSyn.createSynthesizer();
    synthPCM.start();
}

//private void buildUnitGenerators() {
private static void buildUnitGenerators() {
    //synthPCM.add(myOsc = new SineOscillator());
    synthPCM.add(myOsc);
    //synthPCM.add(myOut = new LineOut());
    synthPCM.add(myOut);
    myOsc.frequency.set(440.0);  // 440 Hz  //err "cannot be resolved"
    myOsc.amplitude.set(0.5);  // half amplitude  //default 0.7?
                            //err "cannot be resolved"
}

//private void connectUnitGenerators() {
private static void connectUnitGenerators() {
    // connect oscillator to both channels of stereo player
    myOsc.output.connect(0, myOut.input, 0);
    myOsc.output.connect(0, myOut.input, 1);
}

//private void startUnitGenerators() {
private static void startUnitGenerators() {
    // start execution of units.  JSyn 'pulls' data so the only unit
    // you have to start() is the last one, in this case our LineOut
    myOut.start();
}
*/
/** removed this 5th of the 5 example code methods
 * because (a) it was probably only needed to "override" (mandatory),
 * where the original code "extended" some kind of web interface.
 * If the contents are needed at all, I would guess they should
 * go in the
 * (all except the 'stop' one which does not get call initially)
 * ...because maybe that would make it start working!!*/
/** (later note) Yep...it should be used,
 * otherwise the PCM sound continues after a crash
 * (such as disconnect from C2SM).
 * It has been copied to the last 'catch' in main
 * (the one with the text "Whoops! It didn't work")
```

```
 */
/** (even later note) moved to section where MIDI synth closed! */
/*
//public void stop() {
public static void stop() {
    if (synthPCM != null) {
        synthPCM.stop();
        synthPCM = null;
    }
}
*/

/**
 * This method returns the frequency mapped to the latest datum.
 *
 * @param inclinationRadians ...is the rover inclination in radians.
 * @param numbers ...is that hashtable thing I don't understand, and that
 *                 is almost certainly not needed for this method, but
 *                 that I am leaving in in case it happens to be the only
 *                 way to use class constants (instead of method constants).
 *                 I do not yet understand what variables declared where
 *                 and initialized(=?) where are usable (visible) where.
 * @param sonificationModeNum ...is the number corresponding to the
 *                             sonification mode, where:
 *                             1 = perception of linear scale
 *                             2 = perception of quadratic scale
 * @return ...returns the
 */

// MODE 1: The first mode is ... (description)
public static float find_freq_PCM(float inclinationRadians,
                                  Hashtable numbers,
                                  int sonificationModeNum)
{
    /* These variables moved to be class variables
     * instead of method variables.
     * This required putting them outside the 'main' method
     * (since that is static), and
     * calling the variables themselves static.

    float lowestFrequencyCONSTANT = (float) 1000;
    float octavesSpannedCONSTANT = (float) 1; //if 1 octave is spanned,
                                              // highestF = 2 * lowestF
    int frequencyStepsCONSTANT = 97;
    */

    float sonificationFreq = 0;

    //float freqBins[] = new float[frequencyStepsCONSTANT];
    //actually don't need discrete values.  Calculate directly instead

    float inclinationDegrees = inclinationRadians * 180
                               / (float)java.lang.Math.PI;

    if ((sonificationModeNum == 2) || (sonificationModeNum == 4)
            || (sonificationModeNum == 7) || (sonificationModeNum == 9)) {
```

```
            //formula for linear perceptual increase mapping is
            // (where j is lowest f)
            // (for one-octave span, 30-degree limit, x inclination in degrees)
            /** f = j*2^(x/30) */
            sonificationFreq = (float) (LOWEST_FREQUENCY_C
                                * java.lang.Math.pow(1 + OCTAVES_SPANNED_C,
                                                     inclinationDegrees / 30));
        }
        else if ((sonificationModeNum == 3) || (sonificationModeNum == 5)
                || (sonificationModeNum == 8) || (sonificationModeNum == 10)) {
            //formula for quadratic perceptual increase mapping is
            // (where j is lowest f)
            // (for one-octave span, 30-degree limit, x inclination in degrees)
            /** f = j*2^(x^2/900) */
            sonificationFreq = (float) (LOWEST_FREQUENCY_C
                                * java.lang.Math.pow(1 + OCTAVES_SPANNED_C,
                                        java.lang.Math.pow(inclinationDegrees, 2)
                                        / 900));
        }

        return sonificationFreq;
    }


    /** Helps set up the new note from data given */

    public static int find_note1(float var, Hashtable numbers)
    {
        int note_num = 0;
        if (var <= 0)
        {
            note_num = 20;
        } else if (var>0 && var<=0.03490)
        {
            note_num = 21;
        } else if (var>0.03490 && var<=0.06981)
        {
            note_num = 22;
        } else if (var>0.06981 && var<=0.10471)
        {
            note_num = 23;
        } else if (var>0.10471 && var<=0.13962)
        {
            note_num = 24;
        } else if (var>0.13962 && var<=0.17453)
        {
            note_num = 25;
        } else if (var>0.17453 && var<=0.20943)
        {
            note_num = 26;
        } else if (var>0.20943 && var<=0.24434)
        {
            note_num = 27;
        } else if (var>0.24434 && var<=0.27925)
        {
            note_num = 28;
```

```
    } else if (var>0.27925 && var<=0.31415)
    {
        note_num = 29;
    } else if (var>0.31415 && var<=0.34906)
    {
        note_num = 30;
    } else if (var>0.34906 && var<=0.38397)
    {
        note_num = 31;
    } else if (var>0.38397 && var<=0.41887)
    {
        note_num = 32;
    } else if (var>0.41887 && var<=0.45378)
    {
        note_num = 33;
    } else if (var>0.45378 && var<=0.48869)
    {
        note_num = 34;
    } else if (var>0.48869 && var<=0.52359)
    {
        note_num = 35;
    }

    else note_num = 20;

    return note_num;
}

// MODE 2: The second mode is ... (insert description)
public static int find_note2(float var, Hashtable numbers)
{
    int note_num = 0;
    if (var <= 0)
    {
        note_num = 50;
    } else if (var>0 && var<=0.03490)
    {
        note_num = 51;
    } else if (var>0.03490 && var<=0.06981)
    {
        note_num = 52;
    } else if (var>0.06981 && var<=0.10471)
    {
        note_num = 53;
    } else if (var>0.10471 && var<=0.13962)
    {
        note_num = 54;
    } else if (var>0.13962 && var<=0.17453)
    {
        note_num = 55;
    } else if (var>0.17453 && var<=0.20943)
    {
        note_num = 56;
    } else if (var>0.20943 && var<=0.24434)
    {
        note_num = 57;
```

```
    } else if (var>0.24434 && var<=0.27925)
    {
        note_num = 58;
    } else if (var>0.27925 && var<=0.31415)
    {
        note_num = 59;
    } else if (var>0.31415 && var<=0.34906)
    {
        note_num = 60;
    } else if (var>0.34906 && var<=0.38397)
    {
        note_num = 61;
    } else if (var>0.38397 && var<=0.41887)
    {
        note_num = 62;
    } else if (var>0.41887 && var<=0.45378)
    {
        note_num = 63;
    } else if (var>0.45378 && var<=0.48869)
    {
        note_num = 64;
    } else if (var>0.48869 && var<=0.52359)
    {
        note_num = 65;
    }

    else note_num = 20;

    return note_num;
}

public static int find_note3(float var, Hashtable numbers)
{
    int note_num = 0;
    if (var <= 0)
    {
        note_num = 100;
    } else if (var>0 && var<=0.03490)
    {
        note_num = 101;
    } else if (var>0.03490 && var<=0.06981)
    {
        note_num = 102;
    } else if (var>0.06981 && var<=0.10471)
    {
        note_num = 103;
    } else if (var>0.10471 && var<=0.13962)
    {
        note_num = 104;
    } else if (var>0.13962 && var<=0.17453)
    {
        note_num = 105;
    } else if (var>0.17453 && var<=0.20943)
    {
        note_num = 106;
    } else if (var>0.20943 && var<=0.24434)
```

