INFLUENCE OF THE VISUAL FIELD ON MANUAL ROLL AND LATERAL STABILIZATION

by

SCOTT BRADLEY STEPHENSON

B. S. E., University of Michigan - Ann Arbor, 1991

Submitted to the Department of Aeronautics and Astronautics
in Partial Fulfillment
of the Requirements for the Degree of

MASTER OF SCIENCE

in

AERONAUTICS AND ASTRONAUTICS

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Cambridge, Massachusetts
September, 1993

Signature of Author_____
Department of Aeronautics and Astronautics
September, 1993

Certified by_____
Dr. Daniel M. Merfeld
Thesis Supervisor

Certified by_____
Dr. Charles M. Oman
Director, Man-Vehicle Laboratory

Accepted by_____
Professor Harold Y. Wachman
Chairman, Departmental Graduate Committee

INFLUENCE OF THE VISUAL FIELD ON MANUAL ROLL AND LATERAL STABILIZATION

by

SCOTT BRADLEY STEPHENSON

Submitted to the Department of Aeronautics and Astronautics
in Partial Fulfillment of the Requirements for the
Degree of Master of Science
in Aeronautics and Astronautics

ABSTRACT

The experiments presented in this thesis were directed at investigating otolith-canal-visual interactions (more specifically, how the signals from these systems are combined by the central nervous system to yield a wideband sensory system) and operator manual control strategies. Operator performance in the manual roll and lateral stabilization tasks in the presence of five different visual fields was studied to address the following questions: (1) role of vision in low frequency motion sensation; (2) role of high frequency vestibular inputs in motion sensation ; (3) influence of the visual field on the operator control strategy; and (4) accuracy of the McRuer Crossover Model in predicting the operator control strategy.

The human control performance in the closed-loop task of nulling perceived roll tilt was studied. Five types of visual motion cues were presented in the subject's peripheral visual field: (1) DARK - only an illuminated red fixation point could be seen by the subject; (2) CON - a countermoving field which moved in the direction opposite to the trainer but with the same speed; (3) FIX - field fixed with respect to the subject so that reliance was mainly upon vestibular cues; (4) SS - sum of sines pseudo-random stimulus; (5) CV - field moving at a constant velocity with respect to the subject independent of motion platform position or velocity. Subject performance in the nulling task was best with CON, followed by FIX, DARK, SS, and CV; results for CON, DARK, FIX, and CV agreed with those of other experimenters. Subjects responded to SS at frequencies up to approximately 0.15 Hz, confirming the key role of vision in the low frequency region. At high frequencies, subjects were relatively unaffected by the visual field, and relied primarily on vestibular information. Subject response agreed with the predictions of the McRuer Crossover Model, although an additional 30-45 degrees of phase lead was seen that was not predicted by the model. This phase lead was most likely generated by the semicircular canals, which act as good velocity sensors in the frequency range of this experiment.

The human control performance in the closed-loop task of nulling perceived linear velocity along the interaural axis was also studied . The DARK, CON, FIX, and SS visual fields used in the roll experiment were adapted for use in this experiment. The subject performance in the nulling task was best with CON, followed by DARK, SS, and FIX. Subjects did not respond to SS, in agreement with the work of Huang. Subject response

agreed with the predictions of the McRuer Crossover Model for all visual fields (although a strict numerical comparison for FIX was not possible due to a poor model function fit), although an additional 20-40 degrees of phase lead was seen that was not predicted by the model.

A comparison of the manual roll (MRS) and manual lateral stabilization task (MLS) showed that: (1) the mean remnant was much larger (when compared to the vestibular disturbance) and more "scattered" in MLS than MRS; (2) subjects consistently added energy at the high stimulus frequencies in MLS, but rarely did so in MRS; (3) low frequency nulling proficiency was good in both MRS and MLS, and tended to worsen with increasing frequency; (4) subjects responded to the SS visual field in MRS, but did not in MLS; (5) error bars were much smaller in MRS than MLS (particularly at high frequencies), implying a higher repeatability in MRS; and (6) scalar performance measures (SPMs) were higher for all subjects for all visual fields in MRS than MLS.

Hardware and software were developed to conduct the experiments and analyze the data. Analog closed-loop velocity controllers and a roll axis position controller were designed to allow the Link GAT-1 to function as a position or velocity servo for external commands. A velocity controller and high slew-rate power amplifier were developed to allow closed-loop control of the projection system. Sixth order anti-alias Bessel filters were designed and built to filter the data prior to acquisition. Finally, an in-house software package, used to control the MVL sled, was modified to control the Link, and routines were written to generate pseudo-random sum of sines position and velocity disturbances.

Recommendations for further research include an investigation of manual pitch stabilization, modeling visual-vestibular interaction in roll and pitch, quantifying changes in vection strength due to spaceflight, studying manual stabilization in the Z axis, and investigating the Otolith Tilt Translation Reinterpretation (OTTR) hypothesis. Implementations of the latter two experiments are, at the writing of this thesis, underway as a part of the E-072 experiment series for the Spacelab Life Sciences (SLS-2) shuttle mission scheduled to be launched in late 1993.

Thesis Supervisor: Dr. Daniel M. Merfeld
Research Scientist and Lecturer

ACKNOWLEDGMENTS

To my family, Mom, Dad, John, and Dave, I would like to thank you for encouraging me to strive for excellence, and for the occasional pep talk to keep me going.

To Dan Merfeld, my advisor and fellow Badger, thank you for the helpful discussion of the contents of this thesis, and the endless help in preparing this document.

To my friends. Matt Butler, thanks for your friendship, healthy dose of sarcasm, and general love-of-life that kept me laughing through the Hell that was this thesis. To Ted Liefeld, thanks for braving the Grizzly Bears of Montana and reminding me of my love for nature. To Rick Paxson and Marc Schafer for showing me that life does get better after MIT. To Karen Pouliot, for the great hugs and daily phone calls to make sure I was still alive. To Karla, Keoki, Ted, Corrie, Michelle, Juan, Karl, and Chris for making the lab a great environment and the trips to Houston bearable. To Rick, thanks for showing me that nice guys don't always finish last and to Wendy, for your incredible gift of understanding and sound advice. To Yu Hasegawa, thanks for my first teaching experience and trip to Kenya; it has made me a better person.

For technical help, I would like to thank Jim Costello for your patience during the countless hours spent troubleshooting the Link, enthusiasm for my accomplishments, and lessons on life that were instrumental in making all of this possible. Thanks to Ted Liefeld for customizing my LabView data acquisition system, Alan Natapoff for advising me on the statistics and encouraging me to delve into the "guts" of my data, Karla Polutchko for tutoring on the MVL sled and Systat, and Lynetta Frasure for helping to develop the hardware for the Link experiments.

To the lunch time crowd, especially Jim, Beverly, Sherry, Barbara, Kim, and Kristen, for providing the comic relief that helped to keep me sane (as well as providing excellent material for my progress reports...).

Thanks to Jeff Jarosz and ONR for providing the funding which made all of this possible.

An finally, thanks to all of those that I have *temporarily* forgotten in the excitement of completing this thesis, but who have each touched my life.

And now for something completely different...

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. Research Scope and Objectives

This research is concerned with investigating visual-vestibular interaction; more specifically, the influence of the visual field on vestibular responses during rotatory tilt and linear motion. There were three main goals for this research: (1) to study the frequency response of a subject to a multi-frequency pseudo-random motion during the tilt nulling and linear velocity nulling tasks with different types of visual fields; (2) to compare the manual roll position nulling task with the lateral linear velocity nulling task; and (3) to develop preflight/postflight experiments to investigate the Otolith Tilt Translation Reinterpretation (OTTR) Hypothesis as part of Spacelab Life Sciences-2.

1.2. The Vestibular System

The vestibular system, man's motion sensing center, is located in the inner ear and provides information on body orientation and motion relative to the environment. It consists of two sets of organs, the semicircular canals and the otoliths, which are enclosed in a cavity called the bony labyrinth within the temporal bones of the skull.

1.2.1. The Semicircular Canal System

The semicircular canals consist of three approximately circular toroidal canals which lie in approximately orthogonal planes. The canals are suspended in a fluid called perilymph in the temporal bones of the skull. The canals are filled with a viscous (and relatively high inertia) fluid called endolymph, which lags behind the motion of the canals when the head undergoes an angular acceleration. At one end of each canal is an enlarged section, called the ampulla, within which is found a gelatinous, elastically restrained protuberance called the cupula, which is in turn attached to the crista. The crista is a raised section of the inner wall of the ampulla, and contains a layer of sensitive sensory hair cells. When the endolymph moves with respect to the canal, the cupula is displaced, and this relative motion is transduced as a change in firing frequency of the hair cells. This

travels down the eighth cranial nerve to the central nervous system, where it represents head rotation. Each of the canals on the right side of the head are essentially coplanar with a canal on the left side of the head, and are thus pairwise sensitive to rotations about the same axes [14].

1.2.2. The Otolith System

The otolith system is comprised of two anatomically different organs: the utricle and the saccule. The utricle is an oblong chamber common to all three semicircular canals and lies in a plane inclined 20 degrees above the horizontal. The saccule, on the other hand, lies in an almost vertical plane. Each otolith organ contains beds of sensory hair cells and supporting cells called the maculae. Situated on top of the cilia of the hair cells is a gelatinous mass containing a large number of dense crystals of calcium carbonate called otoconia, which are suspended in a layer of endolymph. The specific gravity of the otoconia is high in comparison to the endolymph, and thus the otoliths respond to linear accelerations, including gravity [14].

The utricles are oriented such that the major plane of their sensitivity is parallel to the plane of the horizontal semicircular canals. The saccular organs are oriented so that their plane of sensitivity is perpendicular to the horizontal canals and roughly parallel to the median plane. The utricular maculae are sensitive to linear accelerations in any direction within the utricular plane, and respond to tilt in any direction. The saccular maculae appear to respond to low frequency vibration, gravity, and linear accelerations, especially along the longitudinal axis [14].

1.3. Pathways for Visual-Vestibular Interaction

Neurons in the visual system have been described that have the appropriate direction and velocity coding to combine with activity from the vestibular system for use in visual-vestibular interactions [10]. However, the anatomical pathways that relay visual information to the vestibular nuclei are not clear, nor is it known what parameters of the visual world they convey. In order to interact properly with the vestibular input, visual

pathways should carry information about the direction and velocity of large peripheral moving fields.

Anatomical structures that relay information from the retina to the brainstem are the accessory optic system, the nucleus of the optic tract of the pretectum, and the superior colliculi [10]. To date, none of them have been shown to have direct connections to the vestibular nuclei. However, projections from the accessory optic system and pretectum can be traced indirectly to the vestibular nuclei via the flocculus [10]. Cortico-pontine pathways might also provide visual information to the vestibular system.

1.4. Motivation for Research

It is well known that visual and vestibular information interact in the perception of spatial orientation and motion. The fact that the sensation of motion is not restricted to inputs from the labyrinth can been seen, for example, when one leans over a bridge with the sight of moving water and experiences the illusion of self-motion. In fact, the sense of motion is dependent on the interaction of inputs from virtually every sensory system, including the visual, proprioceptive, tactile, and auditory systems, as well as the labyrinth [10].

The phenomenon of visual-vestibular interaction was first described by Mach in 1875. Before the publications of Mach and Breuer, it was not realized that the labyrinths were separate sensory organs for detecting angular and linear accelerations [17,18,3]. Purkinje subjected himself to different kinds of accelerations, but saw no need to introduce a separate sense for motion in addition to the five accepted senses of smell, vision, taste, hearing, and touch [20]. A few years later, Flourens made the first extensive measurements on the labyrinths. In his studies, he destroyed single semicircular canals or pairs of canals and observed the pathological body posture and movement disorders which resulted [6,7]. Thus began the work that led eventually to the understanding of discrete components of the vestibular system. Even today, after more than 100 years of research and clinical experience, it is still not clear how the sensation of motion is created centrally.

No single unifying theory has evolved, and the mechanisms of visual-vestibular interaction can still be debated.

Much work has been done to quantify the influence of the visual field on motion sensation during earth-vertical rotation and linear acceleration [12-16,27,28]. However, much less work has been directed toward the problem of the influence of the visual field on motion sensation during roll and pitch tilt (the question of otolith/canal/visual interaction has yet to be resolved). The studies which have been done to date to investigate visual-vestibular interaction are summarized in table 1.1, and are discussed in more detail in the following section.

1.4.1. Visual Field Influence on Motion Sensation and Manual Stabilization

Young et al. have shown that thresholds for detection of angular accelerations during yaw circularvection are raised when the acceleration is opposite to the direction of circularvection; times to detect these accelerations are similarly increased [25]. In addition, they have shown that magnitude estimates of angular velocity show the effect of a visually induced velocity offset which is increased slightly by vestibular responses in the same direction, and decreased markedly when the vestibular responses are in the direction opposite to self-rotation. Zacharias et al. have measured manual control performance in the closed-loop task of nulling perceived self-rotation velocity about the vertical axis, and have found that low-frequency visual cues (which dominate low-frequency sensations) are used to augment high-frequency vestibular cues to effect a wide-band sensory system [27]. Huang et al. have verified these results for lateral (inter-aural) linear accelerations [14]. Huang et al. have also shown that the latency time for perception of angular accelerations is reduced from that in the dark when the trial is performed in the light, and further reduced when the subject is allowed to fixate on a central illuminated spot (oculogyral illusion) [12-16].

Researcher(s)	Measurement(s) Made	None	Fixed	Constant Velocity	Counter-moving	Oculogyral Illusion	Dual Input
Dichgans et al.	Induced tilt and manual roll stabilization			X			
Young et al.	Ang. Acc. thresholds; ang. vel. magnitude estimates			X			
Zacharias	Yaw closed-loop velocity nulling		X	X	X		X
Huang	Manual roll and pitch stabilization	X	X	X	X	X	
Huang	Motion sensation in yaw	X	X			X	
Huang	Manual lateral stabilization	X	X	X	X		X

Table 1.1: Summary of previous research and visual fields used.

Dichgans et al. have found that when a moving visual field whose axis of rotation is along the observer's line of sight is observed, the apparent vertical assumes a steady-state offset from the true vertical in the direction of field rotation [4]. Although the magnitude of the perceived tilt has a high inter-individual variation, it is extremely replicable for an individual. Held et al. confirmed the importance of peripheral field stimulation on tilt illusion strength [9]. In addition, they showed that the perceived tilt angle reaches a steady-state value after a latency time of approximately 30 seconds and is linearly dependent on field velocity, reaching saturation at 40 degrees/s. Young et al. hypothesized that visually induced tilt is limited by conflict with otolith information [26]. Huang investigated this hypothesis by extending the experimental and analytical approach of Zacharias in yaw to study manual roll stabilization. In his study, he addressed the two main problems of performing experiments in roll: (1) limited roll angles for a fixed-base simulator within which to produce a strong pseudo-random signal as an effective stimulation; and (2) the addition of the linear acceleration otolith signal to the semicircular canal signal. He hypothesized (based primarily on the work of Zacharias) that although otolith-canal interaction is still an open question, peripheral visual field motion will provide low frequency information driving subjective sensation on manual roll stabilization. His hypothesis proved correct for constant velocity visual fields where he saw a 2 degree average bias in trainer position in the direction of the visual field during manual roll stabilization using a 8 degree/second constant velocity visual field [13]. However, no attempt was made to study the frequency response characteristics of the subject using a multi-frequency pseudo random visual stimulation.

1.5. Approach to the Problem

1.5.1. Subject Task in Roll Position Nulling Experiment

To study the effect of the visual field on manual roll stabilization, subjects are seated in the trainer with the following five types of visual fields:

(1) DARK : Only an illuminated red fixation point can be seen by the subject.

(2) CON : A countermoving field which moves in the direction opposite to the motion platform but with the same speed.

(3) FIX : A field fixed with respect to the subject so that reliance is mainly upon vestibular cues.

(4) SS : Sum of Sines pseudo-random stimulus.

(5) CV : A field moving at a constant velocity with respect to the subject independent of motion platform position or velocity.

For each visual field, the subjects attempt to maintain themselves upright in the presence of a pseudo-random roll disturbance. Time histories of subject response are recorded, from which frequency response information can be derived.

1.5.2. Subject Task in Lateral Velocity Nulling Experiment

To study the effect of the visual field on lateral velocity nulling, subjects are seated in the US Laboratory Sled with visual fields 1-4 listed in section 1.5.1. A constant velocity field was not used, since the velocity drift induced by the resulting linear vection caused the subject to exceed the sled track limits before the trial was completed. For each visual field, the subjects attempt to maintain zero velocity in the presence of a pseudo-random y axis (intra-aural) disturbance. Time histories of subject response are again recorded, from which frequency response information can be derived.

1.6. Thesis Organization and Objectives

There are four parts to this thesis, covering the following topics:

1.6.1. Part One - Link GAT-1 Trainer System Development

This part of the thesis discusses the software and hardware used in (and developed for) the manual roll stabilization experiments. A simple model of the Link GAT-1 trainer is developed, from which a closed-loop three axis velocity controller is designed. Analog circuit diagrams show how the controller was implemented, and Bode plots of the Link velocity dynamics show the closed-loop response of the system. All additional hardware developed for the Link, including the roll position controller, projection system power

amplifier and controller, data acquisition filters, subject control wheel, and counterrotating visual field circuitry are also discussed, as is the Link control software. The materials for this part are included in chapters 2 and 3.

1.6.2. Part Two - Visual Field Influence on Manual Roll Stabilization

The purpose of this work is to study the response of a subject during performance of the manual roll stabilization task with different visual motion cues. The specific questions addressed include: (1) is low-frequency sensation determined by visual input? (2) do high-frequency vestibular inputs complement this information? The materials for this part are included in chapters 4 and 5.

1.6.3. Part Three - Visual Field Influence on Manual Lateral Stabilization

The purpose of this work is to study the response of a subject during performance of the manual lateral stabilization task with different visual motion cues. Specific questions addressed are the same as with the roll experiments given in section 1.6.2. The materials for this part are included in chapters 6 and 7.

1.6.4. Part Four - Discussion of Results

This chapter discusses the results of the manual roll and lateral stabilization experiments, and compares them in the time and frequency domains. This comparison provides valuable insight into the importance of the otolith cue in the nulling task. Recommendations for future studies are also discussed here. This material is included in chapter 8.

PART ONE

LINK GAT-1 TRAINER SYSTEM DEVELOPMENT

2. DESIGN OF A CLOSED-LOOP THREE AXIS VELOCITY CONTROLLER

2.1. Introduction

A Northgate 386 IBM clone and an A/D board were purchased to control the Link GAT-1 flight trainer using a modified version of the MVL Sled software. In order for the Link to track computer commands, a closed-loop three axis analog velocity controller had to be designed and implemented. This chapter details the design of the analog yaw velocity controller and the modifications which were made for the roll and pitch axes.

2.2. Definition of System Variables

The following is intended as a complete list of the system variables which are referred to in the sequel:

J..... Link polar moment of inertia

f..... Link dynamic friction coefficient

n..... gear reducer ratio

R_a.... armature resistance

L_a.... armature inductance

I_a..... armature current

I_f..... field current

K_m.... motor torque constant

K_b..... back emf constant

T_m..... motor torque

T_L..... load torque

T_d..... disturbance torque

V_a.... armature drive voltage

V_b.... back emf voltage

θ_m.... motor shaft angular displacement

θ_L.... load angular displacement

ω_m... motor shaft angular velocity

ω_L.... load angular velocity

2.3. Yaw Open Loop Velocity Transfer Function

The first step in designing the analog yaw closed-loop velocity controller was to develop a model of the yaw open-loop velocity transfer function. The magnitude Bode plot of the actual yaw dynamics, shown by the x's in figure 2.1, was obtained using a 0.02 to 6.0 Hz sine sweep. Due to non-linearities present in the Link drive, the corresponding phase data could not be obtained accurately from the measurements. However, assuming a minimum phase system, the Bode gain-phase theorem allows the transfer function phase to be reconstructed from the magnitude plot. The validity of the minimum phase assumption will be addressed in the following section.

2.4. Modeling the Link Harmonic Drive

In order to determine the order of the transfer function to fit to the actual magnitude data shown in figure 2.1, the Yaw Harmonic Drive was modeled as an armature controlled dc motor and gear reducer assembly, as shown in figure 2.2. The armature controlled dc motor utilizes a constant field current (I_f), and therefore the motor torque is

$$T_m(s) = K_m I_a(s) \tag{2.1}$$

Using the node method, the relation between the armature current and input voltage is

$$V_a(s) = (R_a + L_a s) I_a(s) + V_b(s) \tag{2.2}$$

where $V_b(s)$ is the back electromotive-force (emf) voltage proportional to motor speed. That is

$$V_b(s) = K_b(s) \omega_m(s) \tag{2.3}$$

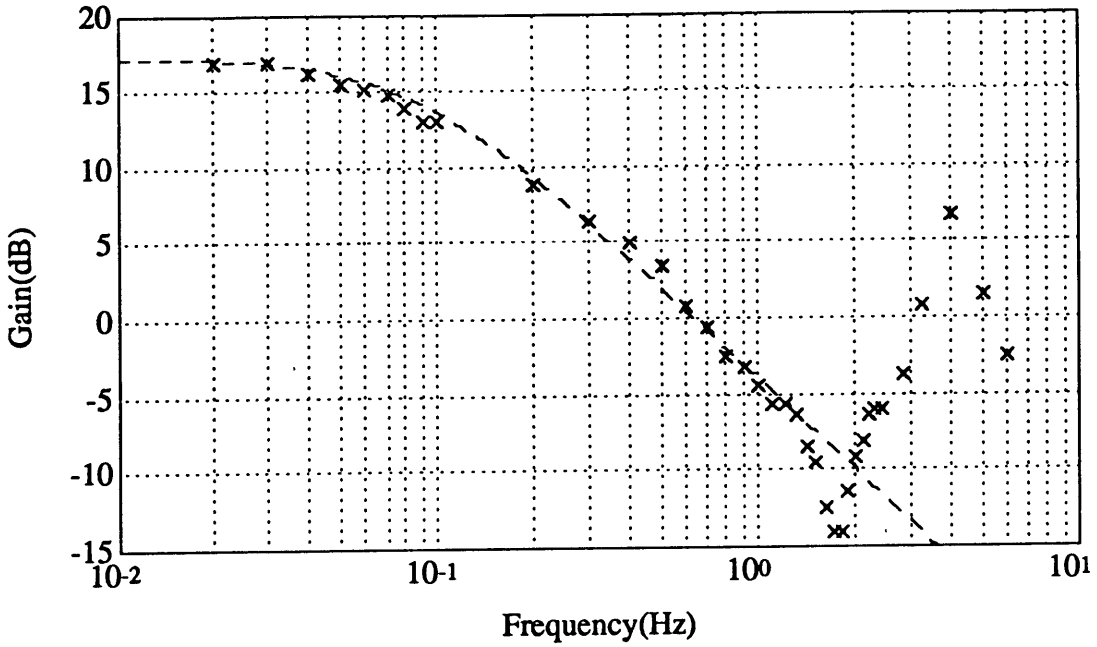


Figure 2.1: Open-Loop Yaw Velocity Transfer Function and First Order Fit.

Substituting (2.3) into (2.2) and solving for $I_a(s)$, we have

$$I_a(s) = \frac{V_a(s) - K_b\omega_m(s)}{(R_a + L_a s)} \tag{2.4}$$

The load torque on the motor due to the Link and gear reducer assembly is given by

$$T_L(s) = n(Js + f)\omega_m(s) = T_m(s) - T_d(s) \tag{2.5}$$

Figure 2.3 shows equations (2.1), (2.4), and (2.5) in a block diagram for the armature controlled dc motor. Using standard block diagram reduction techniques, the transfer function for this system is

$$\frac{\omega_L(s)}{V_a(s)} = \frac{K_m}{(R_a + L_a s)(Js + f) + \frac{K_b K_m}{n}} \tag{2.6}$$

The time constant for the armature (τ_a) is given by

$$\tau_a = \frac{L_a}{R_a} \tag{2.7}$$

For many armature controlled dc motors, the armature dynamics are *much* faster than those of the load. In the case of the Link we have

$$\tau_a = \frac{L_a}{R_a} = \frac{100 \cdot 10^{-6} H}{0.75\Omega} = 1.33 \cdot 10^{-4} s \tag{2.8}$$

This is very fast in comparison to the slow dynamics of the Link, and thus the armature dynamics can be safely neglected; the resulting transfer function is

$$\frac{\omega_L(s)}{V_a(s)} = \frac{\frac{K_m}{R_a f + \frac{K_b K_m}{n}}}{\left(\frac{R_a J}{R_a f + \frac{K_b K_m}{n}}\right) s + 1} \tag{2.9}$$

Inspection of the transfer function (2.9) yields two important results. First, modeling suggests that the system should be minimum phase (the modeled system has no zeroes or time delays); with this assumption, we can reconstruct the phase Bode plot from the

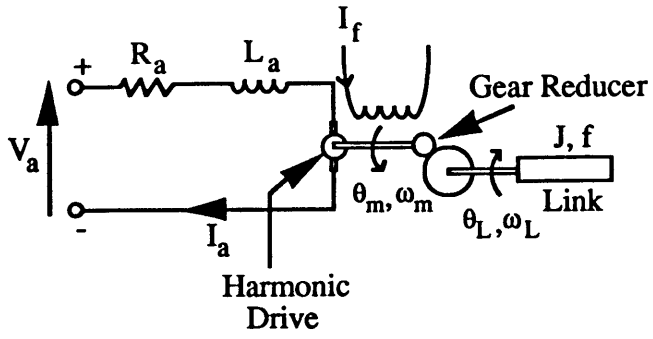


Figure 2.2: Model of Yaw Harmonic Drive.