```
    {
        note_num = 107;
    } else if (var>0.24434 && var<=0.27925)
    {
        note_num = 108;
    } else if (var>0.27925 && var<=0.31415)
    {
        note_num = 109;
    } else if (var>0.31415 && var<=0.34906)
    {
        note_num = 110;
    } else if (var>0.34906 && var<=0.38397)
    {
        note_num = 111;
    } else if (var>0.38397 && var<=0.41887)
    {
        note_num = 112;
    } else if (var>0.41887 && var<=0.45378)
    {
        note_num = 113;
    } else if (var>0.45378 && var<=0.48869)
    {
        note_num = 114;
    } else if (var>0.48869 && var<=0.52359)
    {
        note_num = 115;
    }

    else note_num = 20;

    return note_num;
}

public static int find_note4(float var, Hashtable numbers)
{
    int note_num = 0;
    if (var <= 0)
    {
        note_num = 150;
    } else if (var>0 && var<=0.03490)
    {
        note_num = 151;
    } else if (var>0.03490 && var<=0.06981)
    {
        note_num = 152;
    } else if (var>0.06981 && var<=0.10471)
    {
        note_num = 153;
    } else if (var>0.10471 && var<=0.13962)
    {
        note_num = 154;
    } else if (var>0.13962 && var<=0.17453)
    {
        note_num = 155;
    } else if (var>0.17453 && var<=0.20943)
    {
```

```
            note_num = 156;
        } else if (var>0.20943 && var<=0.24434)
        {
            note_num = 157;
        } else if (var>0.24434 && var<=0.27925)
        {
            note_num = 158;
        } else if (var>0.27925 && var<=0.31415)
        {
            note_num = 159;
        } else if (var>0.31415 && var<=0.34906)
        {
            note_num = 150;
        } else if (var>0.34906 && var<=0.38397)
        {
            note_num = 151;
        } else if (var>0.38397 && var<=0.41887)
        {
            note_num = 152;
        } else if (var>0.41887 && var<=0.45378)
        {
            note_num = 153;
        } else if (var>0.45378 && var<=0.48869)
        {
            note_num = 154;
        } else if (var>0.48869 && var<=0.52359)
        {
            note_num = 155;
        }

        else note_num = 20;

        return note_num;
    }

    public static int find_note5(float var, Hashtable numbers)
    {
        int note_num = 0;
        if (var >= 0.5239)
        {
            note_num = 10;
        }

        else note_num = 0;

        return note_num;
    }
} //end class client




/************
 * APPENDIX *
 ************
```

```
 ********************************
 * LIST OF VARIABLES - DATA_NUM *
 ********************************

 *0 timestamp in yyyyMMdd_HHmmss_fff format
 *1 translation along x in meters
 *2 translation along y in meters
 *3 translation along z in meters
 *4 rotation about x in radian
 *5 rotation about y in radian
 *6 rotation about z in radian
 *7 pan angle
 *8 tilt angle
 *9 speed

 ***********************
 * LIST OF INSTRUMENTS *
 ***********************

#0: Piano 1
#1: Piano 2
#2: Piano 3
#3: Honky-tonk
#4: E.Piano 1
#5: E.Piano 2
#6: Harpsichord
#7: Clav.
#8: Celesta
#9: Glockenspiel
#10: Music Box
#11: Vibraphone
#12: Marimba
#13: Xylophone
#14: Tubular-bell
#15: Santur
#16: Organ 1
#17: Organ 2
#18: Organ 3
#19: Church Org.1
#20: Reed Organ
#21: Accordion Fr
#22: Harmonica
#23: Bandoneon
#24: Nylon-str.Gt
#25: Steel-str.Gt
#26: Jazz Gt.
#27: Clean Gt.
#28: Muted Gt.
#29: Overdrive Gt
#30: DistortionGt
#31: Gt.Harmonics
#32: Acoustic Bs.
#33: Fingered Bs.
#34: Picked Bs.
#35: Fretless Bs.
#36: Slap Bass 1
```

```
#37: Slap Bass 2
#38: Synth Bass 1
#39: Synth Bass 2
#40: Violin
#41: Viola
#42: Cello
#43: Contrabass
#44: Tremolo Str
#45: PizzicatoStr
#46: Harp
#47: Timpani
#48: Strings
#49: Slow Strings
#50: Syn.Strings1
#51: Syn.Strings2
#52: Choir Aahs
#53: Voice Oohs
#54: SynVox
#55: OrchestraHit
#56: Trumpet
#57: Trombone
#58: Tuba
#59: MutedTrumpet
#60: French Horns
#61: Brass 1
#62: Synth Brass1
#63: Synth Brass2
#64: Soprano Sax
#65: Alto Sax
#66: Tenor Sax
#67: Baritone Sax
#68: Oboe
#69: English Horn
#70: Bassoon
#71: Clarinet
#72: Piccolo
#73: Flute
#74: Recorder
#75: Pan Flute
#76: Bottle Blow
#77: Shakuhachi
#78: Whistle
#79: Ocarina
#80: Square Wave
#81: Saw Wave
#82: Syn.Calliope
#83: Chiffer Lead
#84: Charang
#85: Solo Vox
#86: 5th Saw Wave
#87: Bass & Lead
#88: Fantasia
#89: Warm Pad
#90: Polysynth
#91: Space Voice
#92: Bowed Glass
```

#93: Metal Pad
#94: Halo Pad
#95: Sweep Pad
#96: Ice Rain
#97: Soundtrack
#98: Crystal
#99: Atmosphere
#100: Brightness
#101: Goblin
#102: Echo Drops
#103: Star Theme
#104: Sitar
#105: Banjo
#106: Shamisen
#107: Koto
#108: Kalimba
#109: Bagpipe
#110: Fiddle
#111: Shanai
#112: Tinkle Bell
#113: Agogo
#114: Steel Drums
#115: Woodblock
#116: Taiko
#117: Melo. Tom 1
#118: Synth Drum
#119: Reverse Cym.
#120: Gt.FretNoise
#121: Breath Noise
#122: Seashore
#123: Bird
#124: Telephone 1
#125: Helicopter
#126: Applause
#127: Gun Shot
#128: SynthBass101
#129: Trombone 2
#130: Fr.Horn 2
#131: Square
#132: Saw
#133: Syn Mallet
#134: Echo Bell
#135: Sitar 2
#136: Gt.Cut Noise
#137: Fl.Key Click
#138: Rain
#139: Dog
#140: Telephone 2
#141: Car-Engine
#142: Laughing
#143: Machine Gun
#144: Echo Pan
#145: String Slap
#146: Thunder
#147: Horse-Gallop
#148: DoorCreaking

```
#149: Car-Stop
#150: Screaming
#151: Lasergun
#152: Wind
#153: Bird 2
#154: Door
#155: Car-Pass
#156: Punch
#157: Explosion
#158: Stream
#159: Scratch
#160: Car-Crash
#161: Heart Beat
#162: Bubble
#163: Wind Chimes
#164: Siren
#165: Footsteps
#166: Train
#167: Jetplane
#168: Piano 1
#169: Piano 2
#170: Piano 3
#171: Honky-tonk
#172: Detuned EP 1
#173: Detuned EP 2
#174: Coupled Hps.
#175: Vibraphone
#176: Marimba
#177: Church Bell
#178: Detuned Or.1
#179: Detuned Or.2
#180: Church Org.2
#181: Accordion It
#182: Ukulele
#183: 12-str.Gt
#184: Hawaiian Gt.
#185: Chorus Gt.
#186: Funk Gt.
#187: Feedback Gt.
#188: Gt. Feedback
#189: Synth Bass 3
#190: Synth Bass 4
#191: Slow Violin
#192: Orchestra
#193: Syn.Strings3
#194: Brass 2
#195: Synth Brass3
#196: Synth Brass4
#197: Sine Wave
#198: Doctor Solo
#199: Taisho Koto
#200: Castanets
#201: Concert BD
#202: Melo. Tom 2
#203: 808 Tom
#204: Starship
```

```
#205: Carillon
#206: Elec Perc.
#207: Burst Noise
#208: Piano 1d
#209: E.Piano 1v
#210: E.Piano 2v
#211: Harpsichord
#212: 60's Organ 1
#213: Church Org.3
#214: Nylon Gt.o
#215: Mandolin
#216: Funk Gt.2
#217: Rubber Bass
#218: AnalogBrass1
#219: AnalogBrass2
#220: 60's E.Piano
#221: Harpsi.o
#222: Organ 4
#223: Organ 5
#224: Nylon Gt.2
#225: Choir Aahs 2
#226: Standard
#227: Room
#228: Power
#229: Electronic
#230: TR-808
#231: Jazz
#232: Brush
#233: Orchestra
#234: SFX
 *
 * */
```

Auditory Display and Simulator Telemetry Program – Secondary Java Class Source Code

```
import java.io.IOException;

import java.lang.*;
import java.io.*;
import java.net.*;
import java.util.Hashtable;

/***************
 * PLEASE READ *
 ***************/
/*****************************************************************
 * SimpleThreads.java is a helper class implemented to help       *
 * the main process in Client.java. The purpose of this file      *
 * is to be able to check for user inputs while running the       *
 * program to play sounds. This program will be responsible       *
 * for storing the inputs in the hash table "numbers" for all     *
 * setting changes including: instruments, data number, maximum   *
 * minimum, and the number of steps.                              *
 *****************************************************************/

/*
 *0 timestamp in yyyyMMdd_HHmmss_fff format
 *1 translation along x in meters
 *2 translation along y in meters
 *3 translation along z in meters
 *4 rotation about x in radian
 *5 rotation about y in radian
 *6 rotation about z in radian
 *7 pan angle
 *8 tilt angle
 *9 speed
 */

public class SimpleThreads implements Runnable {
    private BufferedReader bufferedReader;
    // variable name changed from "br" to "bufferedReader"
    // here and in Client.java for readability --Adrian
    private String cmd;
    private String [] lis = new String[5];
    private int temp;
    private float temp2;
    private char c;
    private Hashtable numbers;

    SimpleThreads(BufferedReader bufferedReader, Hashtable numbers){
        this.bufferedReader = bufferedReader;
        this.numbers = numbers;
    }

    public void run(){
          //  read the username from the command-line; need to use try/catch with the
          //  readLine() method
```

```
          //  use format of cmd ### ### ### for either:
          //  i instrument#
          //  d data# step# max min
          try {
             cmd = bufferedReader.readLine();
          } catch (IOException ioe) {
             System.out.println("IO error trying to read the instrument name!");
             System.exit(1);
          }

          c = cmd.charAt(0);
          lis = cmd.split(" ");
          // changes the instrument
          if (c == 'i')
          {
              temp = Integer.parseInt(lis[1].trim());
              numbers.put("inst", new Integer(temp));
          }
          // changes everything else
          else if (c == 'd')
          {
              temp = Integer.parseInt(lis[1]);
              numbers.put("num", new Integer(temp));

              temp = Integer.parseInt(lis[2]);
              numbers.put("step", new Integer(temp));

              temp2 = new Float(lis[3]);
              numbers.put("max", new Float(temp2));