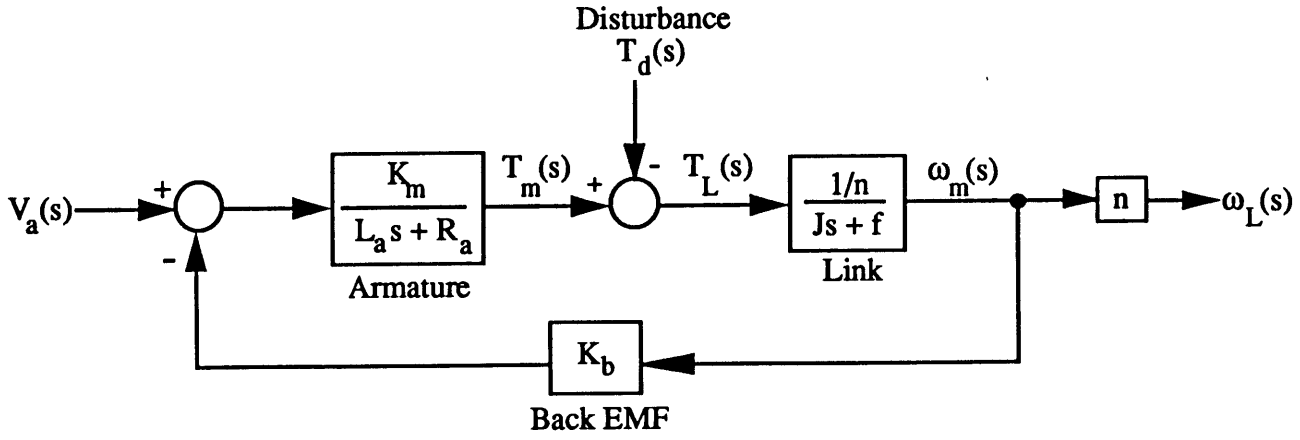


Figure 2.3: Block Diagram of Armature Controlled dc Motor.

magnitude Bode plot. Second, the appropriate transfer function fit for the actual magnitude Bode plot appears to be first order, and is shown by the dashed line in figure 2.1. The approximate transfer function (from input voltage to tachometer output voltage) is

$$\frac{\omega_L(s)}{V_a(s)} = \frac{7.3}{1.818s+1} \tag{2.10}$$

Figure 2.4 shows the open-loop Bode plot for this system for this fit to the Link yaw open-loop dynamics.

2.5. Yaw Closed-Loop Controller

The closed-loop feedback structure which will be used for the yaw controller is shown in figure 2.5. This section details the design of K(s), the yaw velocity compensator.

2.5.1. Design Specifications

The primary design specification for the yaw compensator was to achieve less than 10 percent tracking error for unit sine waves below 1 Hz (a reasonable frequency bound for manual control experiments). The 10 percent tracking error, it was felt, could be compensated for by increasing the velocity-to-volts conversion factor in the Link control software.

For the closed-loop system shown in figure 2.5, the sensitivity transfer function is given by

$$\frac{e(s)}{r(s)} = \frac{1}{1+K(s)G(s)} \tag{2.11}$$

where K(s) is the compensator and G(s) is the open-loop Link yaw velocity transfer function. For large |K(s)G(s)| and unit sine inputs for which |r(s)|=1, (2.10) becomes

$$|e(s)| = \frac{1}{|K(s)G(s)|} \leq 0.10 \tag{2.12}$$

Rearranging and solving for |K(s)G(s)|, we get the tracking performance requirement

$$|K(s)G(s)| \geq 10.0 \tag{2.13}$$

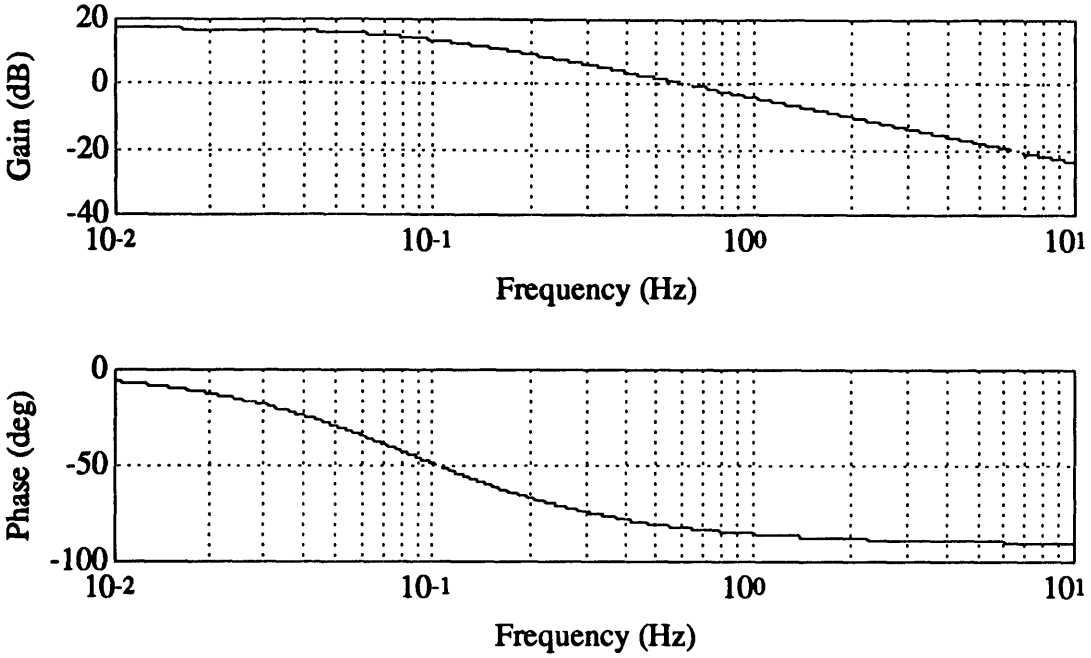


Figure 2.4: Uncompensated Yaw Open-Loop Bode Plot.

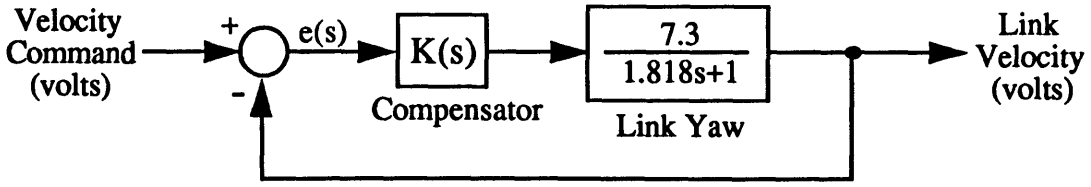


Figure 2.5: Yaw Controller Block Diagram.

2.5.2. Constant Gain Compensation

The inherent simplicity of the first order plant lends itself well to constant gain compensation, i.e. where K(s)=K. Substituting for G(s) in (2.13), rearranging and solving for K, we get the constant gain tracking performance requirement

$$K \geq \frac{10.0}{|G(s)|} = 10.0\frac{\sqrt{(1.818*\omega)^2+1^2}}{7.3} \qquad (2.14)$$
$$= 10.0\frac{\sqrt{(1.818*2*\pi)^2+1^2}}{7.3} = 15.71$$

Selecting K=16 to satisfy (2.14), the closed-loop yaw velocity transfer function for figure 2.5 is

$$\frac{\omega_L(s)}{V_a(s)} = \frac{116.8}{1.818s+117.8} \qquad (2.15)$$

Figure 2.6 shows a Bode plot of this system. Note the nearly 0 dB gain and 0 degree phase region extending out through 1 Hz characteristic of good closed-loop tracking performance.

2.5.3. Analog Implementation

The controller designed above was implemented in analog as shown in figure 2.7, where

$$R_1 \cong R_2 \cong R_6 \cong R_7 \cong 10k\Omega$$
$$R_3 \cong R_4 \cong 1k\Omega$$
$$R_5 \cong 16k\Omega$$

The tach signal is inverted and summed with the computer command to generate an error signal, which is then amplified by a factor of approximately 16. Since the error signal is inverted during the amplification stage, the last stage again inverts the signal to return to the proper polarity. This is the circuit currently used to control the Link yaw axis.

2.6. Roll and Pitch Closed-Loop Controller

Due to the gravitational perturbation torque present in the roll and pitch axes, the compensator gain was increased to K=20 in order to achieve good command following at

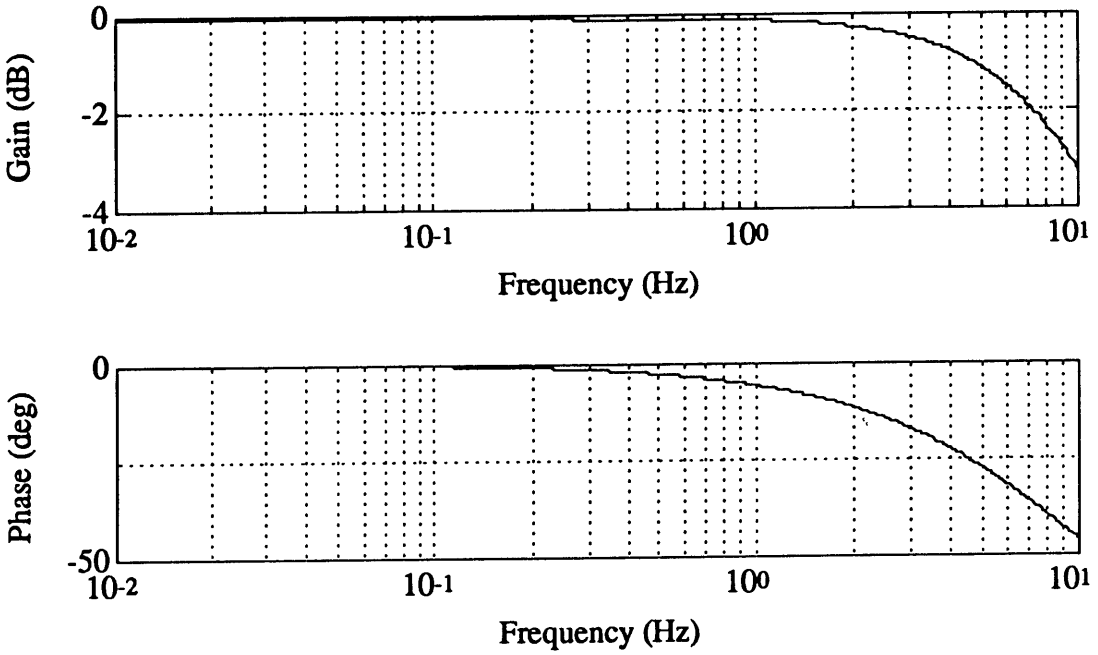


Figure 2.6: Model Gain Compensated Yaw Closed-Loop Dynamics (K=16).

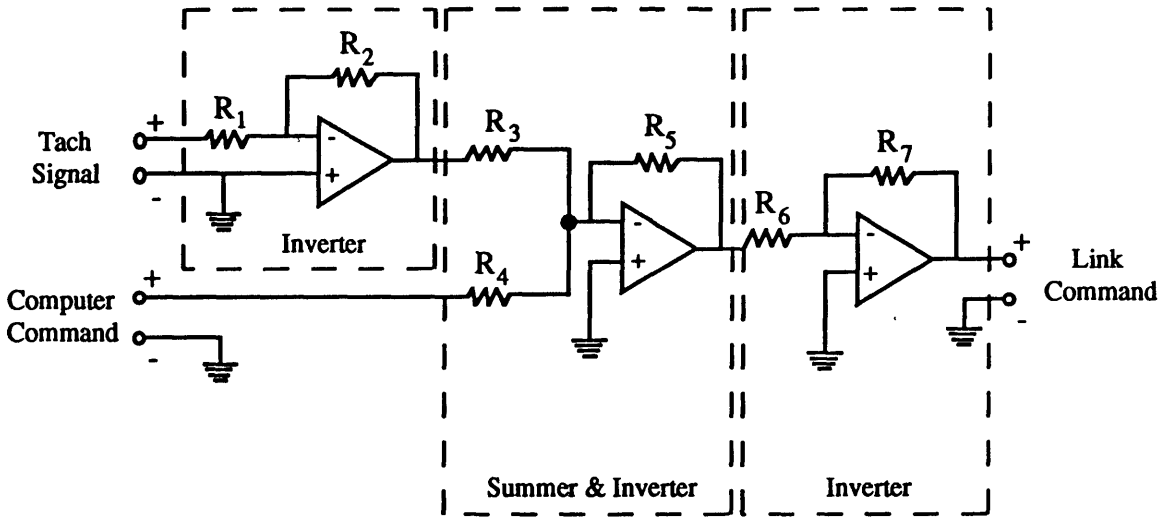


Figure 2.7: Diagram of Analog Constant Gain Compensator Circuit.

low frequencies. The circuit design is the same as Figure 2.7, but with the following values for the resistances:

$$R_1 \cong R_2 \cong R_6 \cong R_7 \cong 10k\Omega$$
$$R_3 \cong R_4 \cong 1k\Omega$$
$$R_5 \cong 20k\Omega$$

2.7. Closed-Loop Link Performance

This section discusses closed-loop Link performance using the compensators designed above. Gain and phase closed-loop Bode plots for the roll, pitch, and yaw axes were obtained using a closed-loop sine sweep and Fast Fourier Transform. The transfer function plotted in each case is from the command to tachometer output (volts-to-volts).

2.7.1. Roll Axis

Figure 2.8 shows the closed-loop Link dynamics for the roll axis. Note the region of nearly 0 dB gain and 0 degree phase extending out through 1 Hz characteristic of good command following.

2.7.2. Pitch Axis

Figure 2.9 shows the closed-loop Link dynamics for the pitch axis. Once again, the requirement for good command following below 1 Hz has been achieved.

2.7.3. Yaw Axis

Figure 2.10 shows the Link yaw closed-loop dynamics. Again the requirement for good command following below one Hertz has been achieved. However, note the "valley" between 1 and 2 Hertz, where the magnitude rapidly drops and recovers. This appears to be caused by play in the rubber drive belts connecting the yaw motor to the Link chassis. At frequencies above 1 Hz, the slow Link dynamics are essentially unexcited, and thus the motor is "unloaded." Since the tachometer measures motor velocity instead of Link velocity, measurements are of the fast motor dynamics. This explains the gain recovery. Similar behavior is seen in the roll and pitch axes, although

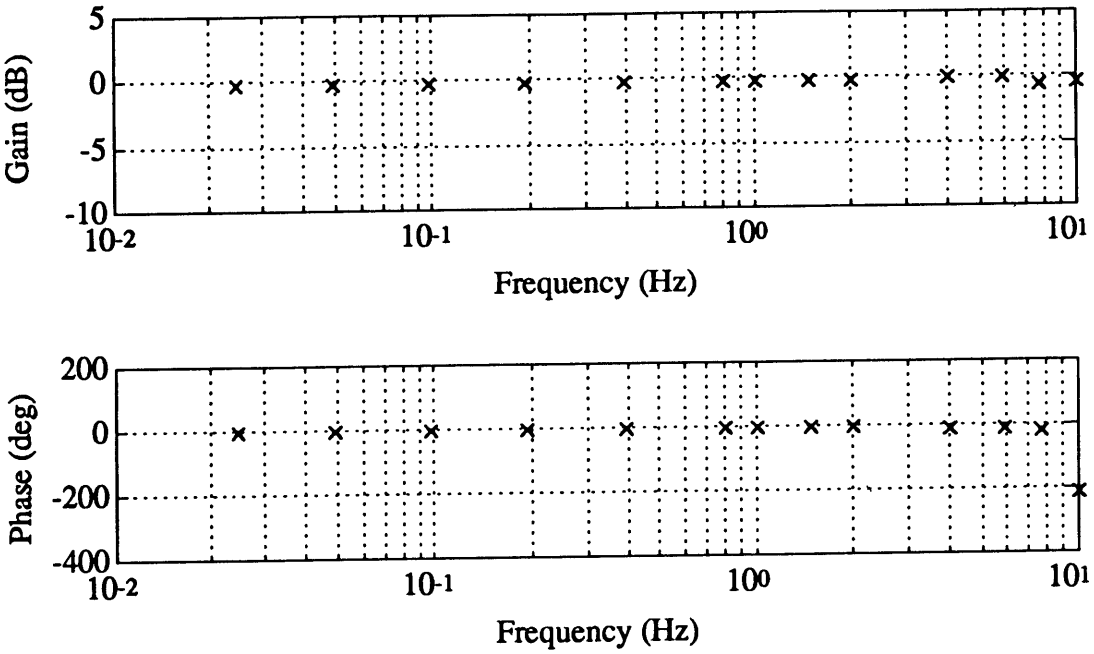


Figure 2.8: Link Gain Compensated Roll Closed-Loop Dynamics (K=20).

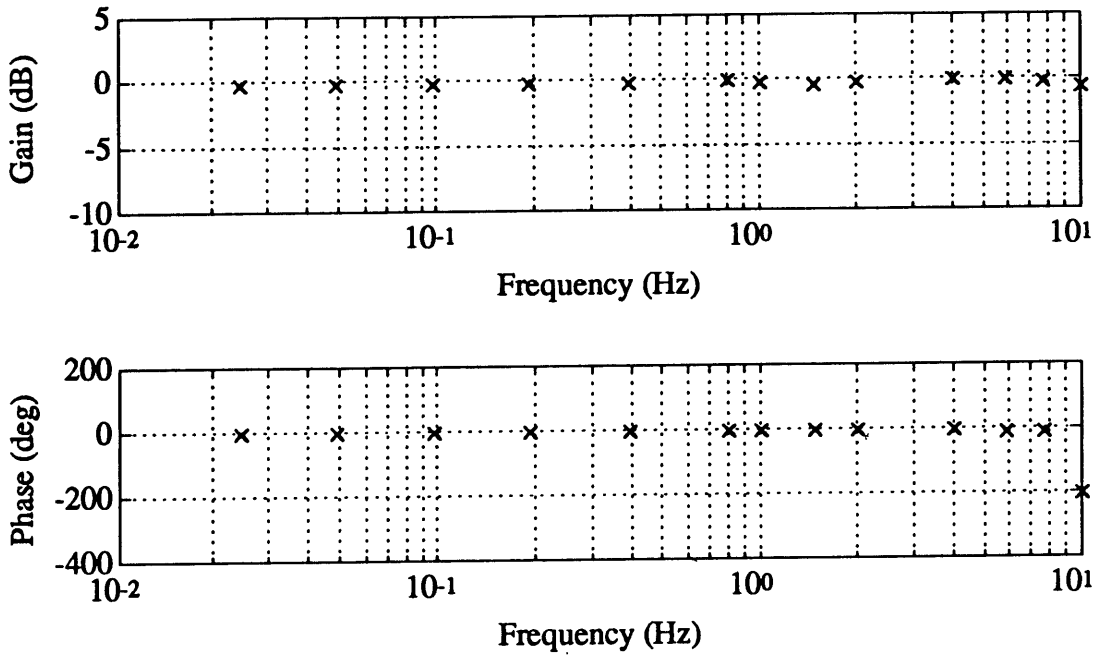


Figure 2.9: Link Gain Compensated Pitch Closed-Loop Dynamics (K=20).

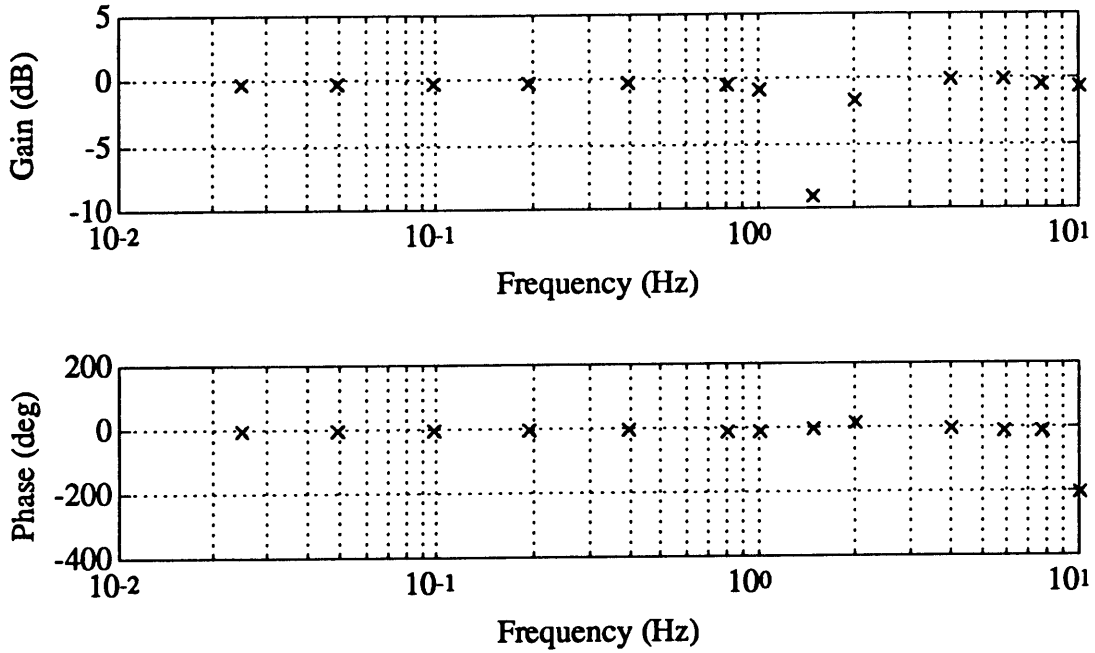


Figure 2.10: Link Gain Compensated Yaw Closed-Loop Dynamics (K=16).

much less pronounced. This is most likely due to the fact that steel cables connect the motors to the Link chassis rather than rubber belts, thus reducing the play.

2.8. Summary

This chapter has discussed the design of the analog roll, pitch, and yaw closed-loop velocity controllers for the Link. Constant gain compensation was chosen as the control technique, with a gain of K=20 for the roll and pitch axes, and K=16 for the yaw axis. The controllers were implemented using the circuit shown in figure 2.7. As can be seen in the closed-loop Bode plots in figures 2.8, 2.9, and 2.10, the performance requirement for good command following for sine waves with frequency below 1 Hz was achieved for all three axes.

3. SUPPLEMENTARY EQUIPMENT FOR THE LINK TRAINER

3.1. Position Control of the Link Trainer

In addition to the velocity controllers designed in chapter 2, a constant gain compensation position controller was designed for the roll axis. Since open-loop position dynamics were not available in roll , the compensator gain was manually adjusted until good command following was achieved. Figure 2.7 in chapter 2 is a diagram of the circuit used for position control, with the following component values:

$$\begin{aligned} &R_1 \cong R_2 \cong R_6 \cong R_7 \cong 10k\Omega \\ &R_3 \cong R_4 \cong 1k\Omega \\ &R_5 \cong 5k\Omega \end{aligned} \qquad (3.1)$$

In addition to the component changes, note that the Link tach signal input in figure 2.7 is replaced by the Link roll position potentiometer signal to facilitate closed-loop following of the position command. As before, the computer command is supplied by the Link software main channel. Figure 3.1 shows a Bode plot of the Link position transfer function using this constant gain compensation controller.

3.2. Velocity Control of the Projection System

Two pieces of equipment were built in order to operate the projection system closed-loop: a (1) constant gain compensation velocity servo, and a (2) high slew rate power amplifier. Figure 3.2 shows a Bode plot of the projector velocity transfer function using this constant gain compensation controller and power amplifier, each of which are discussed below. The larger than expected phase lag of the projection system is most likely due to the deadzone of the drive motor.

3.2.1. Velocity Controller

The velocity controller for the projection system was identical to the circuit shown in figure 2.7 in chapter 2, with the following component values:

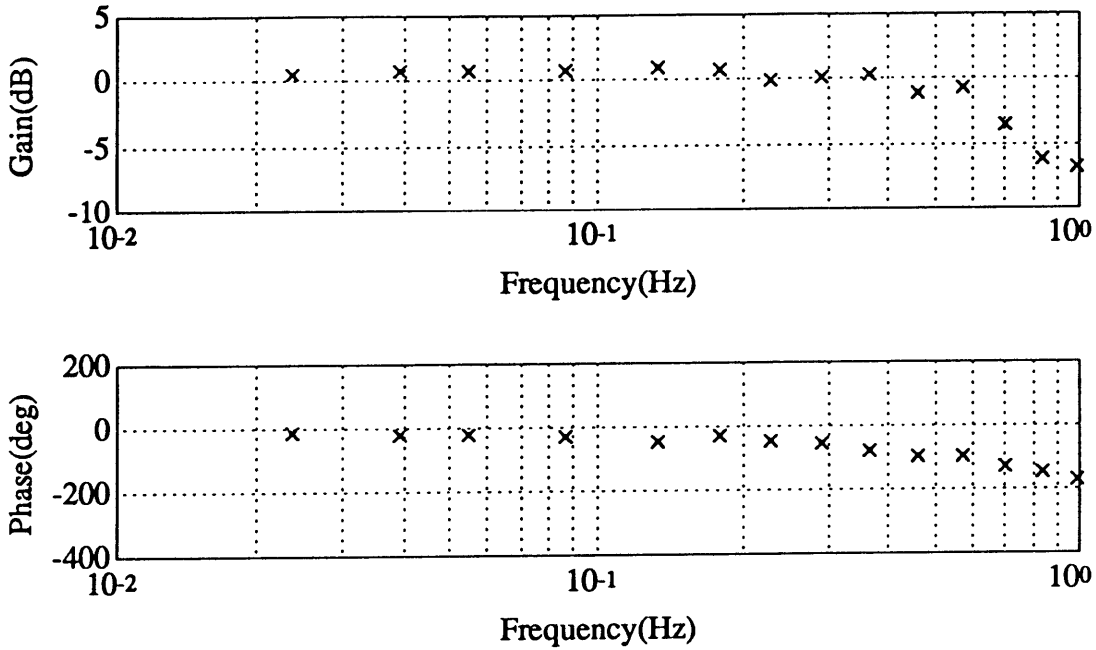


Figure 3.1: Link roll gain compensated position closed-loop dynamics.