              temp2 = new Float(lis[4]);
              numbers.put("min", new Float(temp2));
          }
    }
}
```

Appendix C – Subject Condition Sequences (Balanced Latin Squares)

\ Position Participant \	1	2	3	4	5	6	7	8
A	1	2	8	3	7	4	6	5
B	2	3	1	4	8	5	7	6
C	3	4	2	5	1	6	8	7
D	4	5	3	6	2	7	1	8
E	5	6	4	7	3	8	2	1
F	6	7	5	8	4	1	3	2
G	7	8	6	1	5	2	4	3
H	8	1	7	2	6	3	5	4

\ Position Participant \	1	2	3	4	5	6	7	8
I	7	4	6	5	1	2	8	3
J	8	5	7	6	2	3	1	4
K	1	6	8	7	3	4	2	5
L	2	7	1	8	4	5	3	6
M	3	8	2	1	5	6	4	7
N	4	1	3	2	6	7	5	8
O	5	2	4	3	7	8	6	1
P	6	3	5	4	8	1	7	2

Appendix D – Questionnaires

NASA Task Load Index

Auditory Conditions 1 through 8.

Please mark your response, for each condition, with the number of that condition.
(e.g. enter “1” at your response levels for the first auditory condition)

1. How mentally demanding was the task?

Mental Demand

Low High

2. How hurried or rushed was the pace of the task?

Temporal Demand

Low High

3. How hard did you have to work to accomplish your level of performance?

Effort

Low High

4. How insecure, discouraged, irritated, stressed, and annoyed were you?

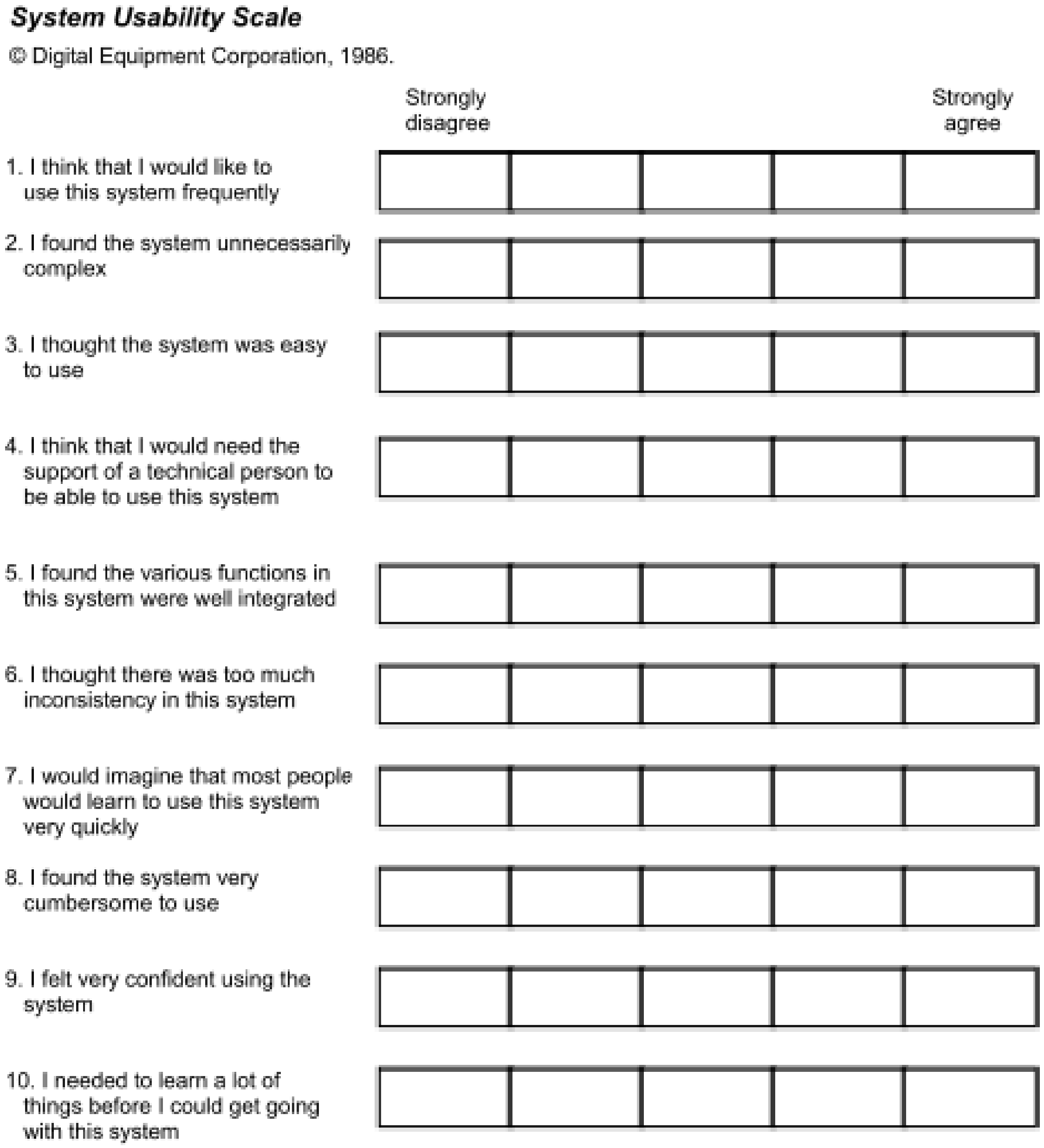
Frustration

Low High

5. How successful were you in accomplishing what you were asked to do?

Performance

Perfect Failure



System Usability Scale

	Strongly disagree				Strongly agree
1. I think that I would like to use this system frequently	☐	☐	☐	☐	☐
2. I found the system unnecessarily complex	☐	☐	☐	☐	☐
3. I thought the system was easy to use	☐	☐	☐	☐	☐
4. I think that I would need the support of a technical person to be able to use this system	☐	☐	☐	☐	☐
5. I found the various functions in this system were well integrated	☐	☐	☐	☐	☐
6. I thought there was too much inconsistency in this system	☐	☐	☐	☐	☐
7. I would imagine that most people would learn to use this system very quickly	☐	☐	☐	☐	☐
8. I found the system very cumbersome to use	☐	☐	☐	☐	☐
9. I felt very confident using the system	☐	☐	☐	☐	☐
10. I needed to learn a lot of things before I could get going with this system	☐	☐	☐	☐	☐

Participant Alias: ___________

Background Questionnaire

Visual and Auditory Aids in Telerobotic Control

Name: ______________________________________ **Age:** _____________yrs

☐ Male
☐ Female

Education Level Completed
Undergraduate
☐ 1st year ☐ 2nd year ☐ 3rd year ☐ 4th year

Graduate (please write in degree program): ___________________
☐ 1st year ☐ 2nd year ☐ 3rd year ☐ 4th year
☐ 5th year ☐ 6th year ☐ 7th year ☐ 8th year

Discipline/Department:
☐ Aerospace
☐ Biomaterials & Biomedical Engineering
☐ Chemical Engineering & Applied Chemistry
☐ Computer Science
☐ Civil Engineering
☐ Engineering Science
☐ Environmental Engineering & Energy Systems
☐ Electrical & Computer Engineering
☐ Industrial Engineering
☐ Materials Science & Engineering
☐ Mechanical Engineering
☐ Electrical & Computer Engineering
☐ Other: ______________________

How often do you play video or computer games per week on average?
☐ I do not play at all
☐ <5 hours ☐ 5-10 hours
☐ 10-15 hours ☐ 15-20 hours
☐ 20-25 hours ☐ >25 hours

If you do play video or computer games, what % of time do you use classic (e.g., keyboard, joystick) versus motion-sensing controllers?
_____% Classic (e.g., keyboard, joystick)
_____% Motion-sensing (e.g., Kinect, and (some) Wii games)

If you do play video or computer games, rank the top three genres of video games you play the most? (1 = most played; 2 = second most played; 3 = 3rd most played)
_____ Action & Adventure (e.g., Street Fighter, Zelda)
_____ Role-playing (e.g., Final Fantasy)
_____ Shooting - First Person (e.g., Unreal or Doom)
_____ Shooting – Third Person (e.g., Project Starfighter)
_____ Simulation – Life (e.g., SimCity)
_____ Simulation – Vehicle (e.g., flight simulator, Mario Kart, need for speed)
_____ Strategy (e.g., command and conquer, civilization)
_____ Sports – First and Third Person combined
_____ Sports – Third Person only
_____ Other: _________________________

[added after experiment proper:] "How would you rate your musical ability (based on music performance, music appreciation, and music education), on a scale from 1 to 7? – scale: (low)1, 2, 3, 4, 5, 6, 7(high)"

Appendix E – Data Analysis Code

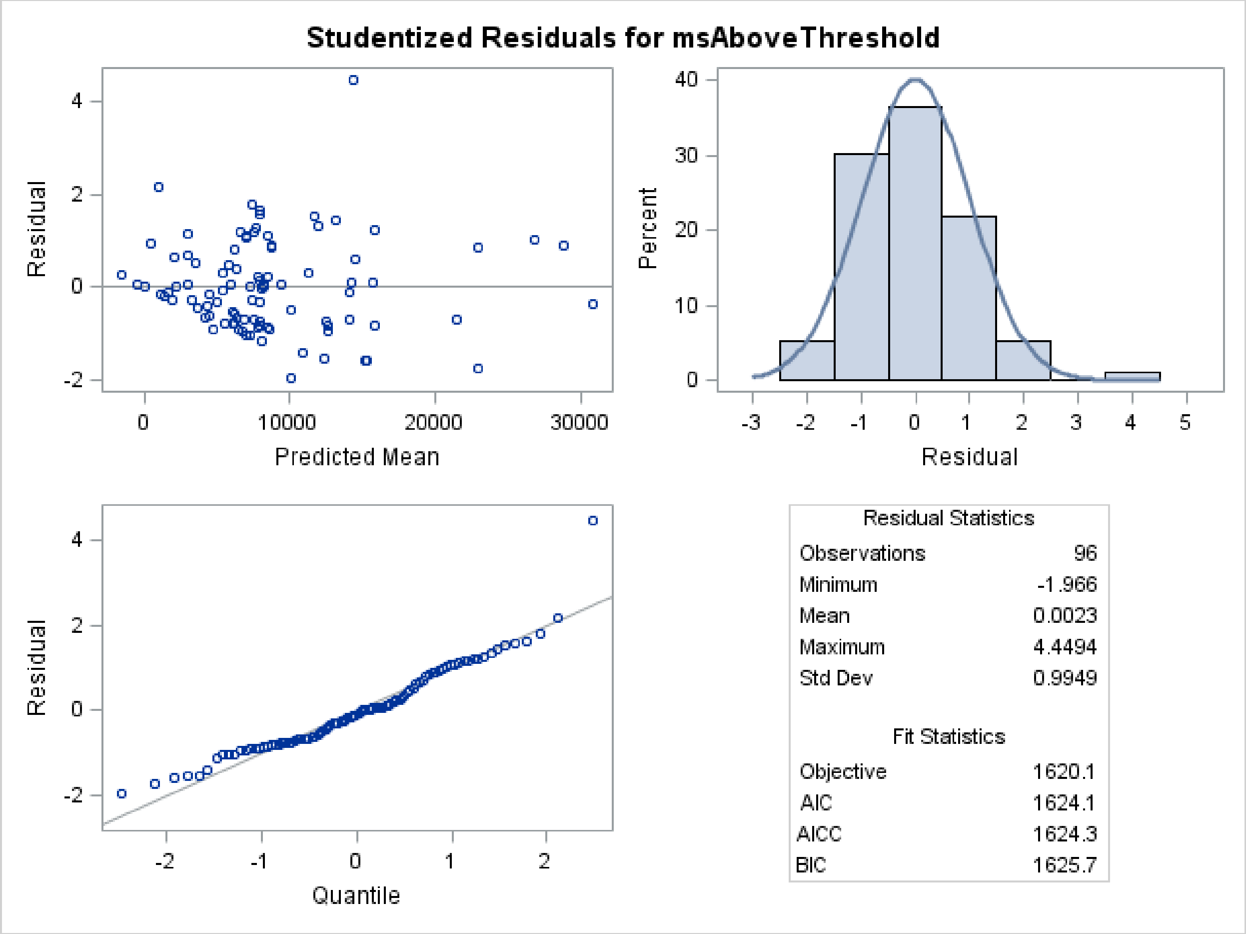
```
/* 2 rounds of stats on DV "msAboveThreshold" (cumulative time span above threshold)
/* -----------------------------------------------------------------------------
/* Start

/* planned contrasts using Datum Subset 1 (all observations except where sonification of data without
reference) */
PROC MIXED DATA=LELR2.DataWithAllCovs(where=(DataSubset1_no_sDatYessRefNo = 'sub1y'));
                                /*datum subset excluding conditions with sonification of data only*/
  CLASS subject orderPosition musicalAbility subjGender voice1yes sDat_sRef_map0lin1expon2NA; /*
expect to remove gender */
  MODEL msAboveThreshold = orderPosition musicalAbility subjGender /* covariates */
       voice1yes sDat_sRef_map0lin1expon2NA voice1yes*sDat_sRef_map0lin1expon2NA / RESIDUAL
solution outp = res;
  REPEATED  / SUBJECT=subject TYPE=CS;
  ESTIMATE 'noSoni vs soniData&Ref' sDat_sRef_map0lin1expon2NA 1 -0.5 -0.5 / cl;
  ESTIMATE 'noSoni vs soniData&Ref, when no voice' sDat_sRef_map0lin1expon2NA 1 -0.5 -0.5
sDat_sRef_map0lin1expon2NA*voice1yes 1 -0.5 -0.5 0 0 0 /cl;
  ESTIMATE 'mapPercepLin vs mapPercepExpon, when soniData&Ref' sDat_sRef_map0lin1expon2NA 0
1 -1 / cl;
  ESTIMATE 'mapPercepLin vs mapPercepExpon, when soniData&Ref, when no voice'
sDat_sRef_map0lin1expon2NA 0 1 -1 sDat_sRef_map0lin1expon2NA*voice1yes 0 1 -1 0 0 0/ cl;
lsmeans sDat_sRef_map0lin1expon2NA / pdiff cl;
lsmeans voice1yes / pdiff cl; /*added this.  may not be needed*/
run;
/* planned contrasts using Datum Subset 2 (all observations except where no background voice
communications) */
PROC MIXED DATA=LELR2.DataWithAllCovs(where=(DataSubset2_no_voiceNo = 'sub2y'));
                                /*datum subset excluding conditions with no voice*/
  CLASS subject orderPosition musicalAbility subjGender sDat_sRef_map0lin1expon2NA; /* expect to
remove gender */
  MODEL msAboveThreshold = orderPosition musicalAbility subjGender /* covariates */
       sDat_sRef_map0lin1expon2NA / RESIDUAL solution outp = res;
  REPEATED  / SUBJECT=subject TYPE=CS;
  ESTIMATE 'noSoni vs soniDataOnly, when voice' sDat_sRef_map0lin1expon2NA -1 0.5 0.5 0 0/ cl;
 /*used to be -1 0 0 0.5 0.5 (wrong, I say)*/
  ESTIMATE 'soniDataOnly vs soniData&Ref, when voice' sDat_sRef_map0lin1expon2NA 0 0.5 0.5 -0.5 -
0.5/ cl;
  ESTIMATE 'noSoni vs soniData&Ref, when voice' sDat_sRef_map0lin1expon2NA -1 0 0 0.5 0.5/ cl;
  ESTIMATE 'mapPercepLin vs mapPercepExpon, when voice' sDat_sRef_map0lin1expon2NA 0 0.5 -0.5
0.5 -0.5/ cl;
  lsmeans sDat_sRef_map0lin1expon2NA / pdiff cl; /*added this.  may not be needed*/
quit;

/* Finish
```

```
/* 2 rounds of stats on DV "msCourse" (time span for course completion (raw start to finish))
/* ------------------------------------------------------------------------------
/* Start

/* planned contrasts using Datum Subset 1 (all observations except where sonification of data without
reference) */
PROC MIXED DATA=LELR2.DataWithAllCovs(where=(DataSubset1_no_sDatYessRefNo = 'sub1y'));
                              /*datum subset excluding conditions with sonification of data only*/
  CLASS subject orderPosition musicalAbility subjGender voice1yes sDat_sRef_map0lin1expon2NA; /*
expect to remove gender */
  MODEL msCourse = orderPosition musicalAbility subjGender /* covariates */
       voice1yes sDat_sRef_map0lin1expon2NA voice1yes*sDat_sRef_map0lin1expon2NA / RESIDUAL
solution outp = res;
  REPEATED  / SUBJECT=subject TYPE=CS;
  ESTIMATE 'noSoni vs soniData&Ref' sDat_sRef_map0lin1expon2NA 1 -0.5 -0.5 / cl;
  ESTIMATE 'noSoni vs soniData&Ref, when no voice' sDat_sRef_map0lin1expon2NA 1 -0.5 -0.5
sDat_sRef_map0lin1expon2NA*voice1yes 1 -0.5 -0.5 0 0 0 /cl;
  ESTIMATE 'mapPercepLin vs mapPercepExpon, when soniData&Ref' sDat_sRef_map0lin1expon2NA 0
1 -1 / cl;
  ESTIMATE 'mapPercepLin vs mapPercepExpon, when soniData&Ref, when no voice'
sDat_sRef_map0lin1expon2NA 0 1 -1 sDat_sRef_map0lin1expon2NA*voice1yes 0 1 -1 0 0 0/ cl;
lsmeans sDat_sRef_map0lin1expon2NA / pdiff cl;
lsmeans voice1yes / pdiff cl; /*added this.  may not be needed*/
run;
/* planned contrasts using Datum Subset 2 (all observations except where no background voice
communications) */
PROC MIXED DATA=LELR2.DataWithAllCovs(where=(DataSubset2_no_voiceNo = 'sub2y'));
                              /*datum subset excluding conditions with no voice*/
  CLASS subject orderPosition musicalAbility subjGender sDat_sRef_map0lin1expon2NA; /* expect to
remove gender */
  MODEL msCourse = orderPosition musicalAbility subjGender /* covariates */
       sDat_sRef_map0lin1expon2NA / RESIDUAL solution outp = res;
  REPEATED  / SUBJECT=subject TYPE=CS;
  ESTIMATE 'noSoni vs soniDataOnly, when voice' sDat_sRef_map0lin1expon2NA -1 0.5 0.5 0 0/ cl;
 /*used to be -1 0 0 0.5 0.5 (wrong, I say)*/
  ESTIMATE 'soniDataOnly vs soniData&Ref, when voice' sDat_sRef_map0lin1expon2NA 0 0.5 0.5 -0.5 -
0.5/ cl;
  ESTIMATE 'noSoni vs soniData&Ref, when voice' sDat_sRef_map0lin1expon2NA -1 0 0 0.5 0.5/ cl;
  ESTIMATE 'mapPercepLin vs mapPercepExpon, when voice' sDat_sRef_map0lin1expon2NA 0 0.5 -0.5
0.5 -0.5/ cl;
  lsmeans sDat_sRef_map0lin1expon2NA / pdiff cl; /*added this.  may not be needed*/
quit;

/* Finish
```

```
/* 2 rounds of stats on DV "msCoursePlusPenalty" (time span for course completion + 2 * time span
above threshold, as explained to subjects)
/* -------------------------------------------------------------------------------
/* Start

/* planned contrasts using Datum Subset 1 (all observations except where sonification of data without
reference) */
PROC MIXED DATA=LELR2.DataWithAllCovs(where=(DataSubset1_no_sDatYessRefNo = 'sub1y'));
                                  /*datum subset excluding conditions with sonification of data only*/
  CLASS subject orderPosition musicalAbility subjGender voice1yes sDat_sRef_map0lin1expon2NA; /*
expect to remove gender */
  MODEL msCoursePlusPenalty = orderPosition musicalAbility subjGender /* covariates */
       voice1yes sDat_sRef_map0lin1expon2NA voice1yes*sDat_sRef_map0lin1expon2NA / RESIDUAL
solution outp = res;
  REPEATED  / SUBJECT=subject TYPE=CS;
  ESTIMATE 'noSoni vs soniData&Ref' sDat_sRef_map0lin1expon2NA 1 -0.5 -0.5 / cl;
  ESTIMATE 'noSoni vs soniData&Ref, when no voice' sDat_sRef_map0lin1expon2NA 1 -0.5 -0.5
sDat_sRef_map0lin1expon2NA*voice1yes 1 -0.5 -0.5 0 0 0 /cl;
  ESTIMATE 'mapPercepLin vs mapPercepExpon, when soniData&Ref' sDat_sRef_map0lin1expon2NA 0
1 -1 / cl;
  ESTIMATE 'mapPercepLin vs mapPercepExpon, when soniData&Ref, when no voice'
sDat_sRef_map0lin1expon2NA 0 1 -1 sDat_sRef_map0lin1expon2NA*voice1yes 0 1 -1 0 0 0/ cl;
lsmeans sDat_sRef_map0lin1expon2NA / pdiff cl;
lsmeans voice1yes / pdiff cl; /*added this.  may not be needed*/
run;
/* planned contrasts using Datum Subset 2 (all observations except where no background voice
communications) */
PROC MIXED DATA=LELR2.DataWithAllCovs(where=(DataSubset2_no_voiceNo = 'sub2y'));
                                  /*datum subset excluding conditions with no voice*/
  CLASS subject orderPosition musicalAbility subjGender sDat_sRef_map0lin1expon2NA; /* expect to
remove gender */
  MODEL msCoursePlusPenalty = orderPosition musicalAbility subjGender /* covariates */
       sDat_sRef_map0lin1expon2NA / RESIDUAL solution outp = res;
  REPEATED  / SUBJECT=subject TYPE=CS;
  ESTIMATE 'noSoni vs soniDataOnly, when voice' sDat_sRef_map0lin1expon2NA -1 0.5 0.5 0 0/ cl;
 /*used to be -1 0 0 0.5 0.5 (wrong, I say)*/
  ESTIMATE 'soniDataOnly vs soniData&Ref, when voice' sDat_sRef_map0lin1expon2NA 0 0.5 0.5 -0.5 -
0.5/ cl;
  ESTIMATE 'noSoni vs soniData&Ref, when voice' sDat_sRef_map0lin1expon2NA -1 0 0 0.5 0.5/ cl;
  ESTIMATE 'mapPercepLin vs mapPercepExpon, when voice' sDat_sRef_map0lin1expon2NA 0 0.5 -0.5
0.5 -0.5/ cl;
  lsmeans sDat_sRef_map0lin1expon2NA / pdiff cl; /*added this.  may not be needed*/
quit;