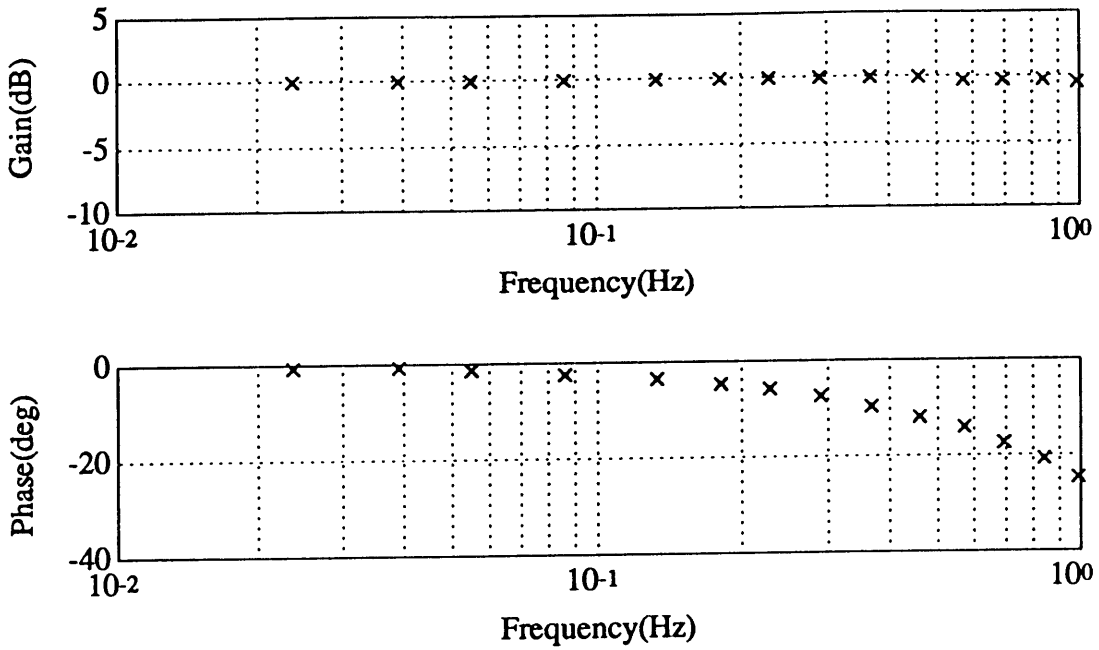


Figure 3.2: Projector gain compensated velocity closed-loop dynamics.

$$R_1 \cong R_2 \cong R_6 \cong R_7 \cong 10k\Omega$$
$$R_3 \cong R_4 \cong 1k\Omega \qquad (3.2)$$
$$R_5 \cong 50k\Omega$$

The overall circuit gain of 50 was selected since it supplied good closed-loop following out through 1 Hertz, the range of interest for the manual control experiments contained within this thesis. The tach signal labeled in figure 2.7 comes from the tach on the projector motor, and the computer command is supplied by the Link software auxiliary channel.

3.2.2. Power Amplifier

Since the velocity controller is unable to supply the current necessary to drive the projection system motor, a high slew rate power amplifier was needed. A diagram of the circuit which was used is shown in figure 3.3. Note that the 47 Ω resistor must be a minimum 10 W resistor, since it must dissipate a large amount of current, especially when the power amplifier is operated without an external load. The 15 Ω resistive load is the expected operating load, and represents the projection system drive motor.

3.3. Filtering of the Data Acquisition Signals

Six pole Bessel filters (10 Hz break frequencies) were selected for filtering of the data acquisition signals since they supply minimum phase lag while still having rapid roll-off characteristics. Figure 3.4 shows the circuit diagram for the filters, including the values for all the components. The voltage divider first stage corrects for the gains of the following three stages, so that the entire filter is unity gain. Four signals are currently filtered: Link position and velocity, subject control wheel command, and projection system stripe velocity.

3.4. Control Wheel Circuitry

Since Link onboard power has a +/- 15 volt range, a voltage divider circuit had to be built for the control wheel to bring its output within the +/- 10 volt range acceptable to the computer. This circuit is shown in figure 3.5, along with a diagram of the 15 pin

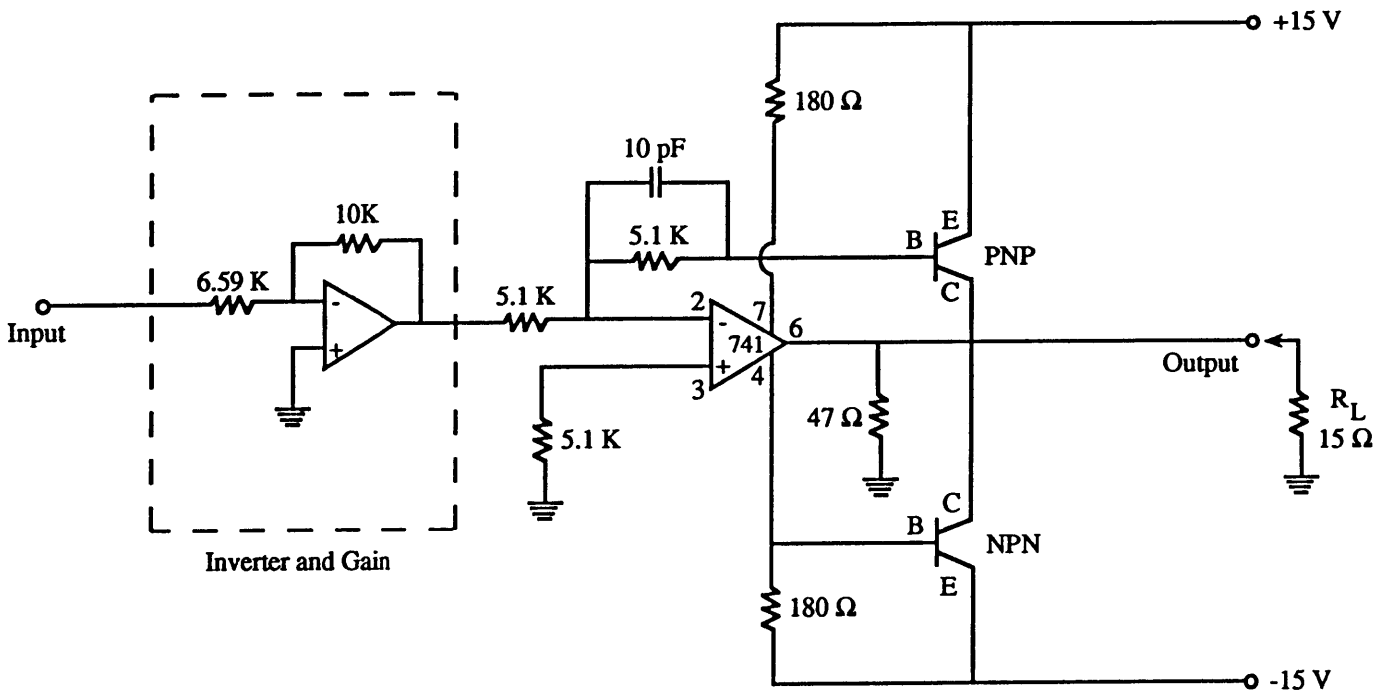


Figure 3.3: High slew rate power amplifier.

connector which allows it to be removed easily from the Link. The following is a list of pin-outs for the connector:

1. +15 volts (from onboard supply)
2. Link power ground
3. Roll command out
6. Pitch command out
7. -15 volts (from onboard supply)
4, 5, 8-15. Unused

3.5. Counterrotating Visual Field Circuitry

In order to perform the trials with the counterrotating visual field (confirming visual cue), a circuit had to be built which would allow the operator to choose between control of the visual field by the auxiliary channel, or by the sled tachometer signal (with appropriate scaling). To compute the appropriate scaling for the tach signal, we will need the following Link scalefactors:

1. Projector Command : 0.7602 volts/deg/s
2. Link Tach Voltage : 0.715 volts/deg/s

Now assume a 7 deg/s Link velocity, which corresponds to a Link tach signal of 7*0.715=5 volts. Now the projector command required to generate a stripe velocity of 7 deg/s is 7*0.7602=5.32 volts. Thus the required gain on the Link tach signal is 5.32/5=1.064. A non-inverting circuit which can supply this gain and allow the operator to choose whether the projection system is controlled by the Link computer or operating in counterrotating mode is shown in figure 3.6.

3.6. Sled Software Modifications

In order for the MVL Sled software to be used with the Link, modification of the software was required. These modifications were primarily of three types:

1. Changing from linear units to angular units.
2. Replacing Sled constants with appropriate constants for the Link.

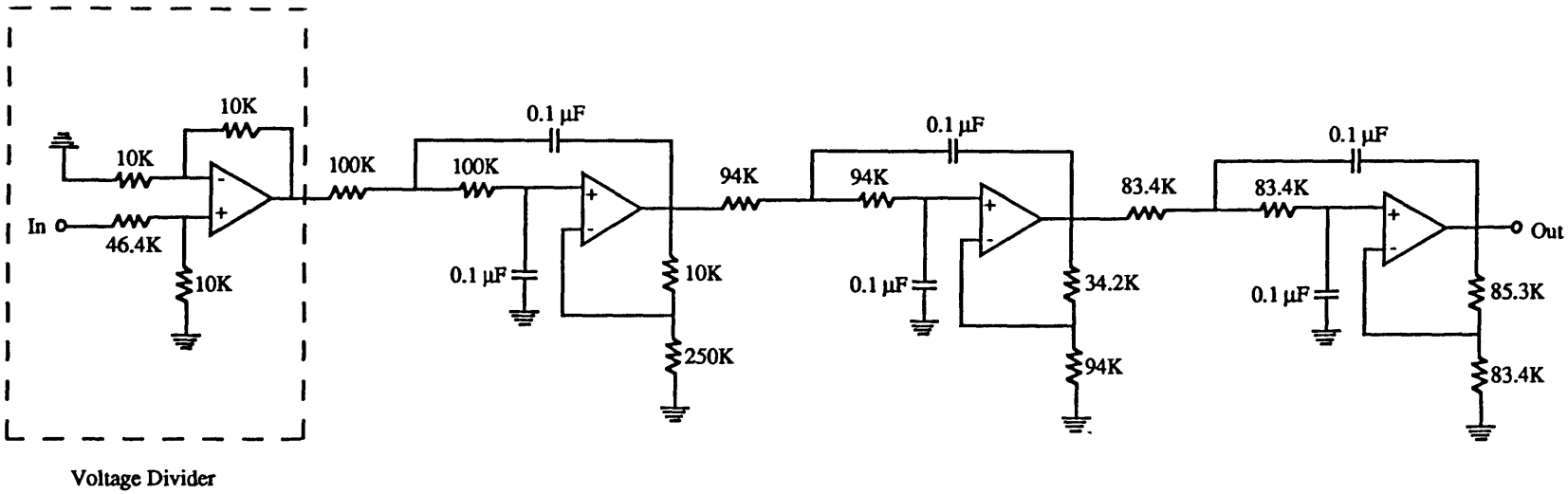


Figure 3.4: Six pole Bessel filter with voltage divider.

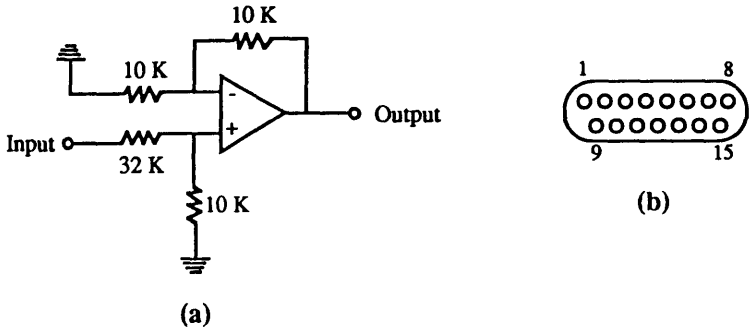


Figure 3.5: Control wheel circuitry (a) and pin connector layout (b).

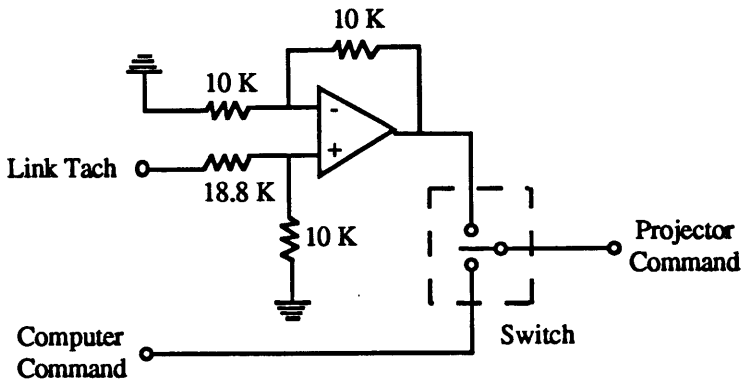


Figure 3.6: Counterrotating visual field circuitry.

3. Developing sum of sines profile generators for the main and auxiliary control channels.

The following three sections summarizes what was involved for each of these tasks.

3.6.1. Linear to Angular Units

Since all of the sled software menus and output windows use linear units (m, m/s, m/s/s, G) it was necessary to modify the routines to output angular units (deg, deg/s, and deg/s/s). This primarily required a large amount of window reformatting since the number of characters in the angular unit names is more than in the linear unit names.

3.6.2. Sled Constants to Link Constants

Since the Link uses different position and velocity scalefactors, and maximum position, velocity, and acceleration abort constants than the sled, these had to be modified in the Link software. The current version of the Link software was created for the roll axis and is called *Linkr*. In addition, since the Link has no on-board accelerometer, the sections of the code which process the sled accelerometer signal were removed.

3.6.3. Sum of Sines Profile Generators

Two types of sum of sines profile generators were written for the Link main and auxilliary control axes (four routines total). The first set of routines allows for double lead-lag filtering of the sine wave amplitudes (this allows a profile to be created with a more gradual transition between the large low frequency amplitudes and the small high frequency

amplitudes). The second set allows the user to enter the amplitudes manually, allowing any sum of sines profile to be created. In both cases, the user has control over period, number of sines (20 maximum), equal amplitude domain (position, velocity, or acceleration), maximum command (volts or degrees/s), frequency of each sine wave, and amplitude of each sine wave. A more detailed description of these routines, including C++ code listings, can be found in appendix A.

PART TWO

INFLUENCE OF THE VISUAL FIELD ON MANUAL ROLL STABILIZATION

4. EXPERIMENTAL METHOD FOR THE ROLL EXPERIMENTS

4.1. Experimental Apparatus

This section discusses the motion platform, projection system, subject control wheel, and supplementary equipment used in the manual roll stabilization experiment.

4.1.1. Motion Platform

The enclosed platform used to rotate the subject is a modified small aircraft trainer, the Link GAT-1 Trainer, driven in roll rotation only as a position servo. The entire system is controlled by a position controller using potentiometer feedback. Position commands to the controller are generated by a Northgate 386 computer running a modified version of a special software package developed for the MVL by Payload Systems, Inc. The position commands follow mathematical trajectories which can be combined with a feedback signal controlled by the subject. The maximum roll tilt angle of the trainer is 15 degrees in either direction. (Section 3.1 in chapter 3 details the design of the position controller, and figure 3.1 in chapter 3 shows a Bode plot of the roll closed-loop dynamics for position control.)

To restrain the subject during roll, there are lap and shoulder belts. The subject's head is fixed using chin and forehead straps, which can be adjusted to fit the subject's head. With these attachments the subject is firmly restrained.

4.1.2. Projection System

A modified Kodak slide projector fitted with a motor driven infinite film loop (mounted on the roof of the Link GAT-1 trainer) is used to provide the visual field motion. Via an arrangement of beam splitters and mirrors, a set of horizontal stripes are projected onto the two translucent side windows of the trainer. For this experiment, the optics are arranged so that as the pattern moves downward on one side window, it moves upward on the other, mimicking rotational movement about the roll axis. When the subject is seated in the trainer looking forward with his head restrained by the headrest, each of the side

windows subtend approximately 64 degrees horizontally (52 degrees forward, 12 degrees aft), and 50 degrees vertically (16 degrees up, 34 degrees down). The alternating black and white stripes projected on the windows each subtend an angle of approximately 6 degrees. The windows are fixed with respect to the subject. The closed-loop system which moves the film loop is driven by the MVL Link software auxiliary channel. (Section 3.2 in chapter 3 discusses the design of the velocity controller and figure 3.2 in chapter 3 shows a Bode plot of the closed-loop projector dynamics for velocity control.)

4.1.3. Subject Control Wheel

A dashboard mounted subject control wheel was used for nulling trainer roll position. The control wheel was circular (8.5 inch diameter, 1/2 inch thick), and had a featureless surface which provided neither visual nor tactile cues as to center and hence zero commanded position. Mechanical stops limit wheel deflection to 60 degrees in either direction, and trainer roll direction corresponds to the direction the wheel is turned. (The circuitry used in this controller is discussed in section 3.4 of chapter 3.)

4.1.4. Supplementary Equipment

A Macintosh II running LabView™ is used for data acquisition, with data sampled at 40 Hertz. Trainer position and velocity were recorded, along with control wheel position and visual field velocity. All four signals were anti-alias filtered prior to data acquisition using 6 pole Bessel filters. (The details of the filters are discussed in section 3.3 of chapter 3.) All subjects wore a set of headphones allowing for two-way communication with the operator. In an attempt to reduce external audio cues (e.g. motor sounds, cable slap, etc.), all subjects wore earplugs and white noise was applied to the headphones at a volume comfortable to the subject. A red LED was mounted on the trainer dashboard to serve as a fixation point for the subject in all the trials. The brightness of the LED was adjusted to a level comfortable to the subject in the dark.

4.2. Subjects and Stimulation

4.2.1. Subjects

A total of six subjects participated in the experiment, 3 males and 3 females, age 21 to 32, in normal health. None of the subjects had previous experience in roll manual control, but one subject was a private pilot.

4.2.2. Vestibular Stimulation

The movement command sent to the Link is identical for all trials, and is a pseudo-random zero-mean sum of sines position command with the parameters shown in table 4.1. A plot of trainer position in response to the disturbance is shown in figure 4.1a.

4.2.3. Visual Stimulation

There were five different types of visual fields used in the experiments: (1) DARK trials were in the dark with only the red LED visible to the subject (dependence was solely on vestibular cues); (2) FIX supplied a stationary visual field with respect to the subject (again dependence was on vestibular cues); (3) CON provided a visual field which rotated in the direction opposite to that of the Link but with the same velocity (these confirming visual cues mimic what we see in everyday life); (4) CV provided a constant velocity visual field at 10 degrees/second (clockwise or counterclockwise); (5) SS was a pseudo-random zero-mean sum of sines velocity command with the parameters shown in table 4.2. In the case of SS, as shown in figure 4.2, the visual stimulus frequencies are interleaved among the movement stimulus frequencies; this ensures that the two stimuli are completely uncorrelated. A plot of the visual field velocity in response to the disturbance is shown in figure 4.1b.

4.3. Experimental Procedure

Each subject was given the following instructions: "Your task is to keep the trainer as erect as possible by concentrating on your sensed roll position. Throughout the trials, please keep your gaze fixed on the red LED on the dashboard in front of you."

Sum of Sines Vestibular Disturbance

1. Duration of profile: 204.8 seconds
2. Fundamental Frequency: 0.0048 HZ
3. Number of Sinusoids: 12
4. Successive Phase Angle: 37 degrees
5. Maximum Tilt Angle: 12 degrees
6. Maximum Angular Velocity: 10.0 degrees/second

Frequency (Hz)	Position (deg)	Velocity (deg/s)	Phase (degrees)
0.014	2.3995	0.2208	0
0.024	2.3988	0.3679	37
0.053	2.3808	0.8034	74
0.083	2.2982	1.1986	111
0.112	2.1044	1.4849	148
0.151	1.7106	1.6269	185
0.200	1.2360	1.5547	222
0.258	0.8709	1.4161	259
0.346	0.6185	1.3473	296
0.434	0.5216	1.4242	333
0.532	0.4786	1.6006	10
0.668	0.4563	1.9181	47

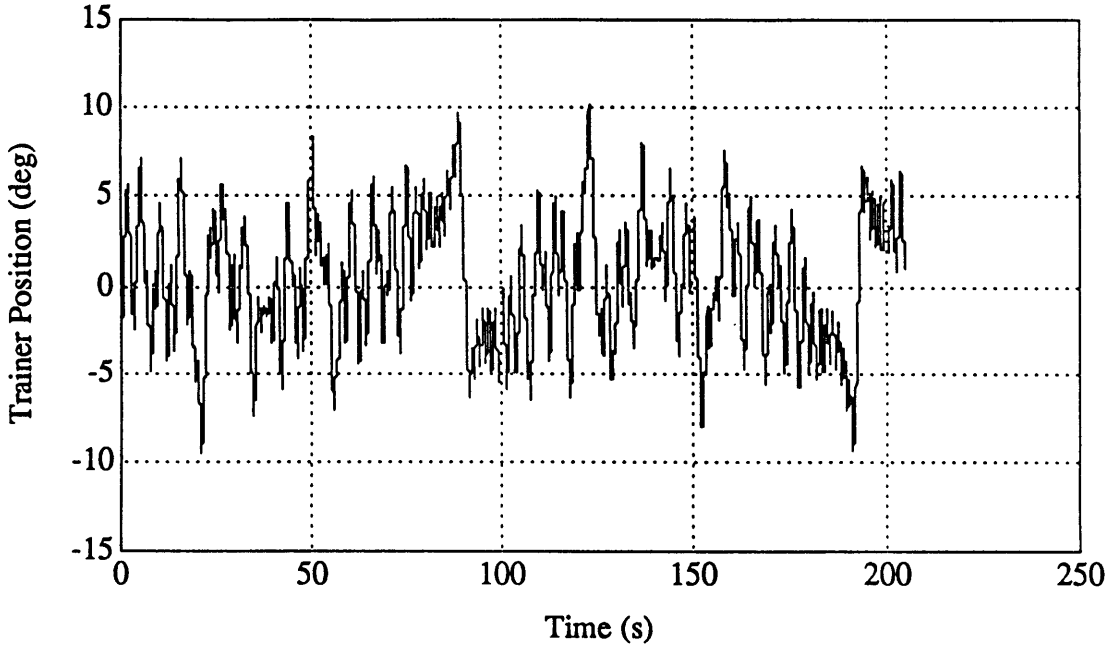
Table 4.1: Sum of Sines vestibular disturbance used in Link manual roll stabilization experiment.

Sum of Sines Visual Disturbance

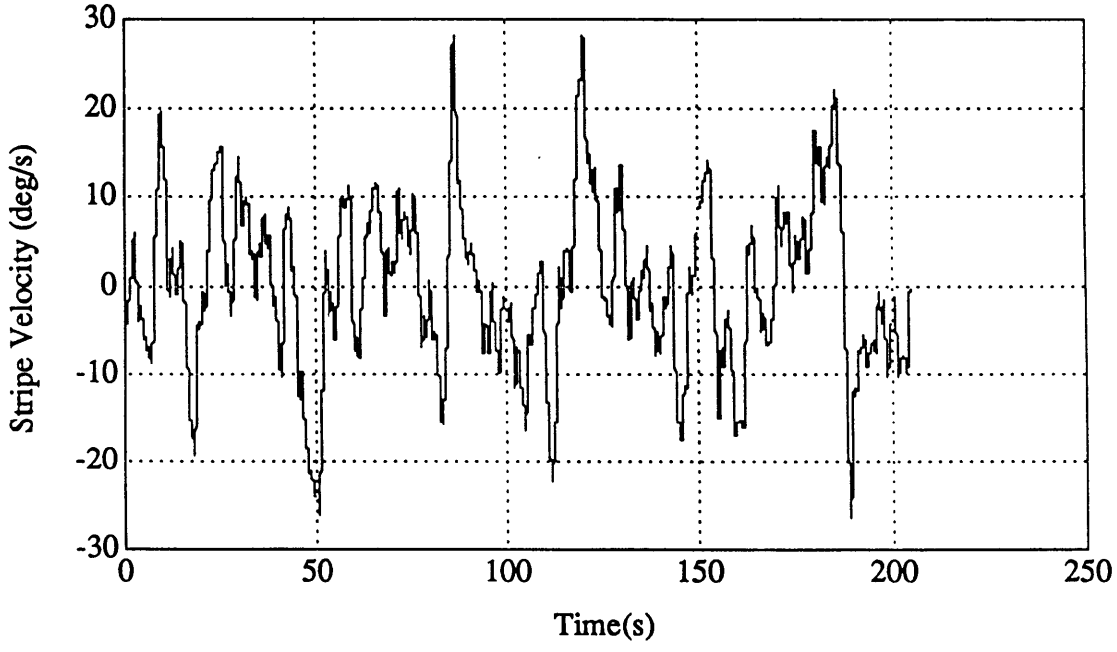
1. Duration of profile: 204.8 seconds
2. Fundamental Frequency: 0.0048 HZ
3. Number of Sinusoids: 12
4. Successive Phase Angle: 37 degrees
5. Maximum Angular Velocity: 27.7 degrees/second

Frequency (Hz)	Position (deg)	Velocity (deg/s)	Phase (degrees)
0.019	48.056	5.737	0
0.034	26.822	5.730	37
0.063	14.276	5.651	74
0.092	9.925	5.737	111
0.141	4.900	4.341	148
0.180	2.980	3.370	185
0.209	2.111	2.772	222
0.297	0.933	1.741	259
0.405	0.512	1.302	296
0.473	0.402	1.195	333
0.620	0.283	1.104	10
0.737	0.233	1.079	47

Table 4.2: Sum of sines visual disturbance used in Link manual roll stabilization experiment.



(a)



(b)

Figure 4.1 : Response of trainer to pseudo-random position disturbance (a) and projection system to pseudo-random velocity disturbance (b).

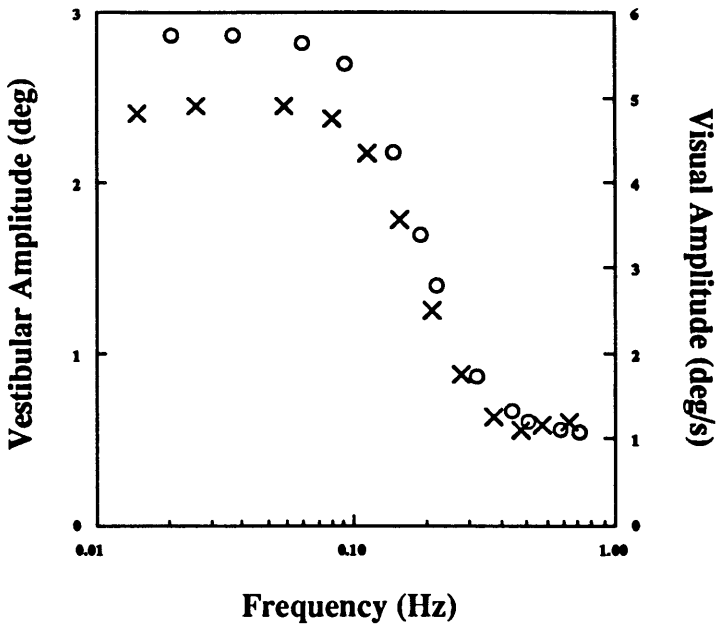


Figure 4.2: Interleaving of vestibular (x's) and visual disturbance components (o's).

Before the experiment, with no wheel input, the disturbance signal was injected into the trainer so that the subject, seated in the trainer, could experience the motion profile he would be asked to null. The white noise volume was then adjusted to a level comfortable for the subject but sufficient to mask external audio cues. The seatbelt and head straps were then tightened, and the head rest was adjusted to fit his head. The subject was then given a practice session with the counterrotating (CON) visual field which provided confirming visual cues. If necessary, subjects were given additional practice sessions with the CON field until they felt comfortable with the nulling task. Only subject D requested an additional run with CON, and her data were not different from that obtained from other subjects.

There were 12 runs for each subject with the following order of visual presentations: DARK, CVL, SS, FIX, CON, SS, FIX, CVR, SS, DARK, SS, and CON. A complete test session took approximately 1 hour.

4.4. Time and Frequency Domain Analysis Methods

This section discusses the time domain and frequency domain analysis methods

used in the manual roll stabilization experiments.

4.4.1. Time Domain

The two time domain measures of subject performance used in this study were mean and RMS trainer position. We used a χ^2 test to investigate whether any of the mean and RMS positions were statistically different from those for the uncompensated vestibular disturbance. Analysis of variance (ANOVA) allowed us to investigate apparent trends in the effect of the visual field on each subject, differences between subjects, and subject population trends. Presentation order effects were not investigated, since the problem was too large for the available statistics package (Systat 5.2™) to solve.

4.4.1.1. Mean and RMS Trainer Position

For each of the 6 subjects who took part in the experiment, a mean and RMS trainer position were computed for each of the 12 runs using the following equations:

$$\bar{\phi} = \frac{\sum_{k=1}^{n} \phi(kT)}{n} \tag{4.1}$$

$$\phi_{RMS} = \sqrt{\frac{\sum_{k=1}^{n} (\phi(kT))^2}{n}} \tag{4.2}$$

where ϕ is the trainer position, k is the sample number, T is the sampling period (in seconds), and n is the total number of samples.

4.4.2. Frequency Domain

The three methods used for investigating the subjects frequency response were the scalar performance measure (SPM), visual response measure (VRM), operator describing function, and open-loop transfer function, each of which is discussed below. We performed an ANOVA on the SPM and VRM data to look for statistically significant trends in the effect of the visual field on each subject, differences between subjects, and subject population trends. As was the case with the time domain measures, presentation order

effects were not investigated, since the problem was too large for the available statistics package (Systat 5.2™) to solve.

4.4.2.1. SPM

One measure of subject performance in the manual roll stabilization task is the non-dimensional normalized scalar performance measure (SPM) developed by Hiltner [11], which is computed using the following formula:

$$SPM = \frac{\sum_{i=1}^{12}(D_1(i) - \phi(i))}{\sum_{i=1}^{12} D_1(i)} \tag{4.3}$$

where $D_1(i)$ and $\phi(i)$ are the amplitude of the input disturbance and actual trainer position (in degrees), respectively, at the i^{th} vestibular frequency. As can be seen from inspection of (4.3), perfect compensation by the subject results in an SPM of 1.0, no subject compensation results in an SPM of 0.0, and over-compensation (energy consistently added to the system by the subject) will result in a negative SPM.

4.4.2.2. VRM

Extending the concept of the SPM to the frequencies in the visual disturbance, we can investigate the degree to which the subject responds to the pseudo-random visual stimulus (SS). Since the subject's task is roll position nulling, we must first determine the visual field position from the velocity disturbance (d_2) using simple integration:

$$p_2(t) = \int_0^t d_2(\tau)d\tau \tag{4.4}$$

Performing a PSD of this signal, we can determine the position amplitudes at the visual disturbance frequencies.

The derived non-dimensional quantity we will define is the visual response measure (VRM), and is computed using the following formula:

$$VRM = \frac{\sum_{i=1}^{12} \phi(i)}{\sum_{i=1}^{12} D_2(i)} \qquad (4.5)$$

where $D_2(i)$ and $\phi(i)$ are the amplitudes of the visual position disturbance (given by a Fourier analysis of 4.4) and actual trainer position (in degrees), respectively, at the i^{th} visual frequency. As can be seen from inspection of (4.5), the larger the component of trainer position at the visual frequencies, the larger will be the VRM, while no subject induced response will result in a VRM of 0.0.

4.4.2.3. Operator Describing Function and Open-Loop Transfer Function

Additional frequency response characteristics for the subject can be obtained from the operator describing function, which can be derived from figure 4.3 (modified from Huang [13]). The figure shows a loop diagram of the closed-loop position nulling task with the operator represented as a linear dual-channel estimator, providing the position estimate for subsequent compensatory wheel deflections. Note that both visual and vestibular cues are assumed to be used by the operator. The operator remnant, n, is defined to be uncorrelated with both of the loop disturbances d_1 and d_2. We can derive the operator's describing function in terms of the three loop inputs d_1, d_2, and n. From conventional block diagram algebra, we have:

$$\lambda(s) = \frac{1}{\Delta}\left(-P_1CE_1d_1 - P_2CE_2d_2 + n\right) \qquad \text{(a)}$$

$$\phi(s) = \frac{P_1}{\Delta}\left(-P_2CE_2d_2 + \frac{1}{K}d_1\right) \qquad \text{(b)} \qquad (4.6)$$

where Δ is defined as:

$$\Delta \equiv \left(\frac{1}{K} + P_1CE_1\right) \qquad (4.7)$$

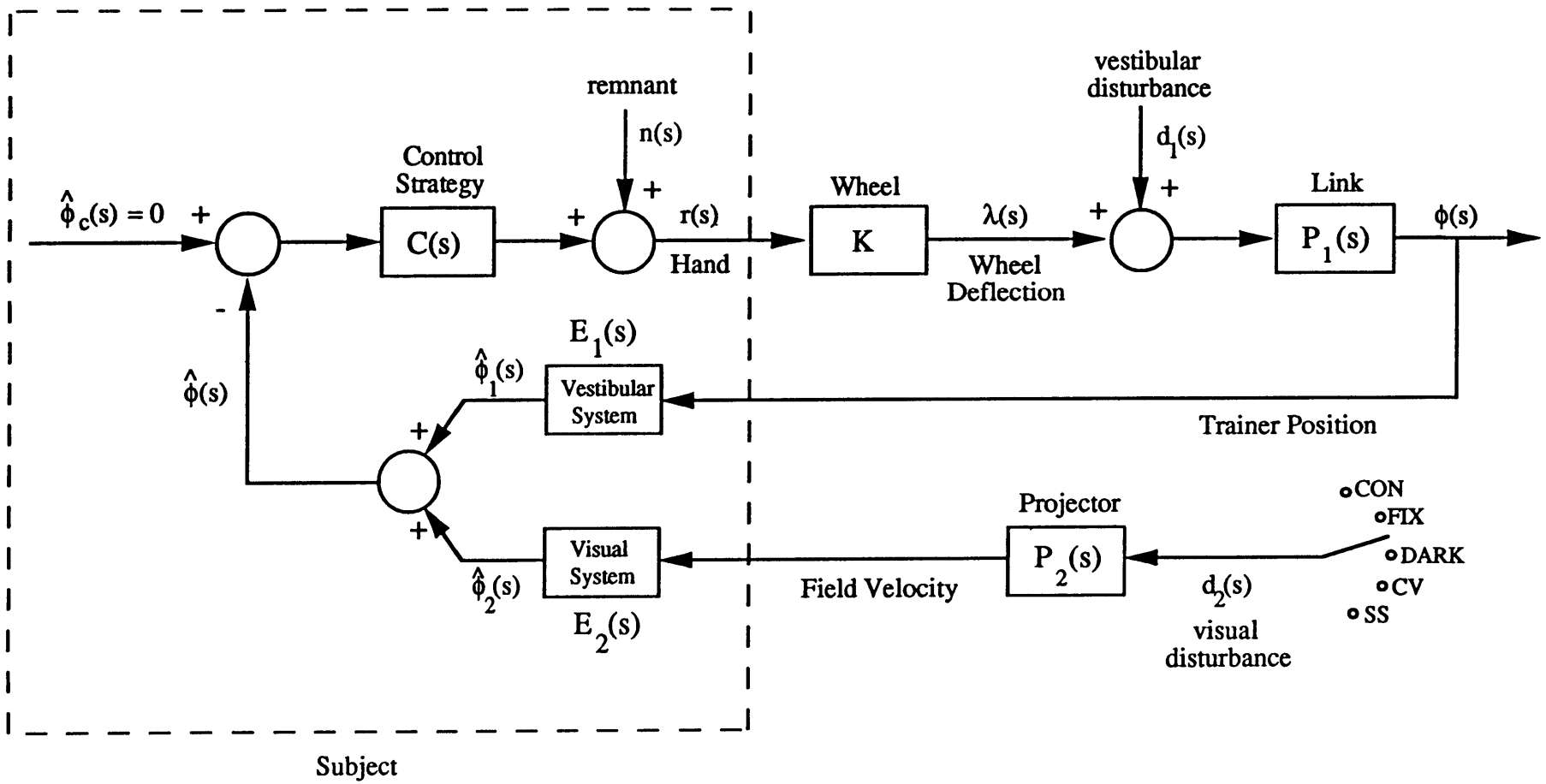


Figure 4.3: Linearized model of closed-loop position nulling.

Correlating d_1 with λ and ϕ and making use of auto and cross-power spectral density functions (see Zacharias for a complete derivation [27]), and operating on (4.6a) and (4.6b), we obtain an expression for CE_1:

$$CE_1 = -K\frac{\Phi_{d_1\lambda}}{\Phi_{d_1\phi}} \tag{4.8}$$

which is the conventional input-output relation defining the operator's describing function. Assuming that the operator's remnant is small with respect to the disturbance injected into the loop, we can rewrite (4.8) as:

$$CE_1(f_i) = -K\frac{\lambda(f_i)}{\phi(f_i)} \tag{4.9}$$

where f_i are the discrete frequencies of the motion stimulus (d_1). This formula allows us to work with conventional Fourier transforms of the operator control wheel signal and trainer position. The describing function given by formula (4.9) is referred to in the sequel as the operator's describing function.

The open-loop transfer function is derived from the operator's describing function by forming the product of (4.9) and the Link dynamics at the vestibular disturbance frequencies (which were obtained using linear interpolation). This derived quantity allows us to compare the results of the manual roll stabilization experiment with those predicted by the Crossover Model.

For both of these transfer functions, a lead-lag transfer function with a pure delay was fit to the data. The fit transfer function had the following form:

$$T(s) = \frac{K(\tau_1 s + 1)}{\tau_2 s + 1} e^{-\tau_d s} \tag{4.10}$$

This form was chosen because it was used by Zacharias and Huang with good results [27,13]. The four parameters were computed by minimizing the weighted squared-error defined by the following formula:

$$SE = \frac{\sum_i \left[\left| CE(j\omega_i) \right|_{fit} - \left| CE(j\omega_i) \right|_{actual} \right]^2}{w_{gain}(j\omega_i)} + \frac{\sum_i \left[\left| \angle(j\omega_i) \right|_{fit} - \left| \angle(j\omega_i) \right|_{actual} \right]^2}{w_{phase}(j\omega_i)} \quad (4.11)$$

where w_{gain} are the gain weights, w_{phase} are the phase weights, and ω_i is the i^{th} vestibular disturbance frequency (gains were in decibels, and phases were in degrees). For those visual fields which had more than one trial, the standard deviation of the gain and phase at each disturbance frequency were used as the gain and phase weights, respectively. If only one trial was available (as with individual fits for CVL and CVR), each of the gain and phase weights were set to 1.0. The MatLab™ optimization routine *OPTIM* was used to perform the minimization. Listings of the m-files can be found in appendix B.

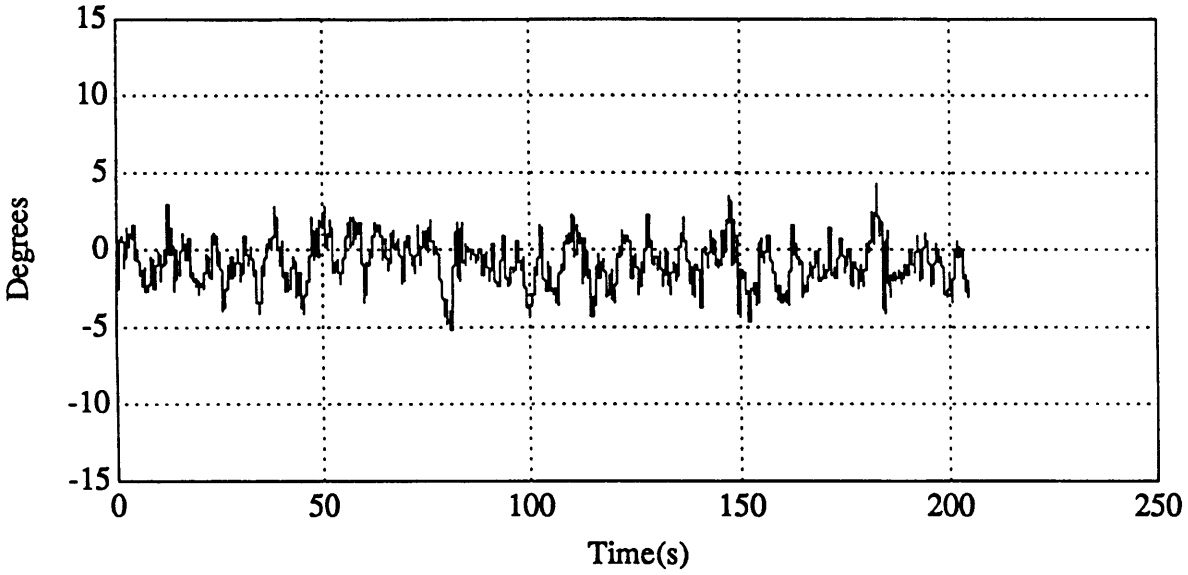
5. RESULTS OF THE ROLL EXPERIMENTS

5.1. Time Domain Analysis of Subject Response

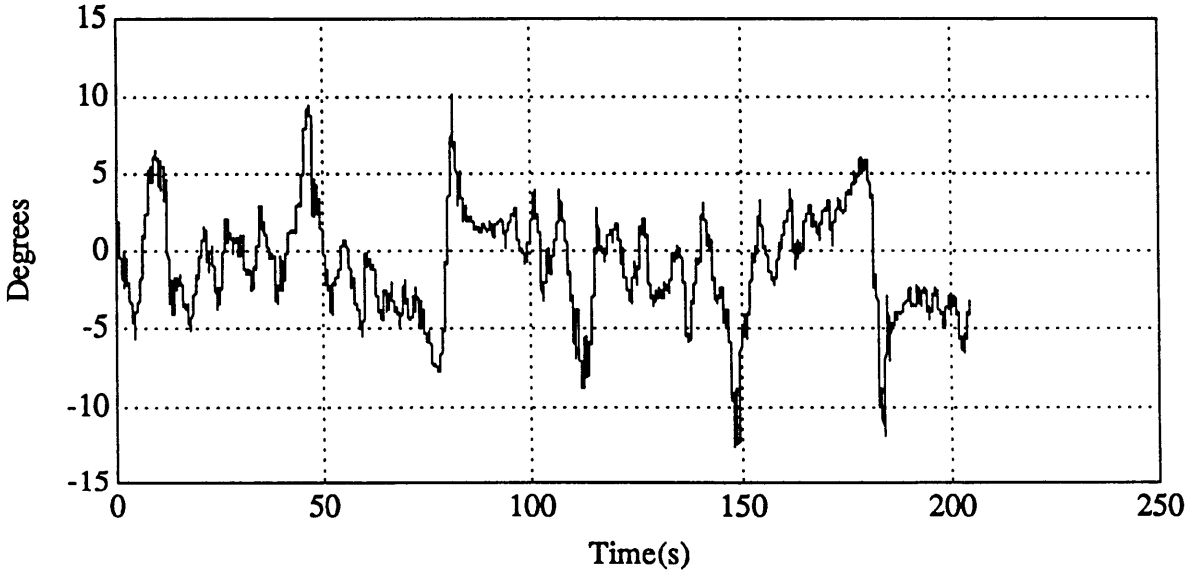
This section presents representative subject response time histories for each of the six different visual fields used in this study and summarizes the individual and population trends seen in the mean and RMS trainer position.

5.1.1. Representative Time Histories

Figure 5.1 shows a set of typical time histories for the trainer position and subject's compensatory control wheel response. Since the visual field is counterrotating (CON), operator performance results in a well-met task objective. The subject reached maximum roll positions of 4.38 and -5.16 degrees, and had a mean position of -0.85 degrees (negative angles indicate leftward roll positions, and positive angles indicate rightward roll positions). Figure 5.2 illustrates subject performance with a constant velocity visual field rotating at 10 degrees/s to the left with respect to the subject (CVL). As a result of roll vection, the subject biased the trainer in the direction of the visual field, resulting in maximum roll positions of 1.91 and -13.07 degrees, and a mean position of -5.85 degrees. Figure 5.3 illustrates subject performance with a constant velocity field rotating at 10 degrees/s to the right with respect to the subject (CVR). The resulting roll vection again caused the subject to bias the trainer in the direction of the visual field, reaching maximum roll positions of 12.39 and -2.18 degrees, and a mean position of 5.22 degrees. Figure 5.4 shows subject performance when the trial is done in the dark, with only a fixation LED visible to the subject (DARK). Performance was slightly worse than with CON but better than with the CV fields, with the subject reaching maximum roll positions of 4.68 and -8.33 degrees, and a mean position of -2.10 degrees. Similarly, figure 5.5 illustrates the subject performance with the fixed visual field (FIX). In this case, the subject reached maximum roll positions of 5.79 and -6.30 degrees, and had a mean position of 0.27

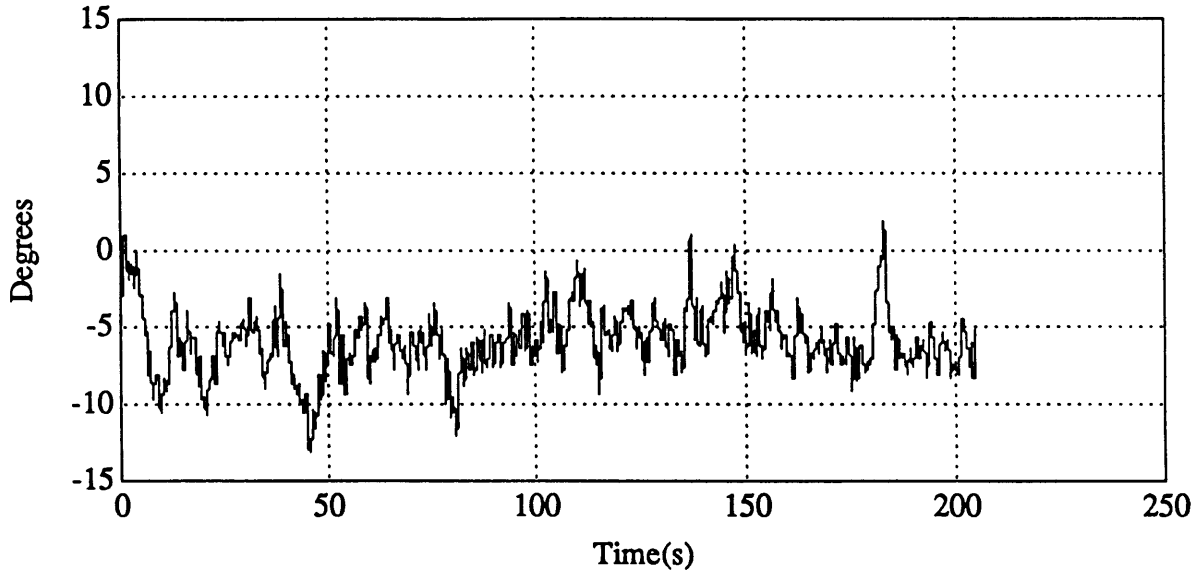


(a)

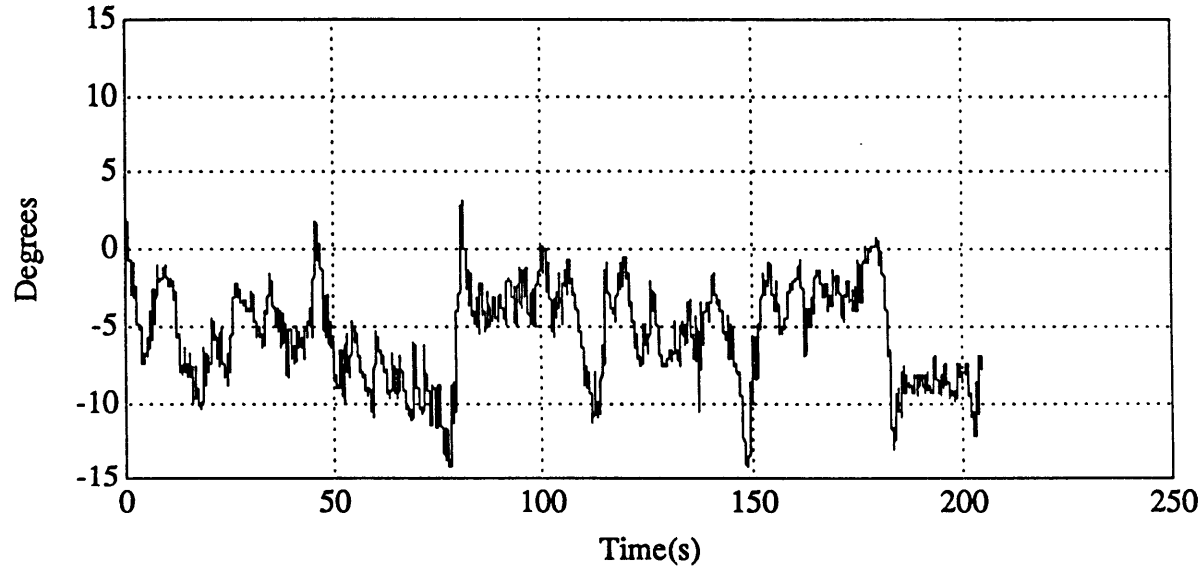


(b)

Figure 5.1: Subject F time history of trainer position (a) and control wheel response (b) for counterrotating visual field (CON).

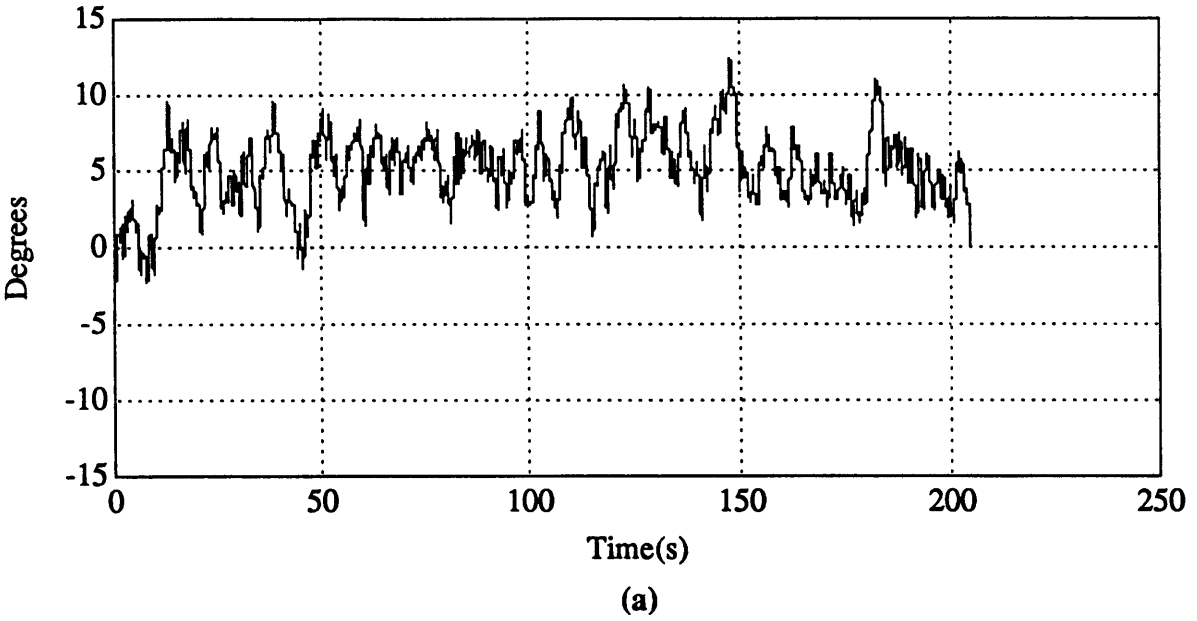


(a)

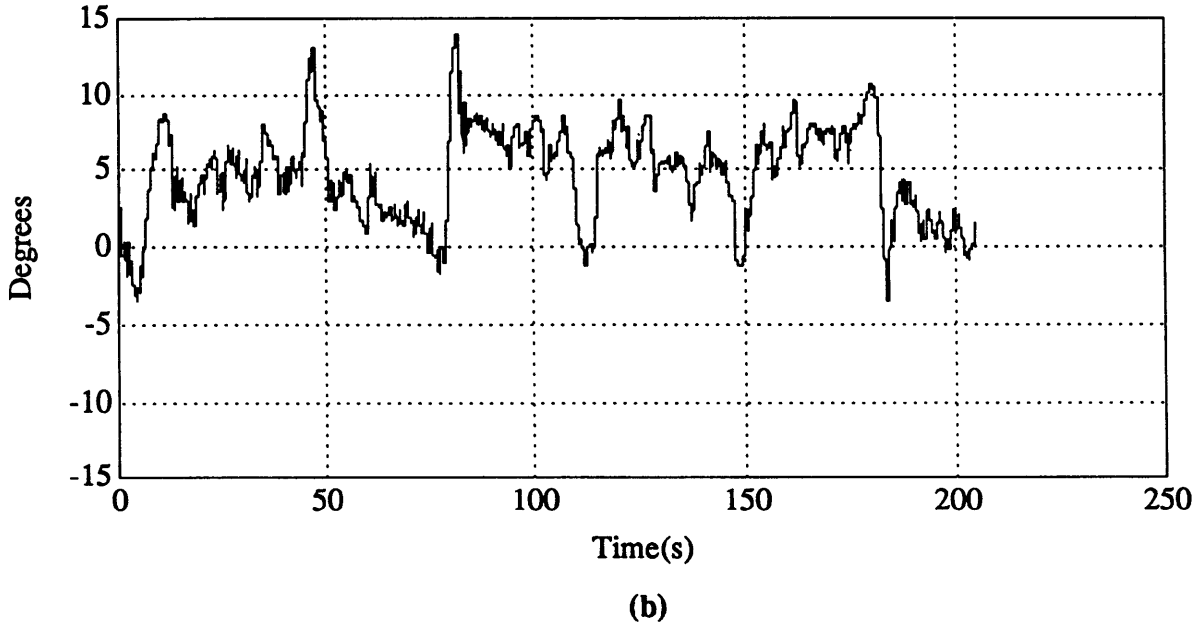


(b)

Figure 5.2: Subject F time history of trainer position (a) and control wheel response (b) for constant velocity visual field (CVL) at 10 degrees/s.