/* Finish
```

```
/* 1 more round of stats on DV "msAboveThreshold" (cumulative time span above threshold), this time on
ALL DATA */
PROC MIXED DATA=LELR2.DataWithAllCovs;  /* (no "where" statement, so full datum set) */
  CLASS subject orderPosition musicalAbility subjGender voice1yes sDat_sRef_map0lin1expon2NA; /*
expect to remove gender */
  MODEL msAboveThreshold = orderPosition musicalAbility subjGender /* covariates */
        voice1yes sDat_sRef_map0lin1expon2NA voice1yes*sDat_sRef_map0lin1expon2NA / RESIDUAL
solution outp = res;
  REPEATED  / SUBJECT=subject TYPE=CS;
lsmeans sDat_sRef_map0lin1expon2NA / pdiff cl;
lsmeans voice1yes / pdiff cl; /*added this.  may not be needed*/
run;

/* 1 more round of stats on DV "msCourse" (time span for course completion (raw start to finish)), this
time on ALL DATA */
PROC MIXED DATA=LELR2.DataWithAllCovs; /* (no "where" statement, so full datum set) */
  CLASS subject orderPosition musicalAbility subjGender voice1yes sDat_sRef_map0lin1expon2NA; /*
expect to remove gender */
  MODEL msCourse = orderPosition musicalAbility subjGender /* covariates */
        voice1yes sDat_sRef_map0lin1expon2NA voice1yes*sDat_sRef_map0lin1expon2NA / RESIDUAL
solution outp = res;
  REPEATED  / SUBJECT=subject TYPE=CS;
lsmeans sDat_sRef_map0lin1expon2NA / pdiff cl;
lsmeans voice1yes / pdiff cl; /*added this.  may not be needed*/
run;