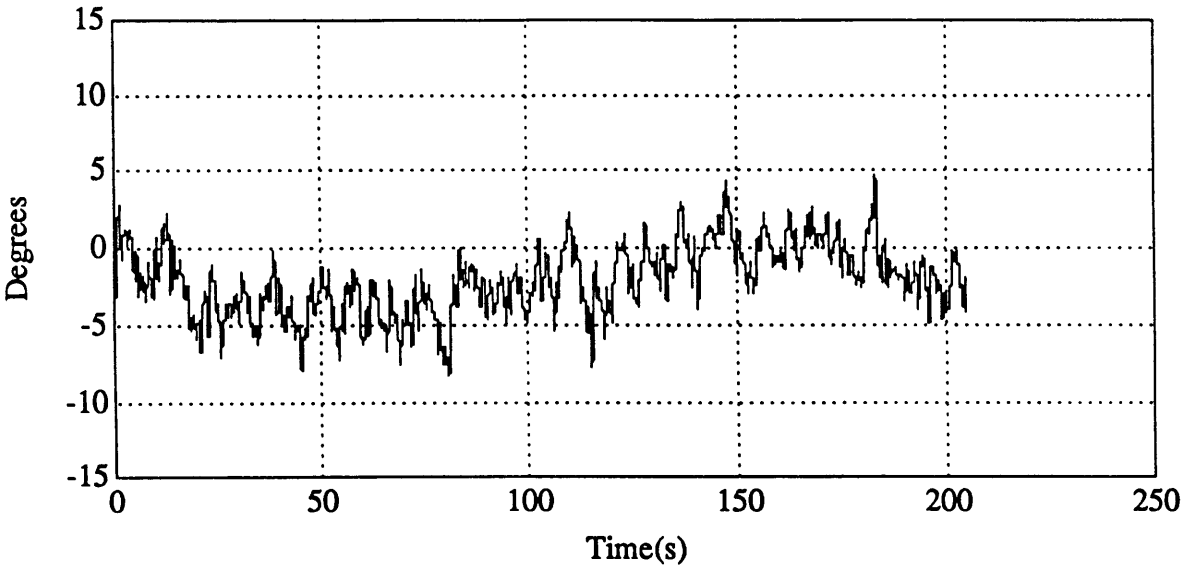


(a)

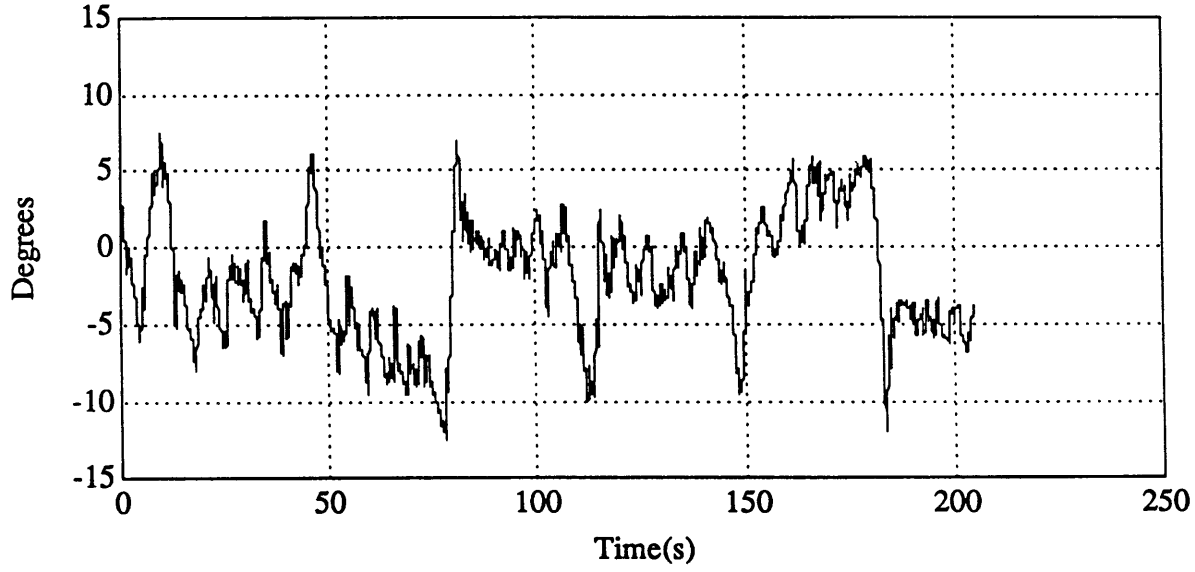


(b)

Figure 5.3: Subject F time history of trainer position (a) and control wheel response (b) for constant velocity visual field (CVR) at 10 degrees/s.



(a)



(b)

Figure 5.4: Subject F time history of trainer position (a) and control wheel response (b) for no visual field (DARK).

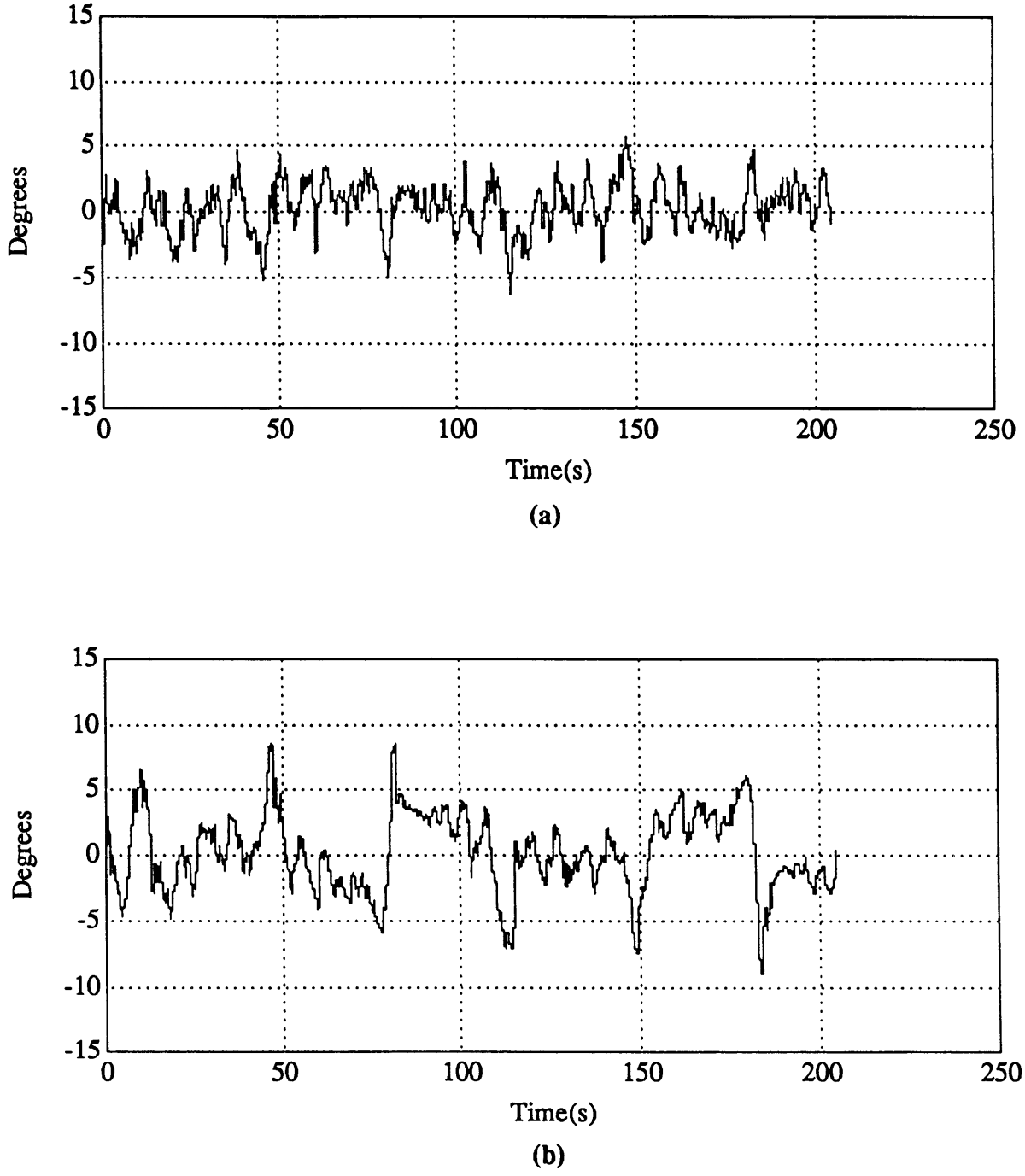


Figure 5.5: Subject F time history of trainer position (a) and control wheel response (b) for fixed visual field (FIX).

degrees. And finally, figure 5.6 illustrates subject performance with the sum of sines pseudo-random visual field (SS). With this field, the subject reached maximum roll positions of 11.95 and -5.78 degrees, and had a mean position of 1.22 degrees. The runs presented here for subject F are representative of the runs for the other five subjects who took part in the experiment.

5.1.2. Individual Subject Results

This section presents the results for individual subjects for mean and RMS trainer position. Mean and RMS position rankings are included for all visual fields except CVL and CVR which are discussed separately since the subject had only one trial with these fields.

5.1.2.1. Mean Trainer Position

The results for mean trainer positions for each subject and visual field condition are given in table 5.1, and in graphical form for easy comparison in figure 5.7. Note the large standard errors and differences between subjects for a given visual field. There are several possible reasons for these discrepancies: (1) too few trials were performed to obtain an accurate measure of the mean, (2) subjects could not estimate zero degrees accurately enough to give a consistent mean across trials, and (3) trainer dynamics might be too slow to allow subjects to maintain a consistent mean position across trials. Whatever the reason, these results show, not surprisingly, that mean trainer position is a poor measure of subject performance in the nulling task.

To investigate which of these mean positions were significantly different from the desired zero degrees (the subject was told to maintain the trainer "as erect as possible"), a χ^2 test was performed, the results of which are given in table 5.2. Subject B had negative mean positions which were statistically different from zero degrees for all visual fields which could be caused by a rightward bias in estimating tilt. The remaining subjects had mean positions for FIX and SS which were not significantly different from zero. Subject D and E had mean positions statistically different from zero degrees for CON and DARK,

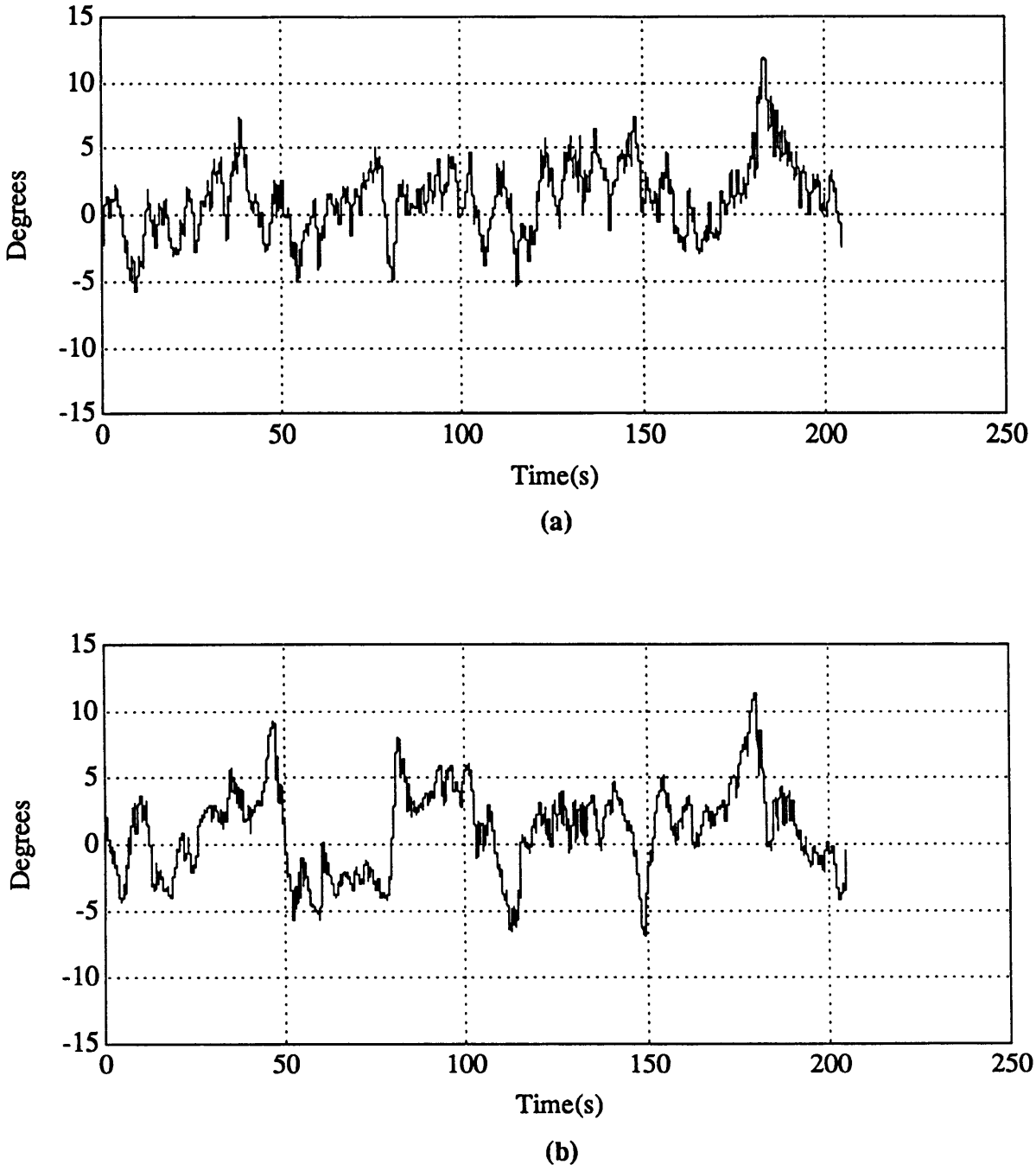


Figure 5.6: Subject F time history of trainer position (a) and control wheel response (b) for sum of sines visual field (SS).

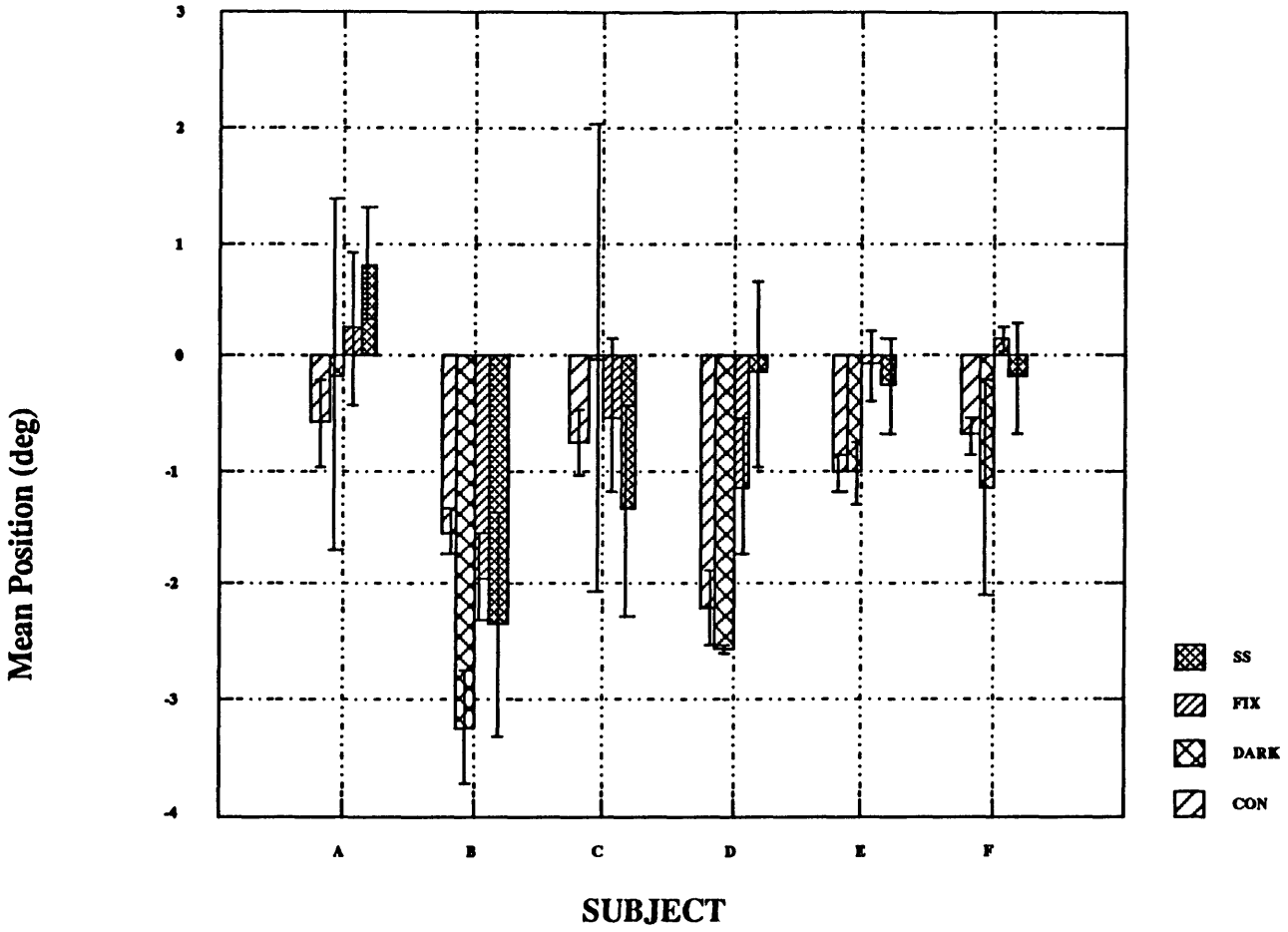


Figure 5.7: Individual subject mean trainer position (± se) for each of the visual field conditions. Visual field plot order is CON, DARK, FIX, SS.

as did subjects C and F for CON. That 21 of 24 mean positions showed a small leftward bias is most likely a measurement artifact. This could be caused by an (1) asymmetric potentiometer signal, an (2) A/D bias during data acquisition, a (3) natural bias in the subject population, or a (4) potentiometer reading of 0 volts actually corresponding to a small rightward tilt (e. g. 0 volts corresponding to a small rightward roll). The first and second reasons can be discarded, since the trainer position in response to the disturbance (without control wheel input) was symmetric with a mean of zero degrees (see figure 4.1 in chapter 4). Therefore, a combination of (3) and (4) is most likely the cause. First off, it is not inconceivable that the six subjects tested had a rightward directional bias in their ability to estimate zero degrees (i. e. to the subjects, a slight leftward roll was "vertical"). Second, although the trainer was set to vertical using a combination of levels and inclinometers, this gave at best 1 degree accuracy in roll position. In addition, although the subject's head was restrained using the chin and forehead straps, this in no way guaranteed that the subject sat level or that his head remained firmly fixed with respect to the trainer. Either of these effects alone, or a combination, could easily explain the very small asymmetry present in the trainer mean position data.

	CON	DARK	FIX	SS
A	-0.584 ± 0.377	-0.159 ± 1.545	0.253 ± 0.668	0.819 ± 0.499
B	-1.537 ± 0.189	-3.246 ± 0.483	-1.942 ± 0.372	-2.347 ± 0.969
C	-0.763 ± 0.286	-0.012 ± 2.063	-0.518 ± 0.689	-1.350 ± 0.935
D	-2.204 ± 0.319	-2.561 ± 0.039	-1.142 ± 0.607	-0.153 ± 0.830
E	-1.013 ± 0.168	-1.026 ± 0.266	-0.080 ± 0.311	-0.258 ± 0.406
F	-0.690 ± 0.159	-1.161 ± 0.936	0.149 ± 0.121	-0.190 ± 0.494

Table 5.1: Subject mean trainer position in degrees (± se).

	CON	DARK	FIX	SS
A	0.182	0.603	0.565	0.058
B	<0.001	<0.001	<0.001	<0.05
C	<0.05	0.607	0.457	0.079
D	<0.001	<0.001	0.103	0.219
E	<0.001	<0.001	0.587	0.182
F	<0.001	0.281	0.284	0.207

Table 5.2: Results of χ^2 test on mean trainer position.

To investigate differences in the visual field effect on mean position for each subject, we performed an ANOVA. As one might expect from the large variances in mean position, no significant trends were seen ($p > 0.19$ for all subjects). This is not surprising, since CON, DARK, FIX, and SS visual fields do not create the unidirectional roll vection necessary to induce a control wheel (and thus position) bias.

Despite the fact that no significant mean position differences were seen with CON, DARK, FIX, and SS, interesting results were seen with CVL and CVR. Although each subject had only one trial for each, it is apparent from table 5.3 that the roll vection induced by the moving visual field did indeed cause the subjects to bias the trainer in the direction of the visual field. For CVL, we see a leftward bias (towards negative angles), and for CVR we see a rightward bias (towards positive angles). Interestingly, the effect does not appear to be as strong for CVR as it was for CVL. This is most likely due to a position measurement artifact (discussed previously).

	A	B	C	D	E	F
CVL	-0.499	-3.170	-7.107	-5.379	-4.510	-5.847
CVR	1.467	1.136	4.694	2.671	1.152	5.218

Table 5.3: Subject mean trainer position in degrees for CVL and CVR.

5.1.2.2. RMS Trainer Position

The results for RMS trainer positions for each subject and visual field condition are summarized in table 5.4, and in graphical form for easy comparison in figure 5.8. To investigate which of these RMS positions were significantly different from the disturbance RMS position of 4.2 degrees, we performed a χ^2 test. Subject B had a statistically significant difference for CON and FIX ($p < 0.001$), and subjects A and E had statistically significant differences for all visual fields ($p < 0.001$). Subjects C and D had statistically significant differences for CON, DARK, and FIX ($p < 0.001$), as did subject D for SS ($p < 0.05$), however no significant difference was seen for subject C for SS. Subject F had statistically significant differences for CON, FIX, and SS ($p < 0.001$), as well as DARK ($p < 0.05$). These results indicate a significant change due to subject nulling. Only subject B ever had an RMS position larger than the disturbance (DARK and SS fields).

	CON	DARK	FIX	SS
A	2.216 ± 0.124	2.989 ± 0.107	2.742 ± 0.203	3.118 ± 0.169
B	2.651 ± 0.024	4.358 ± 0.419	3.326 ± 0.231	4.462 ± 0.453
C	1.974 ± 0.111	3.210 ± 0.173	2.365 ± 0.158	3.834 ± 0.386
D	2.960 ± 0.224	3.440 ± 0.090	2.940 ± 0.292	3.604 ± 0.283
E	2.280 ± 0.208	2.417 ± 0.112	2.200 ± 0.041	2.303 ± 0.038
F	1.715 ± 0.033	2.565 ± 0.499	1.863 ± 0.018	2.966 ± 0.048

Table 5.4: Subject RMS trainer position in degrees (± se).

Rankings for individual subject performance for the four visual field conditions are given in table 5.5. Subjects A, B, C, and F had the identical order (largest RMS first): SS, DARK, FIX, CON. Interestingly, subject D had the least RMS position with FIX rather than CON, but otherwise matched the trend. The largest RMS positions were seen with SS for all subjects except E, whose RMS position was nearly unaffected by the visual field.

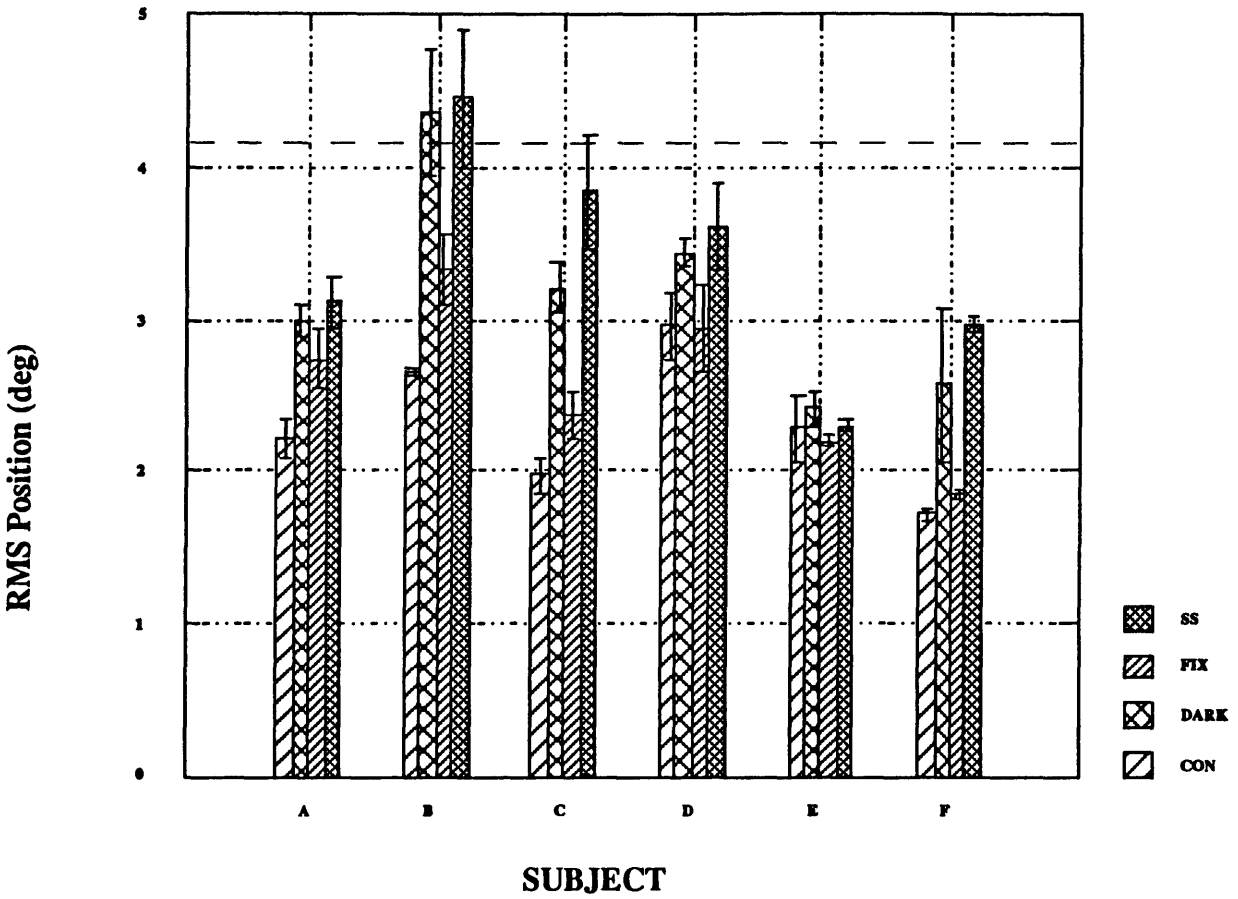


Figure 5.8: Individual subject RMS trainer position (± se) for each of the visual field conditions. Dashed line at 4.2 degrees represents the disturbance RMS.

Again, we performed an ANOVA (post-hoc Tukey test) for each subject. The differences were not statistically significant for subjects B, D, and E, but were significant for subjects A and C between SS and CON ($p < 0.05$). Subject F showed a significant difference between SS and CON and between SS and FIX ($p < 0.05$ in both cases). No significant differences were seen for the other visual field pairs.

	SS	DARK	FIX	CON
A	1	2	3	4
B	1	2	3	4
C	1	2	3	4
D	1	2	4	3
E	2	1	4	3
F	1	2	3	4

Table 5.5: Rankings of visual field effect on RMS trainer position (1=maximum, 4=minimum).

Although each subject had only one trial for CVL and CVR, it is apparent from table 5.6 that the vection induced by the moving visual field resulted in an enhanced RMS position due to the subject biasing the trainer in the direction of the visual field. Due no doubt to the position measurement artifact discussed previously, the effect seems weaker for CVR than CVL. If there were no such artifact, we would expect the effects to be more symmetric.

	A	B	C	D	E	F
CVL	2.863	5.221	7.502	6.535	5.404	6.219
CVR	3.268	3.580	5.307	3.852	2.388	5.741

Table 5.6: Subject RMS trainer positions in degrees for CVL and CVR.

5.1.3. Population Results

Apparent population trends in the subject nulling proficiency are presented in this section. CVL and CVR trials are included in the following discussion, since we have six runs of each with the pooled data.

5.1.3.1. Mean Trainer Position

Due to the large differences in variances for mean trainer position seen in section 5.1.2.1, population results will not be presented for the CON, DARK, SS, and FIX visual fields. However, the roll vection caused by the CVL and CVR caused a consistent mean trainer position bias. Assuming that the variance for all subjects is similar (this assumption is supported by the work of Huang [13]), we can pool the data to determine a subject population mean trainer position. Indeed we see that the rightward roll vection induced by CVL caused a leftward biased mean position (-4.419 ± 0.951 degrees), and the leftward roll vection induced by CVR caused a rightward biased mean position (2.723 ± 0.746 degrees). This result is not surprising, since this trend was seen for each individual subject, and thus must be preserved for the subject population. Again the measurement artifact discussed in section 5.1.2.1 gives the illusion that CVR was less provocative than CVL in inducing a position bias in the direction of the visual field.

5.1.3.2. RMS Trainer Position

Subject rankings for RMS trainer position for a given visual field condition are tabulated in table 5.7. Inspection of this table shows that subjects E and F had the smallest RMS position, subjects B and D the largest, and A and C had intermediate values. Interestingly, the three female subjects (B, C, and D) had the three largest RMS positions for SS and DARK, and the two largest for CON and FIX. At the opposite extreme was subject E who was nearly unaffected by the visual field. In an attempt to quantify these differences, we performed an ANOVA (post-hoc Tukey test) for each visual field (pooling the subject data). The results of this analysis are summarized in table 5.8. An intersection of two subjects gives the fields for which there was a significant difference

in RMS position, with *none* indicating that no significant difference was found for any visual field for that subject pair ($p > 0.05$). Although a significant difference between subjects C and D was seen for CON, no other significant differences were seen among males (subjects A, E, and F) or females (subjects B, C and D), but differences were seen between some individuals.

	A	B	C	D	E	F
CON	4	2	5	1	3	6
DARK	4	1	3	2	6	5
FIX	3	1	4	2	5	6
SS	4	1	2	3	6	5

Table 5.7: Subject rankings for RMS trainer position (1=maximum, 6=minimum).

	A	B	C	D	E
A					
B	SS				
C	*none*	*none*			
D	*none*	*none*	CON		
E	*none*	DARK, SS	SS	SS	
F	*none*	ALL	*none*	CON, FIX	*none*

Table 5.8: ANOVA results comparing subject RMS trainer positions for a given visual field.

Since interesting individual subject trends were apparent in RMS trainer position, we decided to look at population trends. Subjects were more efficient in reducing the trainer RMS position for some visual fields than for others, as can be seen by inspecting table 5.9 or figure 5.9. CVL had the largest RMS position, followed by CVR, SS,

DARK, FIX, and CON. The RMS value of the position disturbance without subject control wheel input was 4.20 degrees. The differences among the visual field tests (with subjects pooled) were explored by an ANOVA (post-hoc Tukey test), the results of which are given in table 5.10. CVL was statistically different from CON, FIX, DARK, and SS ($p < 0.001$), as well as CVR ($p < 0.05$). CVR was statistically different from CON ($p < 0.01$) and FIX ($p < 0.05$), but not statistically different from SS or DARK. And finally, SS was statistically different from CON ($p < 0.01$). No significant differences were seen for any other visual field pairs.

CON	CVL	CVR	DARK	FIX	SS
2.299 ± 0.131	5.624 ± 0.647	4.023 ± 0.519	3.163 ± 0.211	2.573 ± 0.157	3.381 ± 0.175

Table 5.9: Population RMS trainer position in degrees (± se).

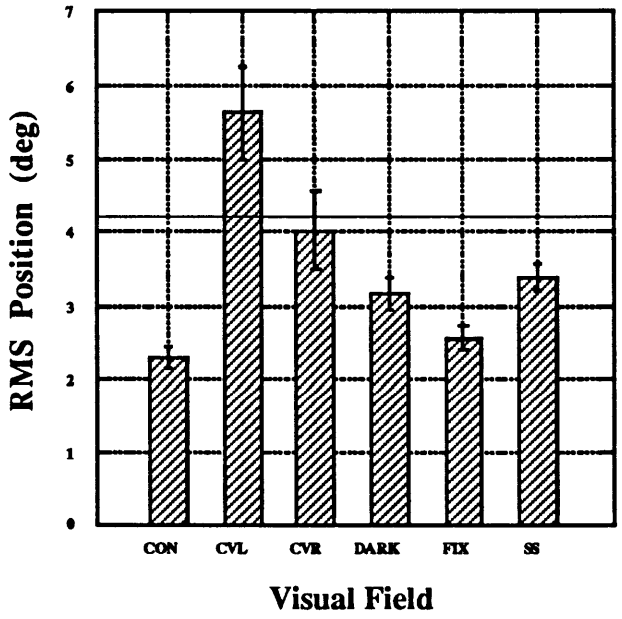


Figure 5.9: Population RMS trainer position (± se) for each of the visual field conditions. Solid line at 4.2 degrees represents disturbance RMS position.

To investigate which of these RMS positions were significantly different from the disturbance RMS position of 4.2 degrees, we performed a χ^2 test. The pooled RMS

position data for CON, DARK, FIX, and SS were statistically different from 4.2 degrees ($p < 0.001$), but the RMS positions for CVL and CVR were not.

5.2. Frequency Domain Analysis of Subject Response

By use of the Fast Fourier Transform (FFT), we can investigate subject performance in the manual roll stabilization task using the subject's frequency response characteristics. In the following sections, we present the data using amplitude spectra, scalar performance measures (SPM), visual response measures (VRM), and describing function analysis (see chapter 4 for a derivation of these measures).

	CON	CVL	CVR	DARK	FIX	SS
CON	1.000					
CVL	<0.001	1.000				
CVR	<0.01	<0.05	1.000			
DARK	0.152	<0.001	0.356	1.000		
FIX	0.970	<0.001	<0.05	0.550	1.000	
SS	<0.01	<0.001	0.581	0.979	0.099	1.000

Table 5.10: ANOVA results for RMS trainer position.

5.2.1. Representative Frequency Domain Results

Figure 5.10 shows a set of mean position amplitude spectra for subject F, illustrating the improvement due to subject nulling for each of the visual fields. In each plot (a-f) in the figure, the 12 position amplitudes of the sum of sines vestibular disturbance are shown by the +'s (connected by a solid line), and the x's show the amplitudes of the trainer position with subject nulling (connected by a dashed line) at these same frequencies. Therefore, the improvement due to subject nulling is simply the area between these two curves, which is measured by the SPM. Finally, the *'s are the subject remnant (the mean of which is shown by a solid horizontal line), and the o's show the amplitudes with subject nulling at the visual disturbance frequencies (which is part of the remnant except in SS

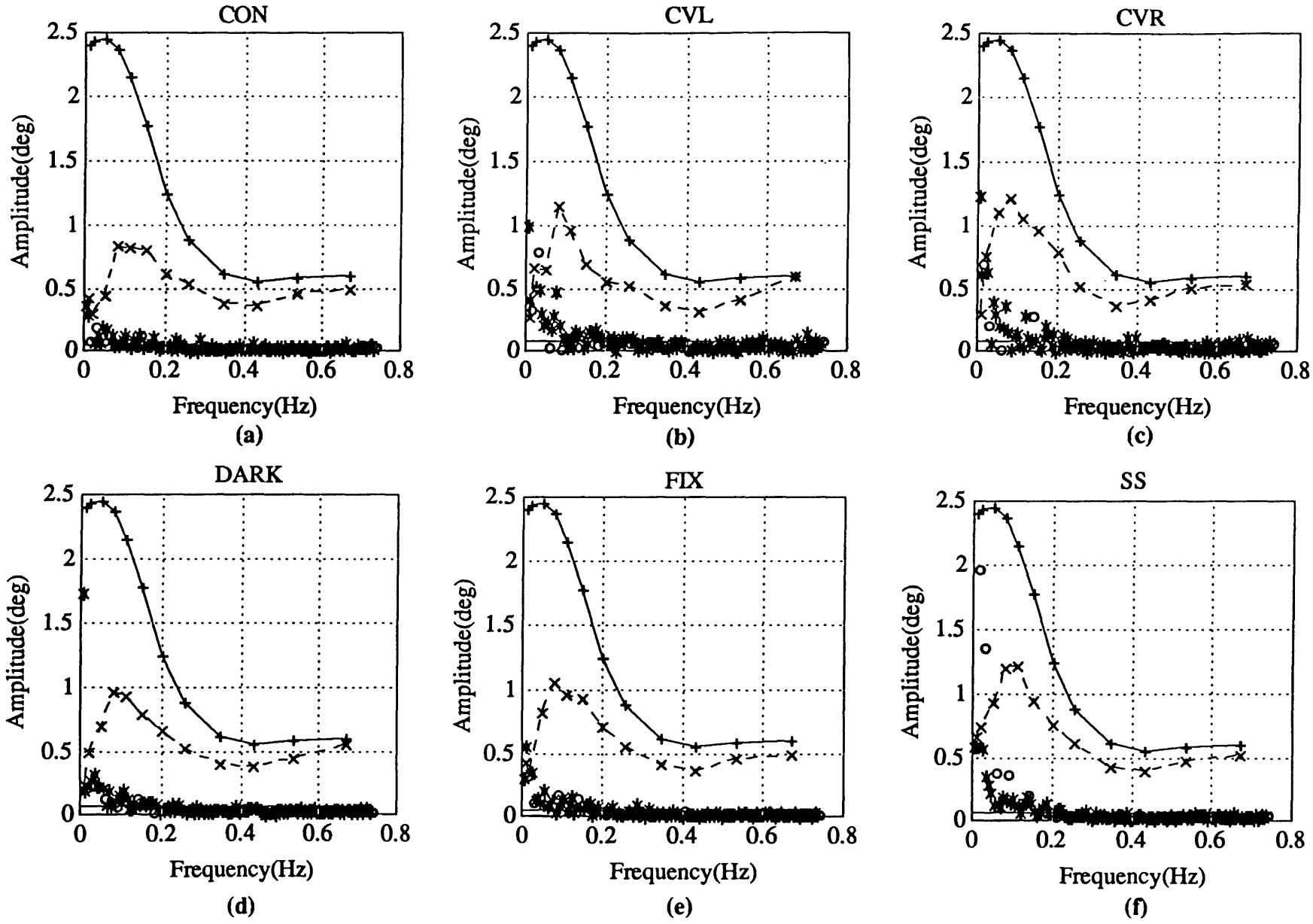


Figure 5.10: Subject F mean position amplitude spectra. +'s are amplitudes if subject did no nulling, x's are at the vestibular frequencies, o's are at the visual frequencies, and *'s are mean subject remnant.

trials). Close inspection of the figure reveals that nulling was best at low frequencies (an amplitude of zero degrees would be perfect), and tended to worsen with increasing frequency, with the subject nulling little of the trainer position at frequencies above approximately 0.4 Hz. Performance was best with CON, worst with CVL, CVR, and SS, and intermediate with DARK and FIX (whose performances were similar). Note the enhanced response at the visual disturbance frequencies with SS (o's are well above the mean remnant). These general trends were typical of all the subjects who took part in the experiment.

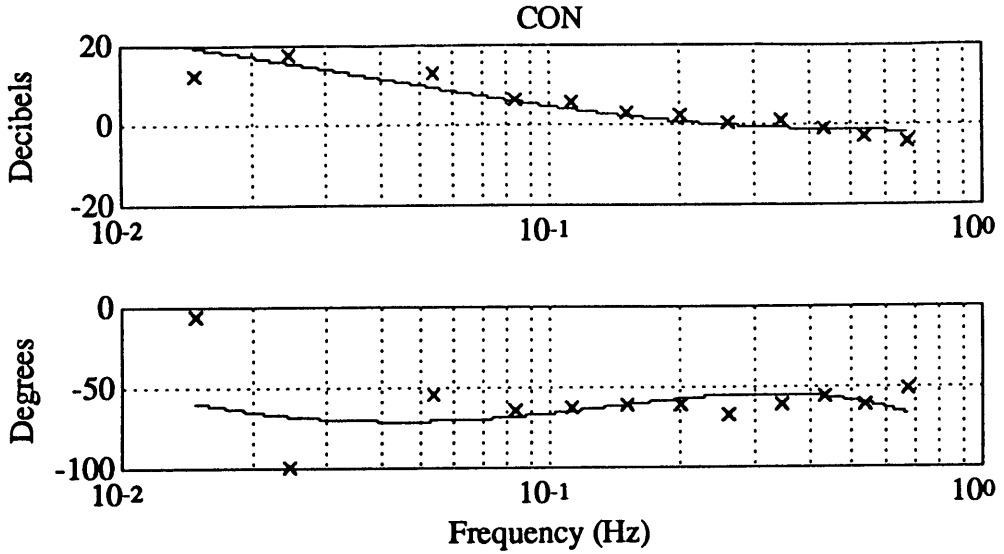
Figure 5.11 shows a set of mean operator describing functions and associated fits for subject F for each of the visual fields used in the experiment. The high gain at low frequencies decays with increasing frequency and appears to level off at high frequencies; phase remains nearly constant. CON had the largest low frequency gain (due to the confirming visual cues), CVL, CVR, and SS had the least (due to the presence of the disorienting visual field), and DARK and FIX were in between (dependence was entirely on vestibular cues). These general trends were typical of all of the subjects who took part in the experiment.

5.2.2. Individual Subject Results

This section presents the results for individual subjects for SPM, VRM, and operator describing functions. SPM and VRM rankings for CVL and CVR are not included, since each subject had only one trial with these visual fields; instead, they are discussed separately.

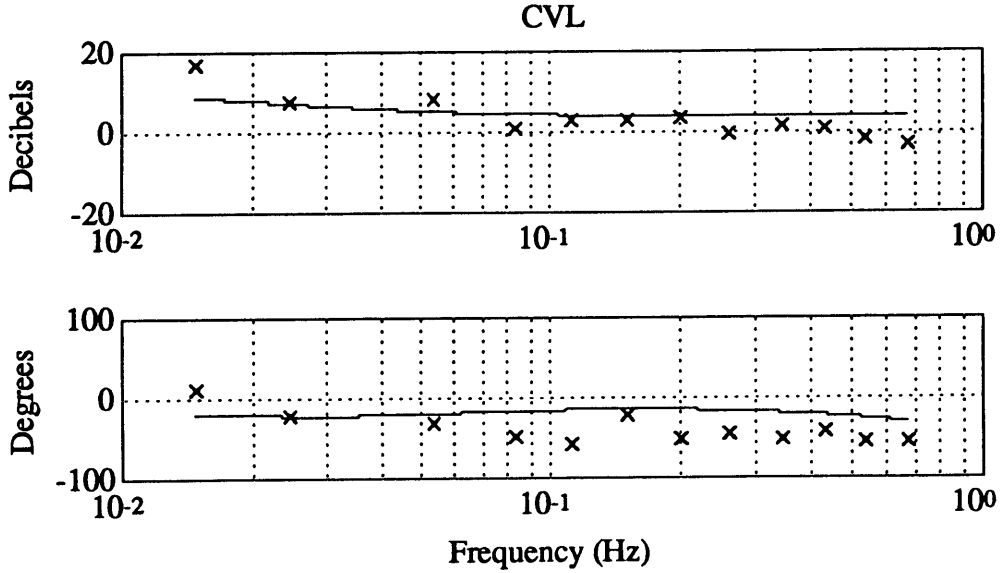
5.2.2.1. SPM

The SPM results for each individual subject and visual field condition are summarized in table 5.11, and in graphical form for easy comparison in figure 5.12. Rankings by individual subject performance for the four visual field conditions are given in table 5.12. All subjects had the largest SPM (and thus best nulling proficiency) with the CON visual field, and 5 subjects had their worst performance with SS (subjects D's worst



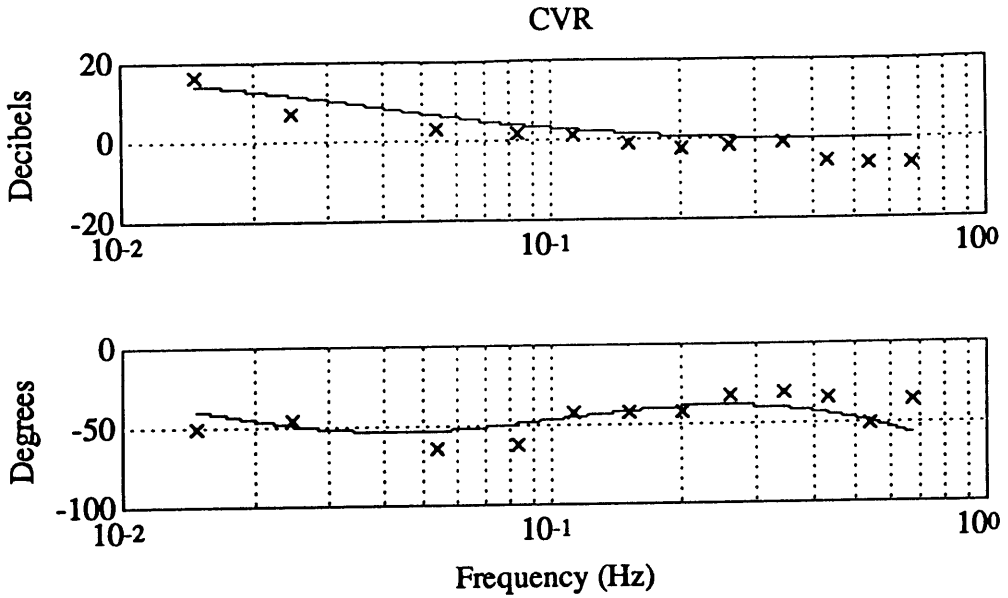
(a)

$$\phi(s) \rightarrow \frac{20.0(0.825s+1)}{(20.645s+1)} e^{-0.209s} \rightarrow -\lambda(s)$$

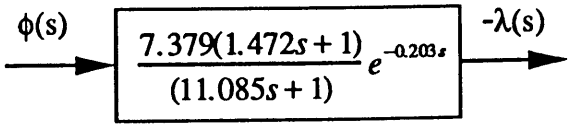


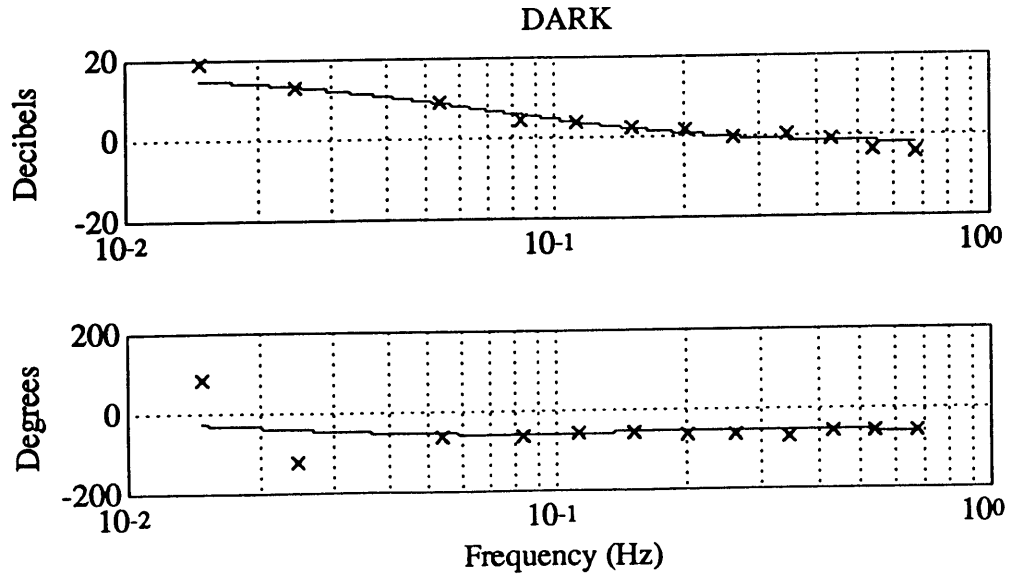
(b)

$$\phi(s) \rightarrow \frac{3.249(4.129s+1)}{(8.440s+1)} e^{-0.116s} \rightarrow -\lambda(s)$$

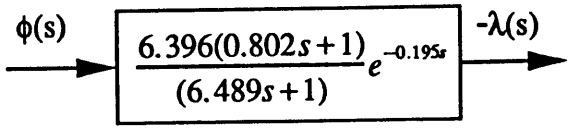

CVR
Decibels
20
0
-20
10-2
10-1
100
Degrees
0
-50
-100
10-2
10-1
100
Frequency (Hz)

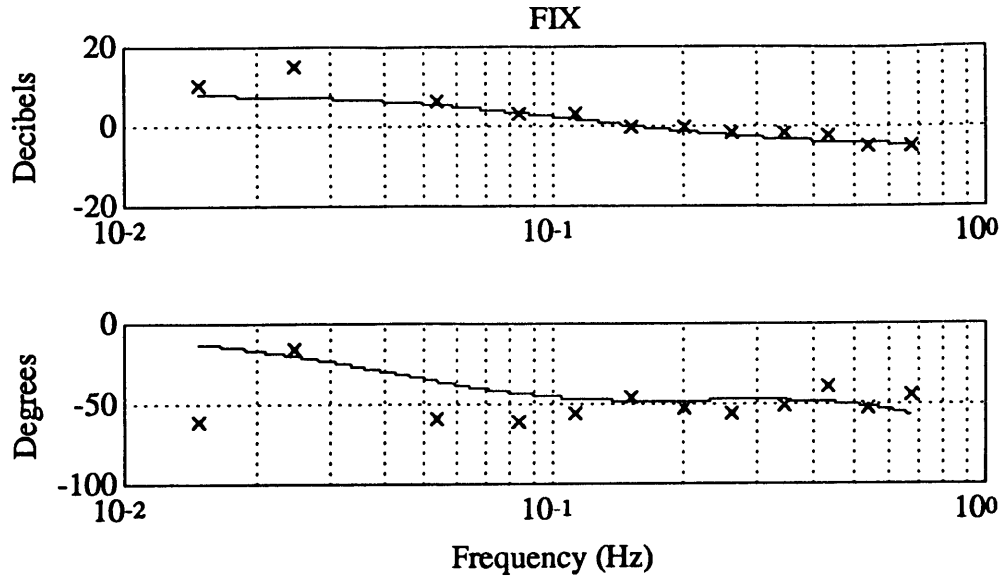
(c)


ϕ(s)
$\frac{7.379(1.472s+1)}{(11.085s+1)}e^{-0.203s}$
-λ(s)

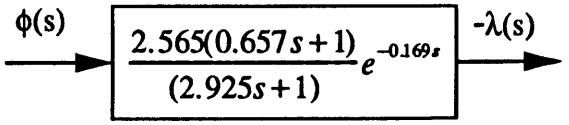

DARK
Decibels
20
0
-20
10-2
10-1
100
Degrees
200
0
-200
10-2
10-1
100
Frequency (Hz)

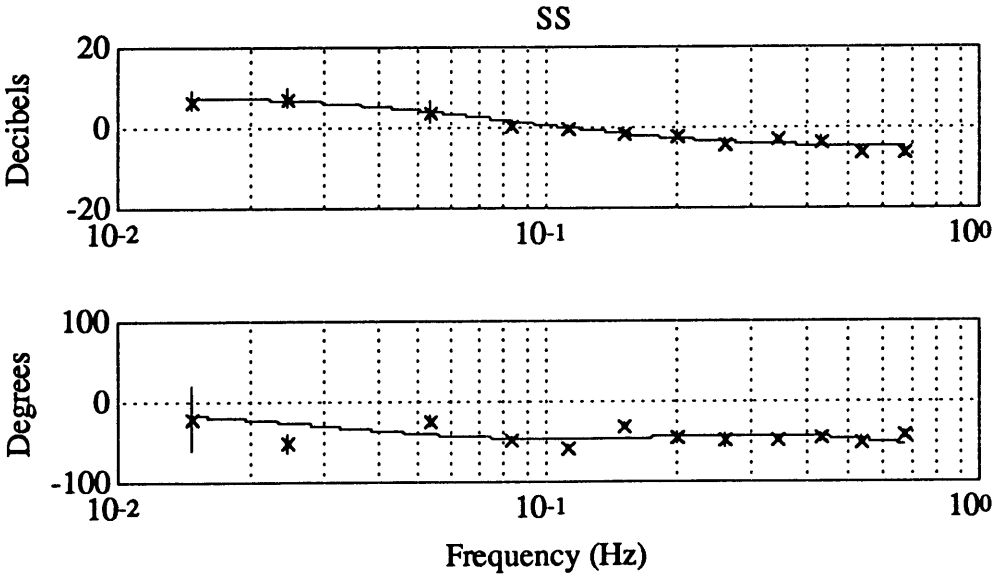
(d)


ϕ(s)
$\frac{6.396(0.802s+1)}{(6.489s+1)}e^{-0.195s}$
-λ(s)



(e)





(f)

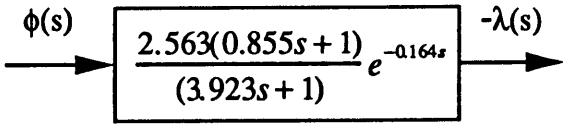


Figure 5.11: Subject F mean operator describing function for each visual field condition. Boxes contain transfer function fit shown by solid line. Error bars for figure (f) are sd.