/* 1 more round of stats on DV "msCoursePlusPenalty" (time span for course completion + 2 * time span
above threshold, as explained to subjects), this time on ALL DATA */
PROC MIXED DATA=LELR2.DataWithAllCovs; /* (no "where" statement, so full datum set) */
  CLASS subject orderPosition musicalAbility subjGender voice1yes sDat_sRef_map0lin1expon2NA; /*
expect to remove gender */
  MODEL msCoursePlusPenalty = orderPosition musicalAbility subjGender /* covariates */
        voice1yes sDat_sRef_map0lin1expon2NA voice1yes*sDat_sRef_map0lin1expon2NA / RESIDUAL
solution outp = res;
  REPEATED  / SUBJECT=subject TYPE=CS;
lsmeans sDat_sRef_map0lin1expon2NA / pdiff cl;
lsmeans voice1yes / pdiff cl; /*added this.  may not be needed*/
run;
```

Appendix F – Data Analysis Output

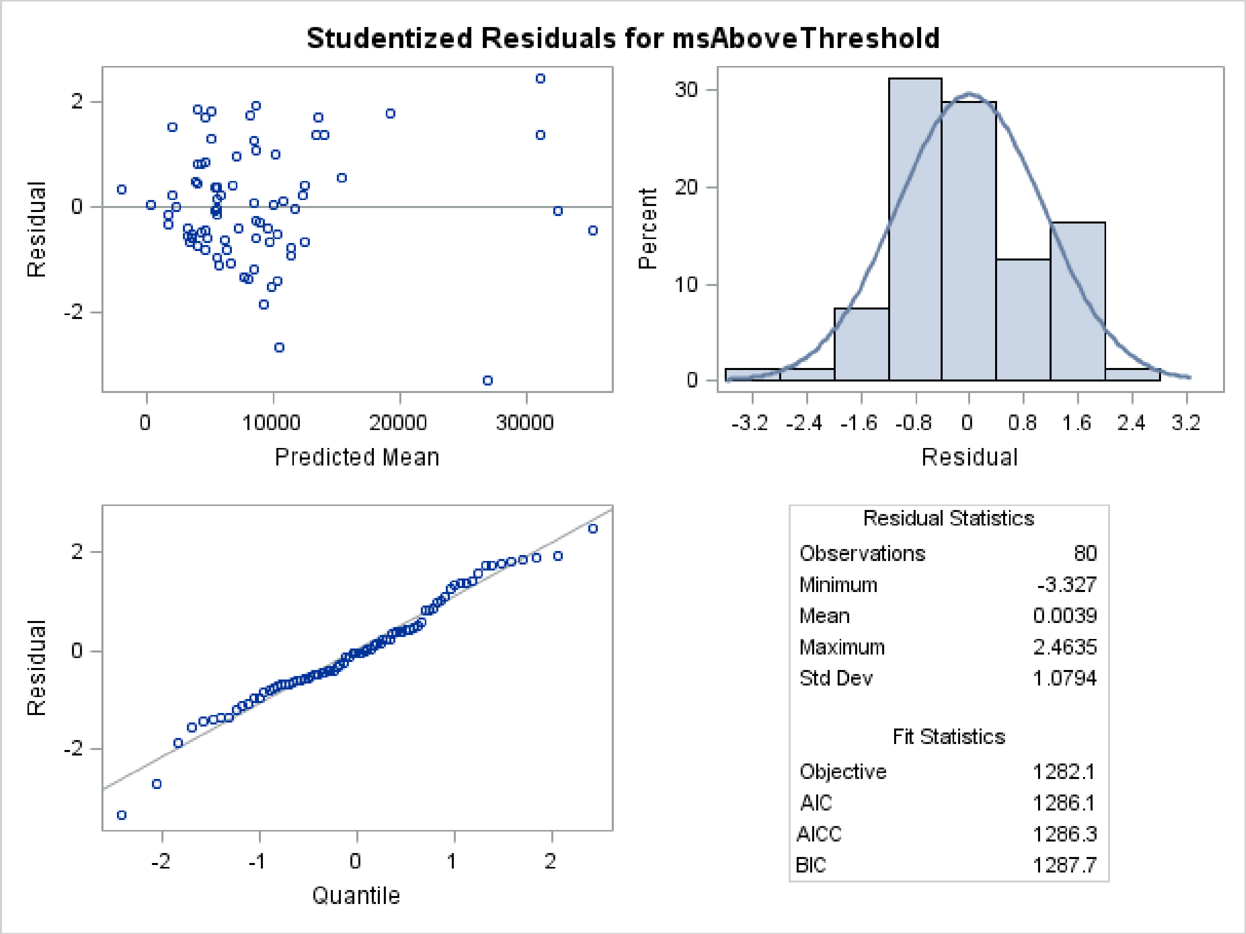


Studentized Residuals for msAboveThreshold

Residual Statistics	
Observations	96
Minimum	-1.966
Mean	0.0023
Maximum	4.4494
Std Dev	0.9949

Fit Statistics	
Objective	1620.1
AIC	1624.1
AICC	1624.3
BIC	1625.7

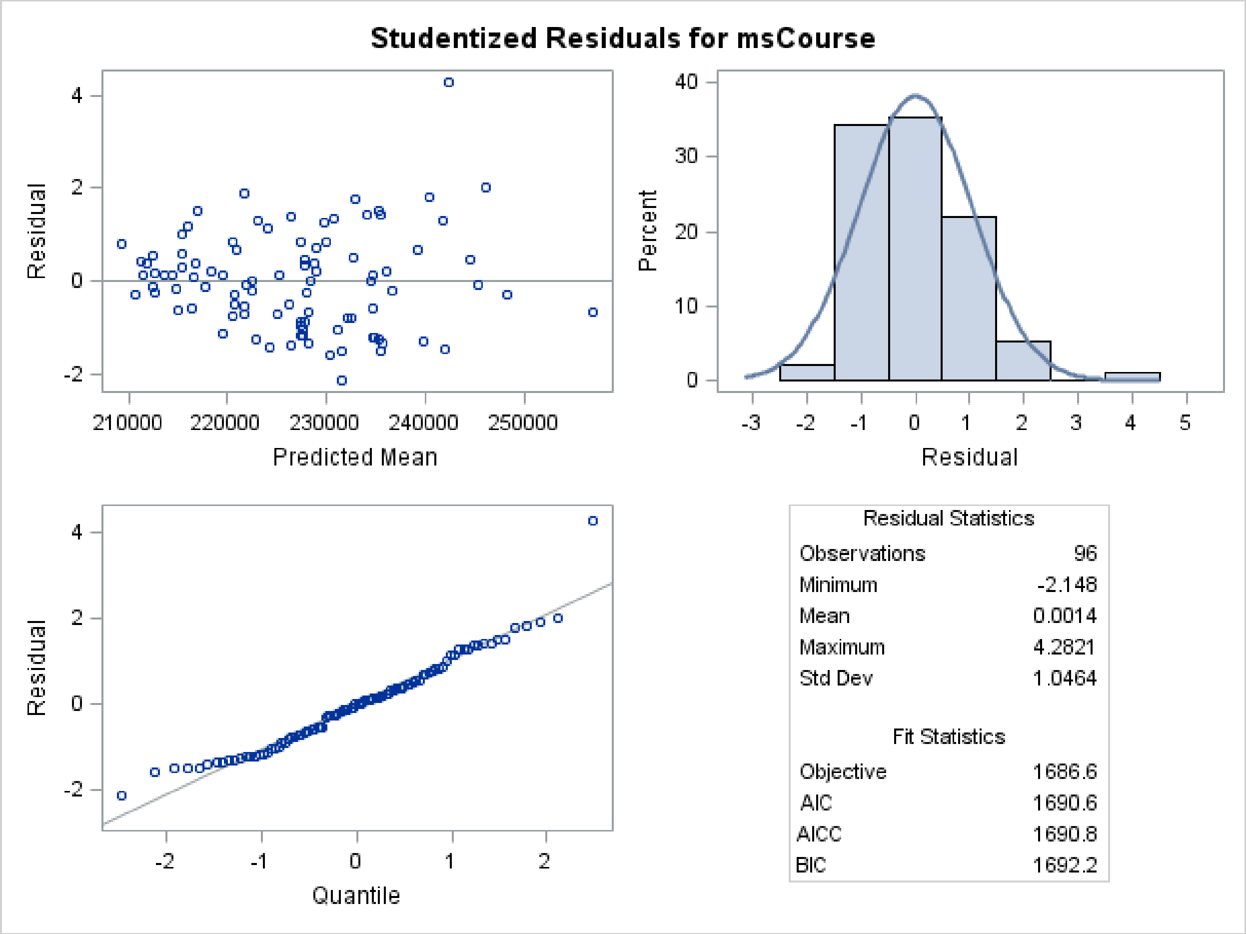
Datum Subset 1



Residual Statistics	
Observations	80
Minimum	-3.327
Mean	0.0039
Maximum	2.4635
Std Dev	1.0794

Fit Statistics	
Objective	1282.1
AIC	1286.1
AICC	1286.3
BIC	1287.7

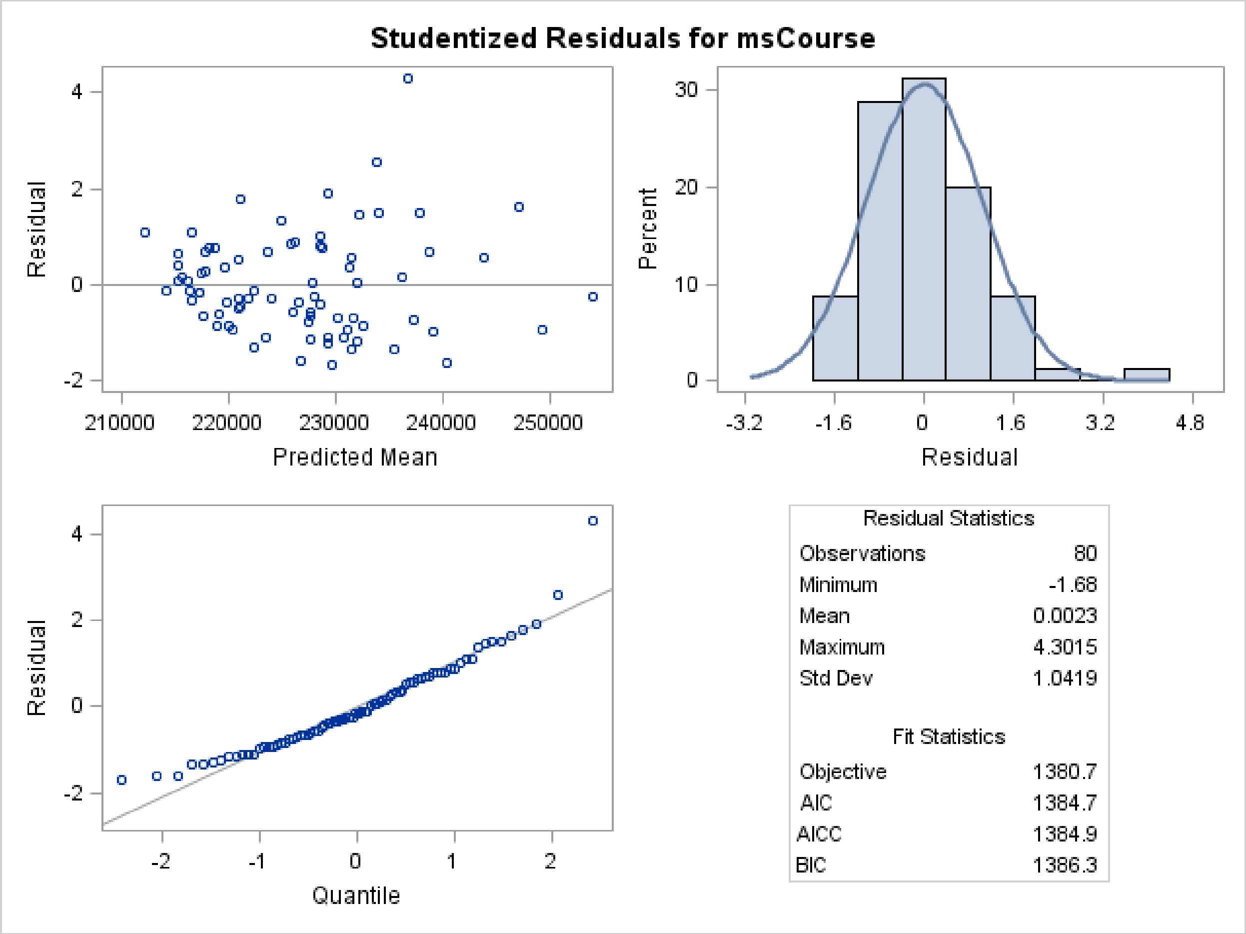
Datum Subset 2



Studentized Residuals for msCourse

```
Residual Statistics
Observations          96
Minimum           -2.148
Mean              0.0014
Maximum           4.2821
Std Dev           1.0464