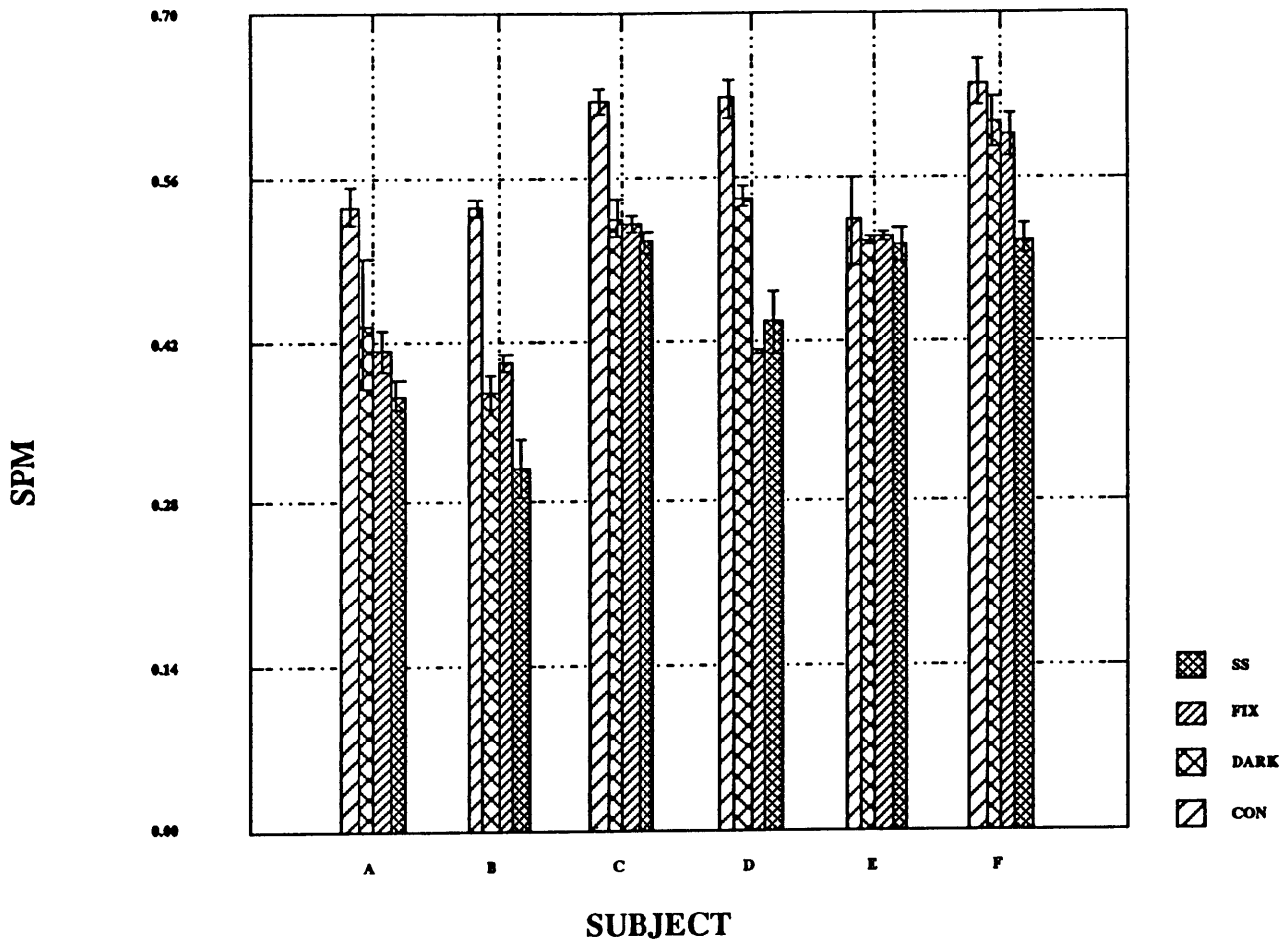


Figure 5.12: Individual subject SPM(± se) for each of the visual field conditions.

performance was with FIX). The latter fact shows that subject nulling proficiency was indeed influenced by the visual field. Four subjects had their second best performance with DARK and two had it with FIX. Three subjects had their third best performance with FIX, and two had it with DARK. Again, we performed an ANOVA (post-hoc Tukey test) on the SPM for each subject. The differences were not statistically significant for subject E. However, the remaining subjects had a significant difference between SS and CON ($p < 0.01$ for all subjects except A with $p < 0.05$). Subject B had significant differences between DARK and CON and FIX and CON ($p < 0.05$), as did subject C ($p < 0.01$). Subject D had a significant difference between FIX and CON, and subject F had significant differences between SS and DARK and SS and FIX ($p < 0.05$). No significant differences were seen for the other visual field pairs.

	CON	DARK	FIX	SS
A	0.537 ± 0.015	0.434 ± 0.056	0.411 ± 0.017	0.373 ± 0.012
B	0.535 ± 0.008	0.376 ± 0.015	0.401 ± 0.008	0.311 ± 0.025
C	0.625 ± 0.012	0.525 ± 0.016	0.521 ± 0.008	0.504 ± 0.010
D	0.628 ± 0.017	0.543 ± 0.010	0.410 ± 0.002	0.436 ± 0.028
E	0.523 ± 0.037	0.505 ± 0.004	0.510 ± 0.004	0.502 ± 0.014
F	0.641 ± 0.021	0.608 ± 0.020	0.596 ± 0.017	0.506 ± 0.012

Table 5.11: Subject mean SPM (± se) for each visual field.

The SPMs for CVL and CVR are tabulated in table 5.13. Most interesting is that the SPMs were still fairly large despite the roll vection induced by the visual field. While the roll vection induced a dc bias in trainer position, the effect was not dominant at the frequencies used in the SPM measure.

	CON	DARK	FIX	SS
A	1	2	3	4
B	1	3	2	4
C	1	2	3	4
D	1	2	4	3
E	1	3	2	4
F	1	2	3	4

Table 5.12: Rankings for visual field effect on SPM (1=maximum, 4=minimum).

	A	B	C	D	E	F
CVL	0.386	0.245	0.598	0.342	0.452	0.604
CVR	0.330	0.278	0.488	0.444	0.525	0.527

Table 5.13: Subject SPM for CVL and CVR.

5.2.2.2. VRM

The VRM results for each individual subject and visual field condition are summarized in table 5.14, and in graphical form for easy comparison in figure 5.13. VRM rankings for the four visual field conditions are given in table 5.15. Ignoring subject E (who was relatively unaffected by the visual field), the remaining subjects had their largest VRM (and thus largest response at the visual frequencies) with the SS visual field. This shows that five of the subjects were indeed influenced by the visual field.

For the remaining three visual field conditions (CON, DARK, and FIX), the VRM simply gives a measure of the subject remnant at the visual stimulus frequencies, since there was no dynamic visual stimulus. For these three visual conditions, four subjects had their largest VRM with DARK, and one subject had it with FIX. Five subjects had their second largest VRM with FIX, and one had it with DARK. All six subjects had their

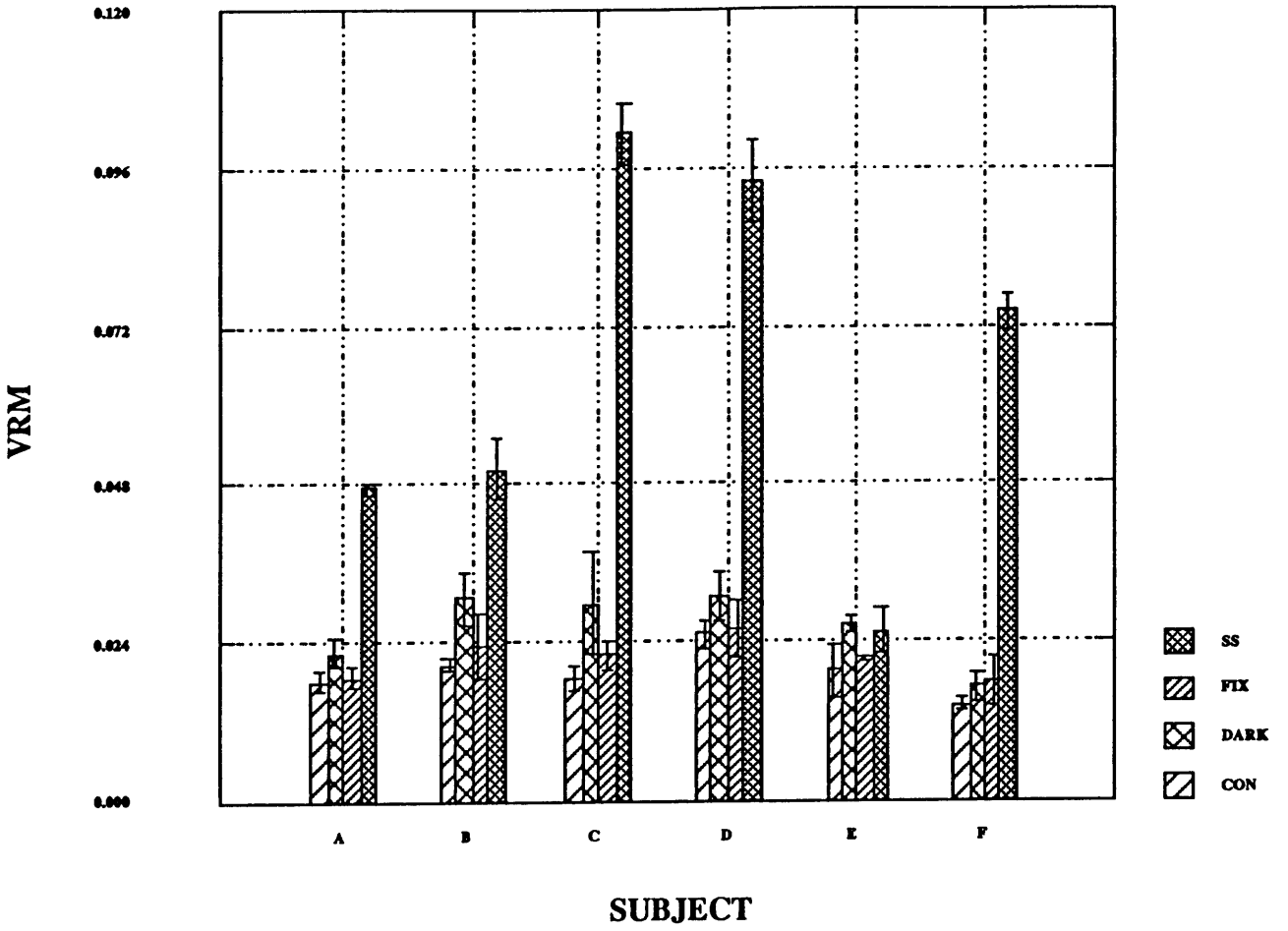


Figure 5.13: Individual subject VRM(± se) for each of the visual field conditions.

smallest VRM with CON. Again to investigate the statistical significance of these differences, we performed an ANOVA (post-hoc Tukey test) for each subject. The differences were not statistically significant for subject E. Subjects A, C, and F had statistically significant differences between SS and CON, SS and DARK, and SS and FIX ($p < 0.001$). Subject B had statistically significant differences between SS and CON and SS and FIX ($p < 0.05$). Finally, subject D had statistically significant differences between SS and CON and SS and FIX ($p < 0.001$), as well as between SS and DARK ($p < 0.01$).

	CON	DARK	FIX	SS
A	0.018 ± 0.002	0.022 ± 0.002	0.018 ± 0.002	0.047 ± 0.001
B	0.021 ± 0.001	0.031 ± 0.004	0.023 ± 0.005	0.050 ± 0.005
C	0.018 ± 0.002	0.030 ± 0.008	0.022 ± 0.002	0.101 ± 0.005
D	0.025 ± 0.002	0.031 ± 0.004	0026 ± 0.004	0.094 ± 0.006
E	0.019 ± 0.004	0.026 ± 0.001	0.021 ± 0.000	0.025 ± 0.003
F	0.014 ± 0.000	0.017 ± 0.002	0.018 ± 0.004	0.074 ± 0.003

Table 5.14: Subject mean VRM (± se) for each visual field.

	SS	DARK	FIX	CON
A	1	2	3	4
B	1	2	3	4
C	1	2	3	4
D	1	2	3	4
E	2	1	3	4
F	1	3	2	4

Table 5.15: Rankings for visual field effect on VRM (1=maximum, 4=minimum).

Although each subject had only one trial for CVL and CVR, it is apparent from table 5.16 that the vection induced by the moving visual field resulted in an enhanced VRM (i.e. enhanced subject remnant at the visual disturbance frequencies) for subjects B, C, D, and E. Subjects A and F had VRMs near those seen for CON, DARK, and FIX.

	A	B	C	D	E	F
CVL	0.016	0.040	0.047	0.053	0.052	0.026
CVR	0.022	0.043	0.023	0.033	0.016	0.029

Table 5.16: Subject VRM for CVL and CVR.

5.2.2.3. Operator Describing Functions

Transfer function fits were made to each of the individual operator describing functions using the method developed in chapter four. Table 5.17 specifies the constraints on the fit parameters and the initial value. In the interest of space, plots of these fits will not be shown for each subject. Instead the transfer function parameters are tabulated in table 5.18. In general, good fits were obtained for all visual fields except CVL and CVR, with only a few trials reaching a parameter constraint. Since there was only one run for the CV visual field conditions, frequency-by-frequency weights could not be computed. This resulting lack of flexibility (particularly at low frequencies) made the fits much more difficult for these visual fields. Notice the large standard errors on the mean fit parameters. This is primarily due to the large low frequency error bars in the gain and phase which resulted from depriving the subject of low frequency visual cues. At high frequencies, dependence was primarily on vestibular cues, so less variation in gain and phase was seen. Therefore, the high frequencies were weighed more heavily in the fitting process, which explains the large variation in K, τ_1, and τ_d (since the value for τ_2 depends upon these, it had large variation as well). This hypothesis is supported by the fact that CON (which had good low frequency performance due to confirming visual cues) had the smallest parameter standard errors.

Cross-subject comparisons show that there were large differences in parameter means for all visual fields. Even in the case of CON (where the fit quality was the best, as discussed above), mean low frequency gain (K) varied from 2.3 to 20.0, low frequency time constant from 0.25 to 1.5 seconds, high frequency time constant from 1.6 to 20.6, and delay time from 0.125 to 0.280 seconds. Similar variations were seen for the remaining visual fields. The variability of the results, both within and between subjects, precludes pooling the data. Therefore, to obtain more accurate fits, we decided to pool the operator describing function data in the frequency domain and perform a single fit to that mean data. The results of this effort are presented in section 5.2.3.3.

	K	τ_1	τ_2	τ_d
initial value	5.00	1.00	1.00	0.01
maximum	20.00	100.00	100.00	1.00
minimum	0.05	0.01	0.01	0.00

Table 5.17: Transfer function fit parameter constraints and initial values.

5.2.3. Population Results

Apparent population trends in the subject frequency response are presented in this section using amplitude spectra, SPM, VRM, operator describing functions, and open-loop transfer functions. An ANOVA was performed on the pooled SPM and VRM data to determine which visual field(s) had statistically significant effects on the subject's control strategy during the manual roll stabilization task; the results of this analysis are presented in section 5.2.3.2.

5.2.3.1. Amplitude Spectra

Figure 5.14 shows the population mean amplitude spectra of the trainer position (with subject compensation) for each of the visual field conditions used in the experiment. Inspection of this figure shows that, for all visual fields, subjects were able to reduce trainer position at the frequencies in the vestibular disturbance. The most efficient nulling

Subject	CON				DARK			
	K	τ_1	τ_2	τ_d	K	τ_1	τ_2	τ_d
A	2.269 ± 0.025	1.520 ± 0.017	3.553 ± 0.019	0.279 ± 0.006	2.749 ± 0.287	1.657 ± 0.085	7.981 ± 0.228	0.240 ± 0.007
B	3.541 ± 0.026	0.252 ± 0.004	2.175 ± 0.035	0.204 ± 0.038	11.36 ± 8.64 [1]	1.287 ± 0.805	52.15 ± 47.84	0.267 ± 0.053
C	14.356 ± 0.145	0.505 ± 0.027	9.607 ± 0.121	0.125 ± 0.004	6.536 ± 0.125	0.729 ± 0.080	7.278 ± 0.566	0.187 ± 0.023
D	3.993 ± 0.383	0.347 ± 0.014	1.604 ± 0.204	0.159 ± 0.007	2.443 ± 0.084	1.005 ± 0.066	3.205 ± 0.285	0.201 ± 0.004
E	4.745 ± 0.695	0.469 ± 0.050	5.395 ± 1.323	0.181 ± 0.010	14.481 ± 2.403	0.685 ± 0.005	18.660 ± 2.969	0.256 ± 0.004
F	20.00 ± 0.00 [2]	0.822 ± 0.023	20.592 ± 0.414	0.208 ± 0.010	7.833 ± 1.162	0.778 ± 0.064	7.730 ± 1.905	0.181 ± 0.031

Table 5.18a: Operator describing function fit parameters (± se) for individual subjects for CON and DARK. Superscript indicates number of parameters included in calculation of mean that were at a constraint.

Subject	FIX				SS			
	K	τ_1	τ_2	τ_d	K	τ_1	τ_2	τ_d
A	6.490 ± 3.998	3.035 ± 0.503	37.33 ± 26.07	0.213 ± 0.001	6.914 ± 2.250	3.153 ± 0.716	62.9 ± 21.6 [2]	0.289 ± 0.044
B	9.224 ± 0.003	1.549 ± 0.019	29.317 ± 0.328	0.135 ± 0.02	2.516 ± 0.264	0.581 ± 0.047	5.794 ± 0.583	0.198 ± 0.015
C	20.00 ± 0.00 [2]	0.954 ± 0.033	31.402 ± 0.275	0.258 ± 0.008	3.286 ± 0.484	0.300 ± 0.065	2.388 ± 0.453	0.199 ± 0.020
D	3.470 ± 0.367	4.279 ± 0.380	22.657 ± 4.378	0.157 ± 0.004	3.263 ± 1.472	2.846 ± 1.109	14.261 ± 6.939	0.272 ± 0.030
E	9.901 ± 1.178	0.755 ± 0.001	15.577 ± 1.922	0.157 ± 0.008	8.551 ± 3.179	0.556 ± 0.026	10.603 ± 3.967	0.145 ± 0.007
F	2.562 ± 0.055	0.656 ± 0.000	2.919 ± 0.059	0.169 ± 0.003	2.641 ± 0.035	0.870 ± 0.012	4.106 ± 0.225	0.165 ± 0.009

Table 5.18b: Operator describing function fit parameters (± se) for individual subjects for FIX and SS. Superscript indicates number of parameters included in calculation of mean that were at a constraint.

Subject	CVL				CVR			
	K	τ_1	τ_2	τ_d	K	τ_1	τ_2	τ_d
A	2.038	1.508	5.950	0.302	1.668	1.078	4.187	0.244
B	0.500	74.105	100.000 *	0.726	20.000 *	1.020	100.000 *	0.221
C	0.569	100.000 *	35.662	0.367	0.569	100.000 *	35.662	0.367
D	0.845	0.010	1.001	0.072	0.500	57.385	30.869	0.247
E	0.500	69.005	38.582	0.341	20.000 *	0.855	48.502	0.227
F	3.281	0.659	2.442	0.161	7.068	1.219	13.185	0.127

Table 5.18c: Operator describing function fit parameters for individual subjects for CVL and CVR visual fields. Superscript * indicates parameter was at a constraint.

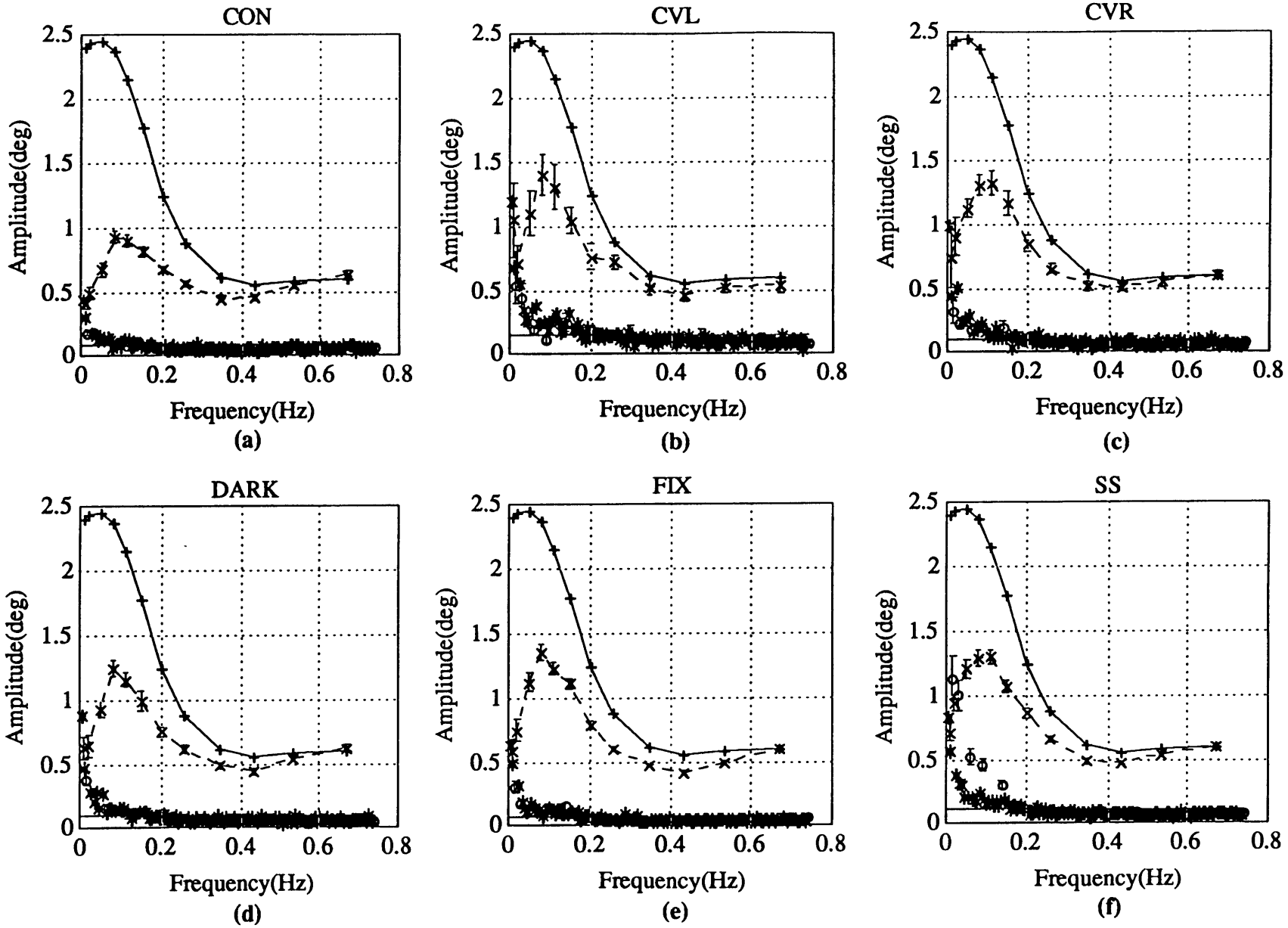


Figure 5.14: Population mean position (± se) amplitude spectra. +'s are amplitudes if subject did no nulling, x's are at the vestibular frequencies, o's are at the visual frequencies, and *'s are mean subject remnant.

was seen at low frequencies, while at high frequencies only small reductions in the roll motion were achieved.

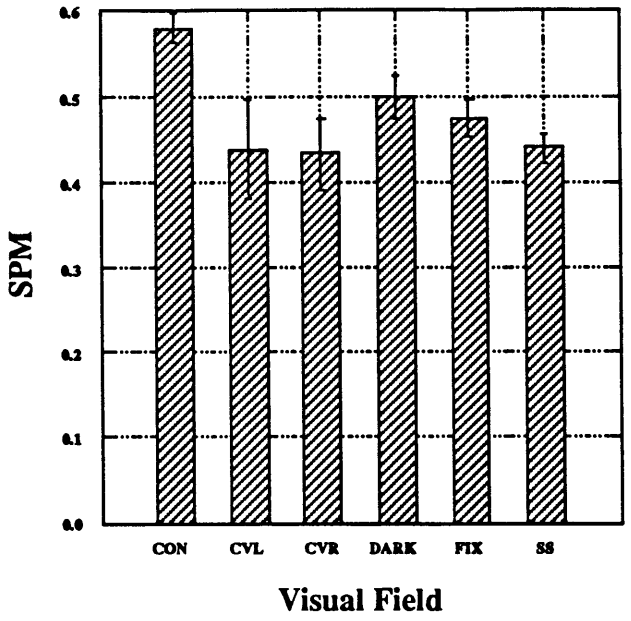
5.2.3.2. SPM

Subject rankings for SPM for a given visual field condition are tabulated in table 5.19. Inspection of this table shows that subject F had the largest SPMs (and thus best nulling proficiency) for all visual fields, and subject B had the smallest SPMs for all visual fields except CON, where she had the second smallest. Subjects C was fairly consistent with either the second or third largest SPM, as was subject A with either the fourth or fifth. Subjects D and E were the least consistent of the subjects. To investigate whether any of these differences were statistically significant, we performed an ANOVA (post-hoc Tukey test) for each subject. The results of this analysis are summarized in table 5.20. An intersection of two subjects gives the fields for which there was a significant difference in SPM, with *none* indicating that no significant difference was found for any visual field for that subject pair ($p > 0.05$). In summary, subject B was statistically different from subjects C, D, and E for DARK, from subjects C, E, and F for FIX, and from all other subjects for SS. Subjects A and D showed no significant differences for any of the visual fields.