Fit Statistics
Objective         1686.6
AIC               1690.6
AICC              1690.8
BIC               1692.2
```

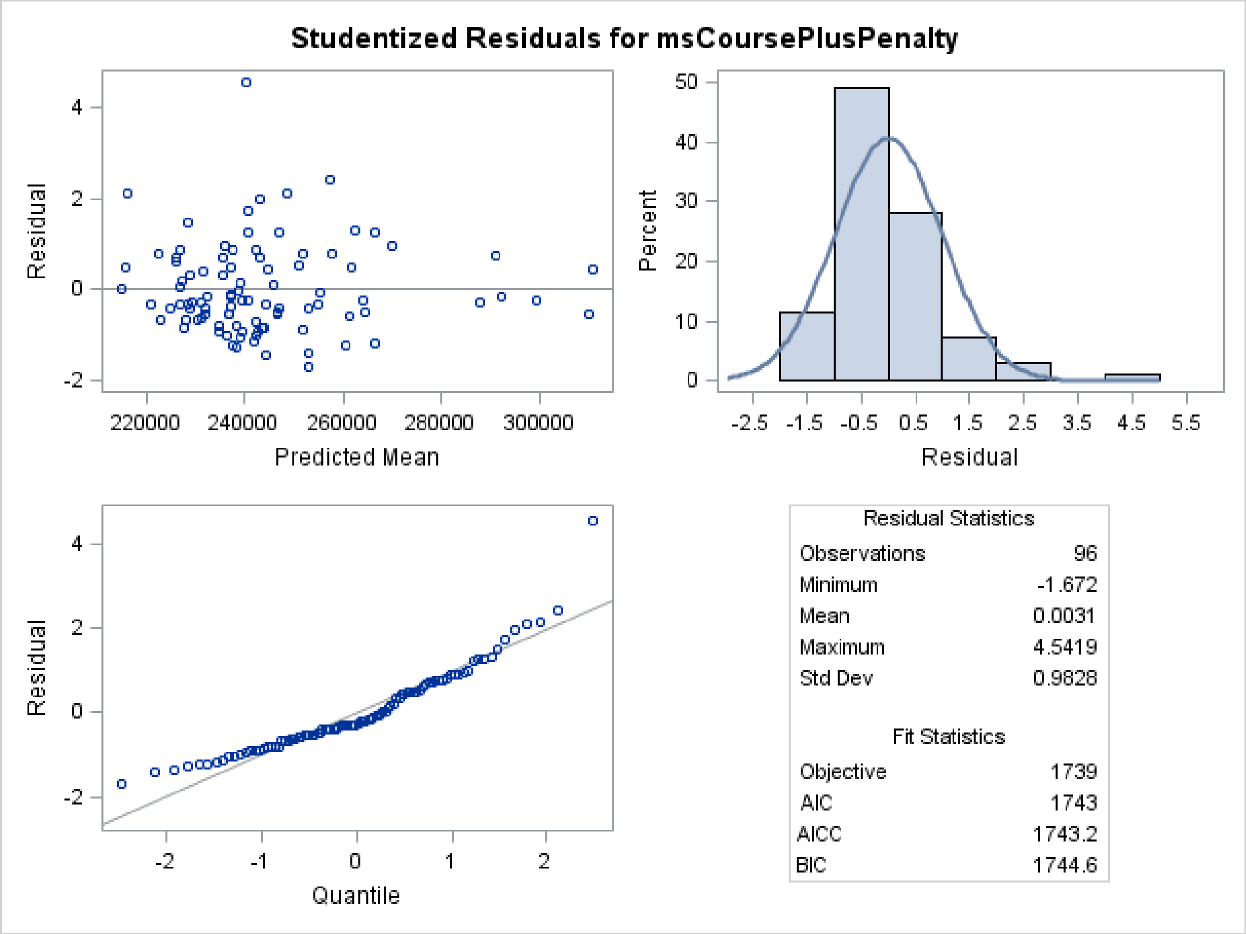
Datum Subset 1



Residual Statistics	
Observations	80
Minimum	-1.68
Mean	0.0023
Maximum	4.3015
Std Dev	1.0419

Fit Statistics	
Objective	1380.7
AIC	1384.7
AICC	1384.9
BIC	1386.3

Datum Subset 2

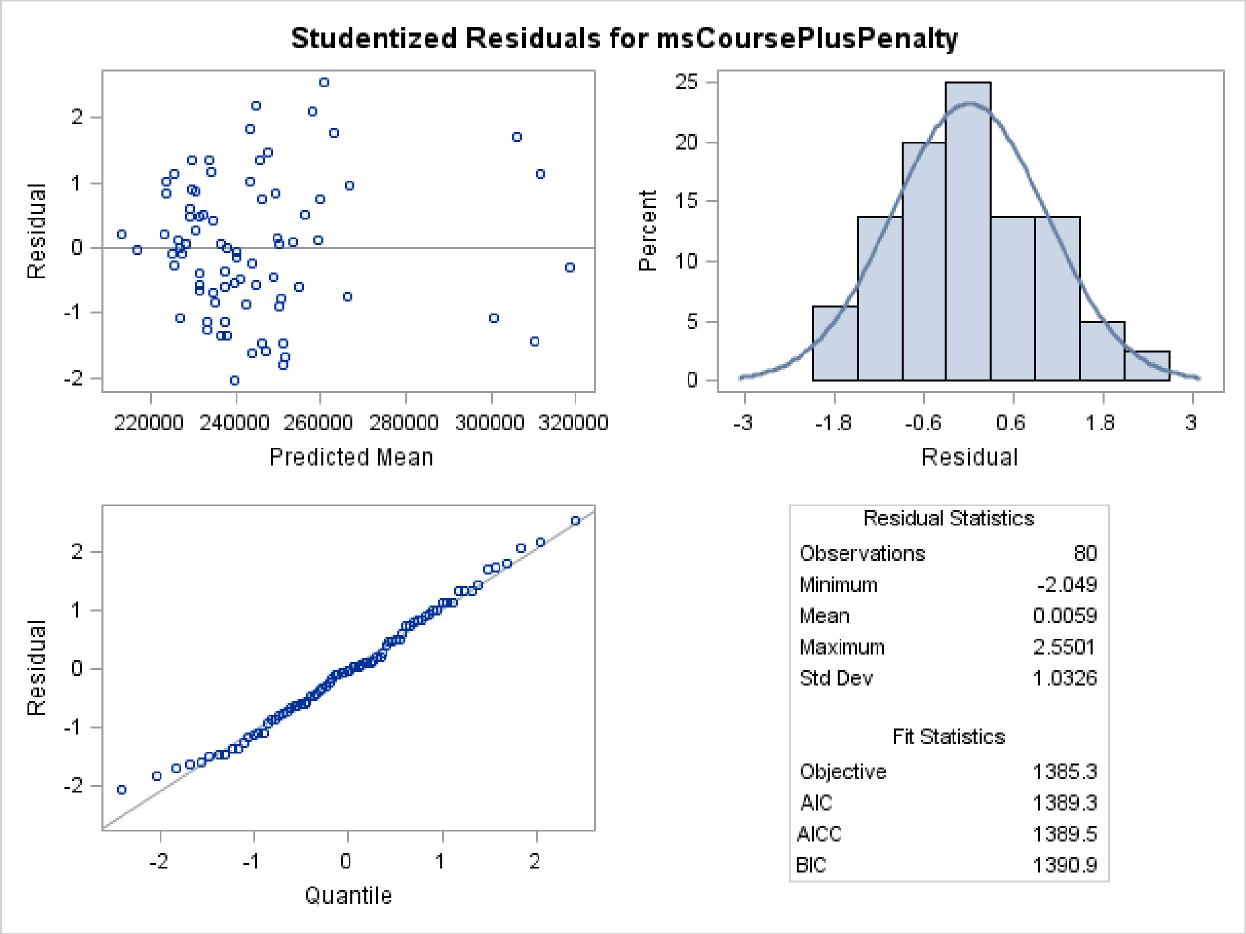


Studentized Residuals for msCoursePlusPenalty

Residual Statistics	
Observations	96
Minimum	-1.672
Mean	0.0031
Maximum	4.5419
Std Dev	0.9828

Fit Statistics	
Objective	1739
AIC	1743
AICC	1743.2
BIC	1744.6

Datum Subset 1



Studentized Residuals for msCoursePlusPenalty

Residual Statistics	
Observations	80
Minimum	-2.049
Mean	0.0059
Maximum	2.5501
Std Dev	1.0326

Fit Statistics	
Objective	1385.3
AIC	1389.3
AICC	1389.5
BIC	1390.9

Datum Subset 2

Copyright Acknowledgements