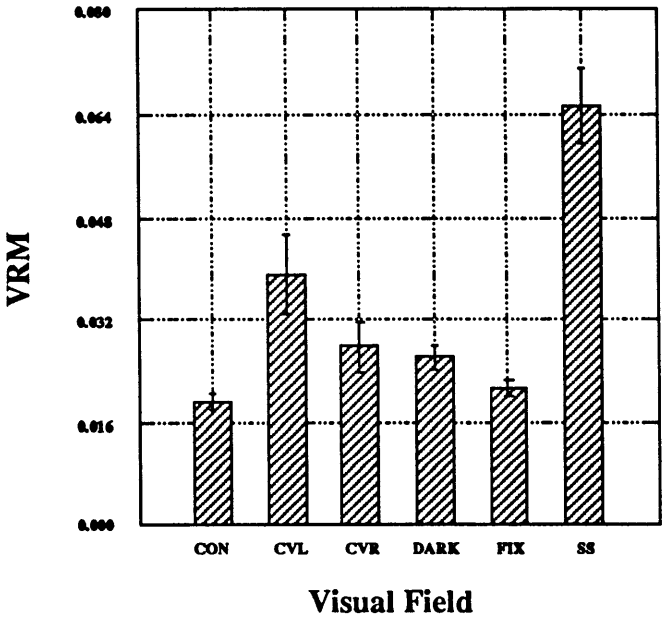
	A	B	C	D	E	F
CON	4	5	3	2	6	1
DARK	5	6	3	2	4	1
FIX	4	6	2	5	3	1
SS	5	6	2	4	3	1

Table 5.19: Subject rankings for SPM (1=maximum, 6=minimum).

Since interesting SPM trends were seen with the individual subjects, we decided to look at population trends. Subject performance, as measured by the SPM, was affected by the visual fields. These results are shown in table 5.21, and in graphical form for easy comparison in figure 5.15a. CON had the largest SPM, followed by DARK, FIX, SS,



(a)



(b)

Figure 5.15: Population mean (+/- se) SPM (a) and VRM (b).

5.22. In summary, CON was statistically different ($p < 0.05$) from all the other visual field conditions except DARK. No other significant differences were seen between other pairs of visual fields.

	A	B	C	D	E
A					
B	SS				
C	FIX, SS	DARK, FIX, SS			
D	*none*	DARK, SS	FIX		
E	FIX, SS	FIX, SS	CON	FIX	
F	DARK, FIX, SS	DARK, FIX, SS	FIX	FIX	FIX

Table 5.20: ANOVA results comparing subject SPM for each visual field.

CON	CVL	CVR	DARK	FIX	SS
0.581 ± 0.016	0.438 ± 0.059	0.432 ± 0.043	0.498 ± 0.024	0.475 ± 0.022	0.439 ± 0.017

Table 5.21: Population SPM (± se).

	CON	CVL	CVR	DARK	FIX	SS
CON	1.000					
CVL	<0.05	1.000				
CVR	<0.05	1.000	1.000			
DARK	0.190	0.726	0.643	1.000		
FIX	<0.05	0.955	0.919	0.985	1.000	
SS	<0.001	1.000	1.000	0.381	0.843	1.000

Table 5.22: Population ANOVA results for SPM.

5.2.3.3. VRM

Subject rankings for VRM for a given visual field condition are tabulated in table 5.23. Inspection of this table shows that subject F had the smallest VRM for CON, DARK, and FIX, while subject D and B had the largest. To investigate whether these differences were statistically significant, we performed an ANOVA (post-hoc Tukey test) for each subject. Significant differences for SS were seen between all subjects except A and B and C and D. No significant differences between subjects were seen for the other three visual fields. This verifies the validity of the VRM for measuring the influence of the visual disturbance on the subject's control strategy.

Since interesting VRM trends were seen with the individual subjects, we decided to look at population trends. The VRM order from smallest to largest (and thus increasing response at the visual disturbance frequencies) for the subject population was CON, FIX, DARK, CVR, CVL, and SS. These results are summarized in table 5.24, and in graphical form for easy comparison in figure 5.15b. Again, we performed an ANOVA (post-hoc Tukey test) on the pooled VRM data, the results of which are tabulated in table 5.25. SS had a VRM significantly different from CON, CVR, DARK, and FIX ($p < 0.001$), as well as from CVL ($p < 0.05$). No significant differences were seen between any other pairs of visual fields.

	A	B	C	D	E	F
CON	5	2	4	1	3	6
DARK	5	2	3	1	4	6
FIX	5	2	3	1	4	6
SS	5	4	1	2	6	3

Table 5.23: Subject rankings for VRM (1=maximum, 6=minimum).

CON	CVL	CVR	DARK	FIX	SS
0.0192 ± 0.0012	0.0391 ± 0.0062	0.0278 ± 0.0039	0.0261 ± 0.0020	0.0213 ± 0.0013	0.0655 ± 0.0058

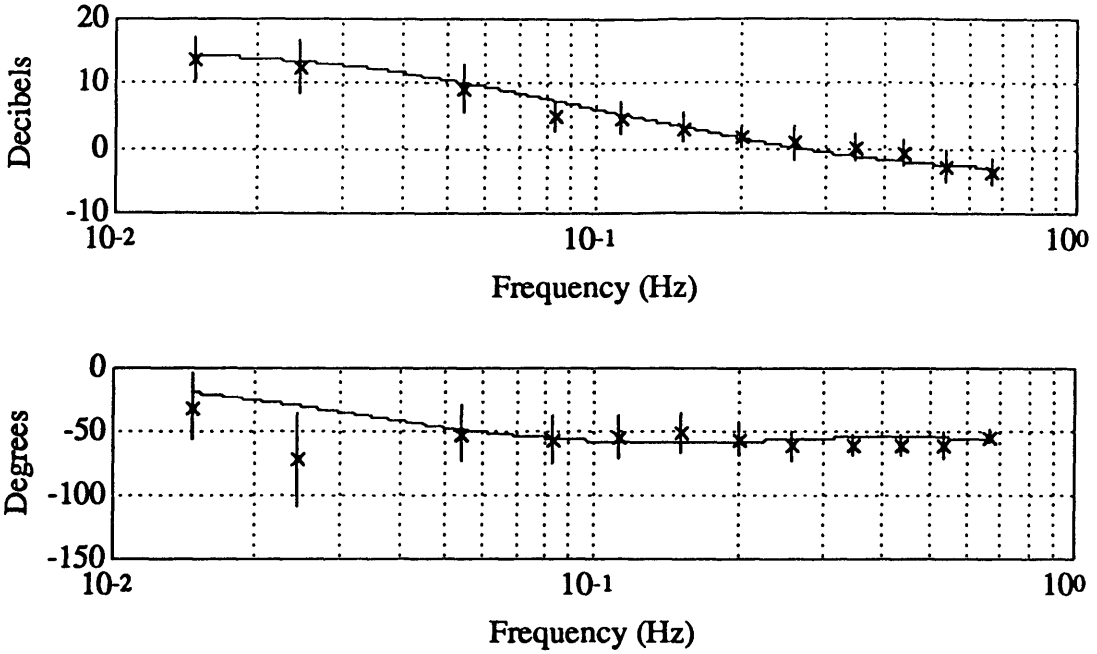
Table 5.24: Population VRM (± se).

	CON	CVL	CVR	DARK	FIX	SS
CON	1.000					
CVL	0.244	1.000				
CVR	0.931	0.882	1.000			
DARK	0.934	0.698	1.000	1.000		
FIX	1.000	0.364	0.979	0.986	1.000	
SS	<0.001	<0.05	<0.001	<0.001	<0.001	1.000

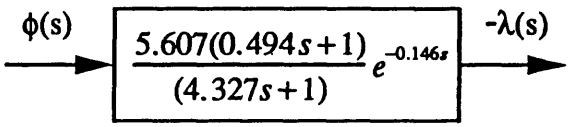
Table 5.25: Population ANOVA results for VRM.

5.2.3.4. Operator Describing Functions

Due to the poor quality of the individual operator describing function fits (see section 5.2.2.3), we decided to pool the individual frequency domain operator describing function data and perform a single fit to the mean data. The same fit parameter constraints and initial values given in table 5.17 were used for the fits to the pooled operator describing functions. Figures 5.16 - 5.21 are Bode plots of the population mean operator describing functions for the six different visual fields used in this experiment. A transfer function fit to the data is shown by the solid line in each figure (see section 4.4.2.2 in chapter 4 for a discussion of the fitting process); the parameters for each of the fits are given in table 5.26. In general, very high quality fits were obtained for the pooled operator describing functions. Fit quality with CVL and CVR was much improved over that with the individual subjects, since pooling the data allowed for computation of frequency-by-frequency fit weights.

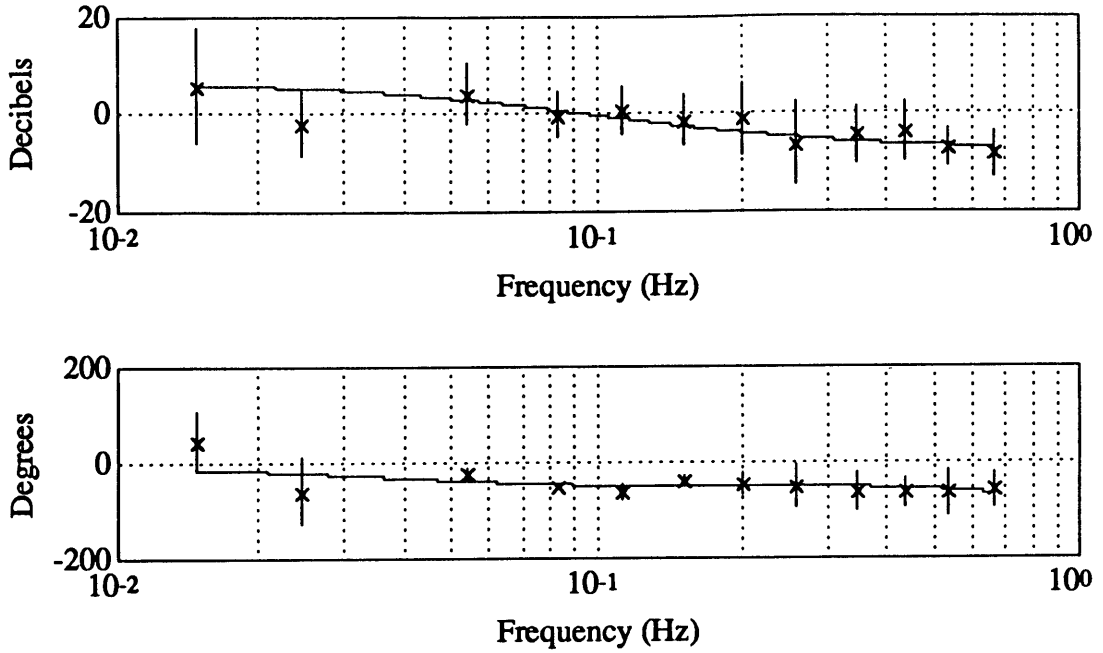


(a)

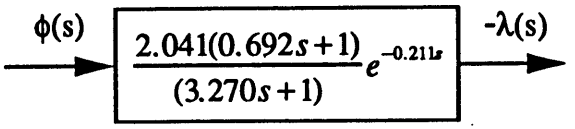


(b)

Figure 5.16: Population mean (± sd) operator describing function for CON visual field (a) and transfer function fit shown by solid line (b).

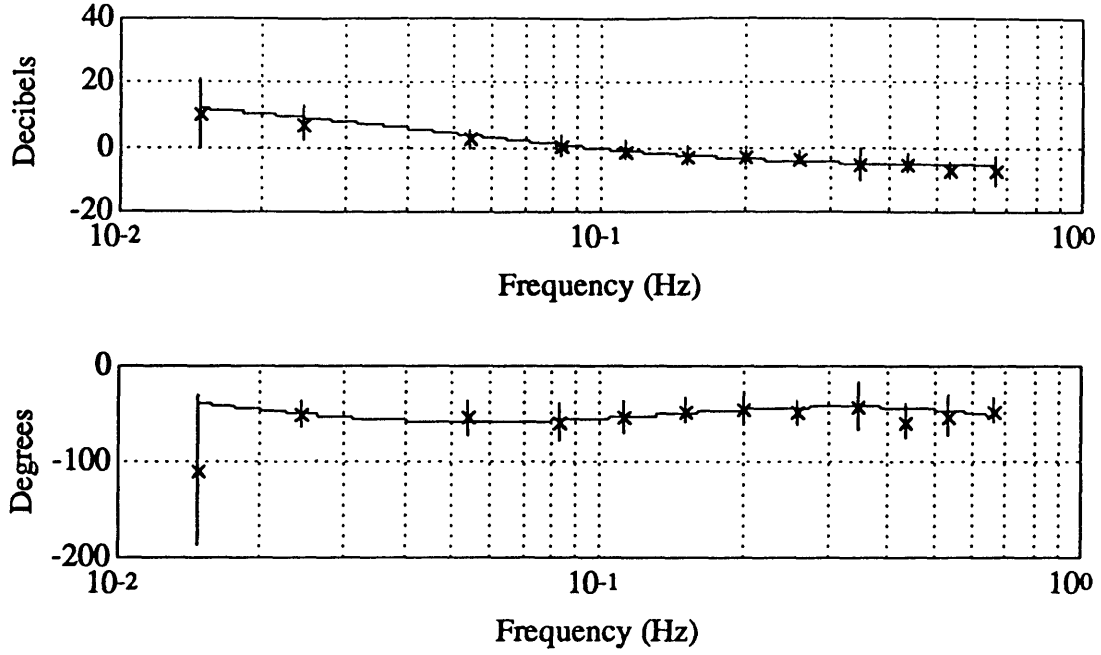


(a)

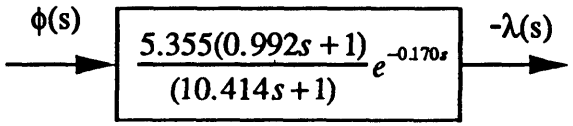


(b)

Figure 5.17: Population mean (± sd) operator describing function for CVL visual field (a) and transfer function fit shown by solid line (b).

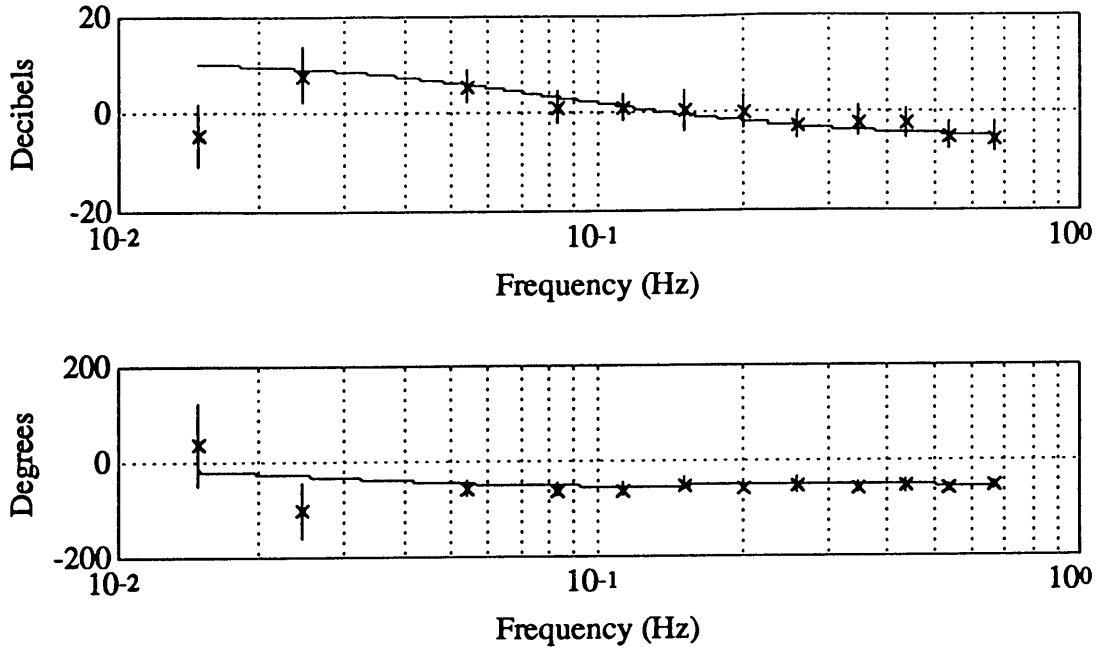


(a)

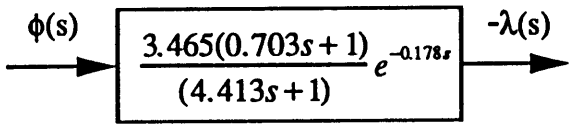


(b)

Figure 5.18: Population mean (± sd) operator describing function for CVR visual field (a) and transfer function fit shown by solid line (b)

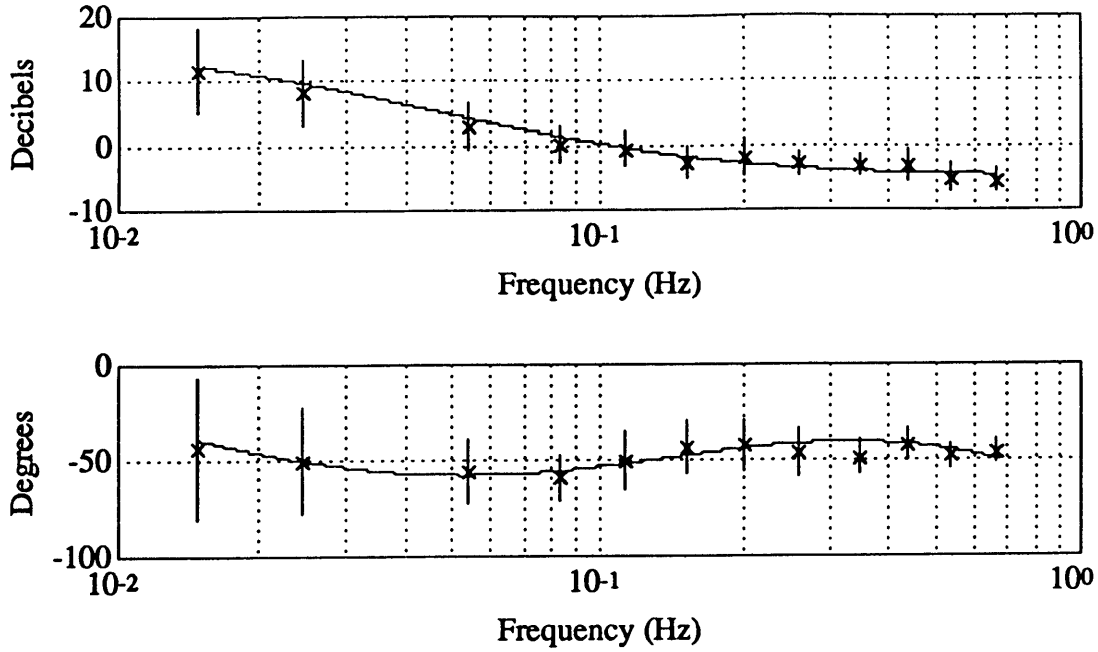


(a)

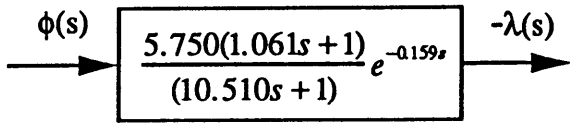


(b)

Figure 5.19: Population mean (± sd) operator describing function for DARK visual field (a) and transfer function fit shown by solid line (b).

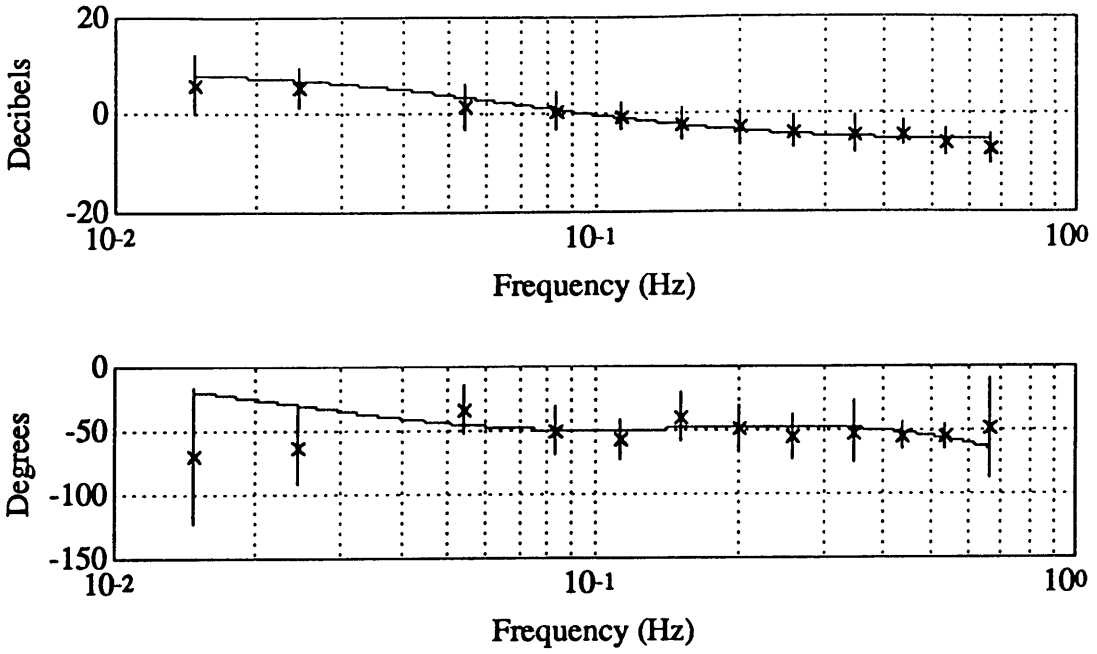


(a)

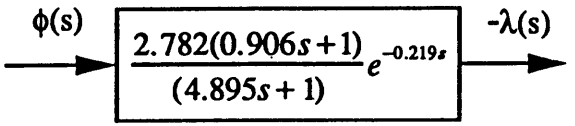


(b)

Figure 5.20: Population mean (± sd) operator describing function for FIX visual field (a) and transfer function fit shown by solid line (b).



(a)



(b)

Figure 5.21: Population mean (± sd) operator describing function for SS visual field (a) and transfer function fit shown by solid line (b).

	CON	DARK	FIX	CVR	SS	CVL
K	5.607	3.465	5.750	5.355	2.782	2.041
τ_1	0.494	0.703	1.061	0.992	0.906	0.692
τ_2	4.327	4.413	10.510	10.414	4.895	3.270
τ_d	0.146	0.178	0.159	0.170	0.219	0.211

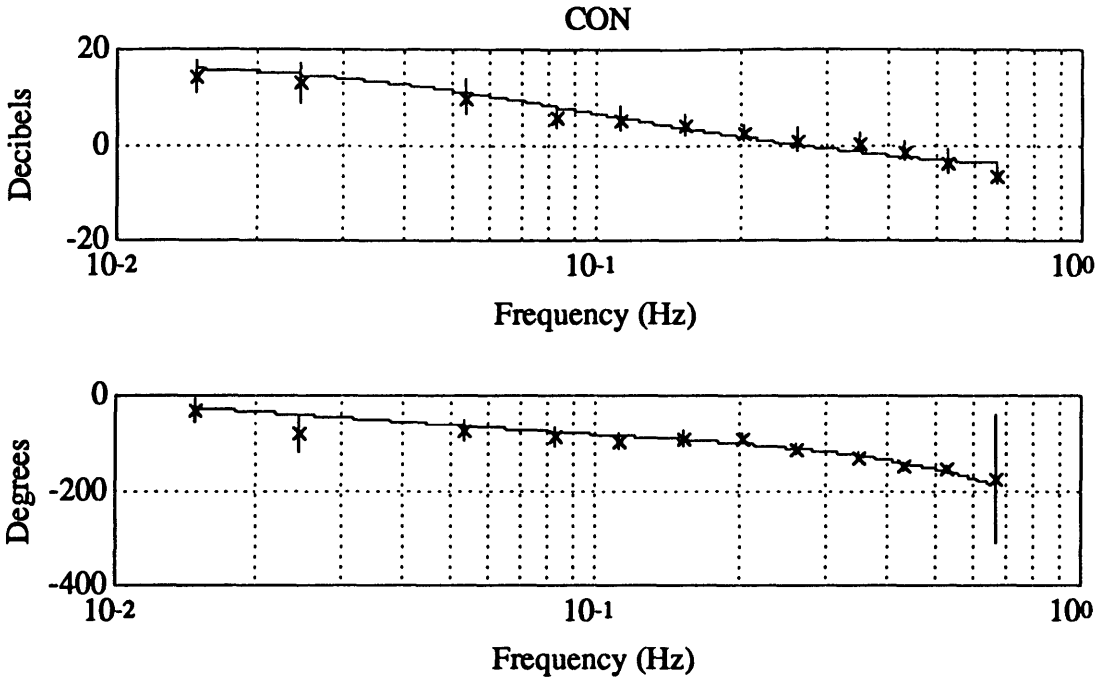
Table 5.26: Summary of mean operator describing function fit parameters for each visual field.

5.2.3.5. Open-Loop Transfer Functions

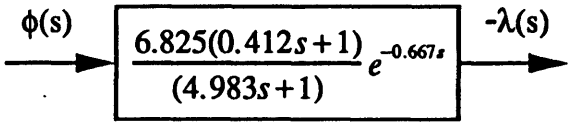
Figures 5.22 - 5.27 are Bode plots of the population mean open-loop transfer functions (product of the operator describing function and Link dynamics) for the six different visual fields used in this experiment. A transfer function fit to the data is shown by the solid line in each figure (see section 4.4.2.2 in chapter 4 for a discussion of the fitting process); the parameters for each of the fits are summarized in table 5.27. The same fit parameter constraints and initial values given in table 5.17 were used for the open-loop transfer function fits. Fit quality was certainly as good as with the pooled operator describing functions, which is not surprising since both the operator describing functions and the closed-loop transfer function of the trainer were well behaved (see figure 3.1 in chapter 3), and thus the product should be as well. The fit quality would likely be improved by the addition of a low-frequency washout term to capture the reduced gain at low frequencies. Little in the way of predictive accuracy was to be gained by resorting to this higher order model, however, so these fits were not performed. In chapter eight, we compare the control strategy of the pooled subjects in the manual roll stabilization task with that predicted by the McRuer Crossover Model.

	CON	DARK	FIX	CVR	SS	CVL
K	6.825	5.211	7.914	7.652	3.796	3.252
τ_1	0.412	0.633	0.697	0.689	0.639	0.516
τ_2	4.983	6.464	11.252	12.010	5.267	4.841
τ_d	0.667	0.680	0.623	0.636	0.686	0.667

Table 5.27: Summary of mean open-loop transfer function fit parameters for each visual field.

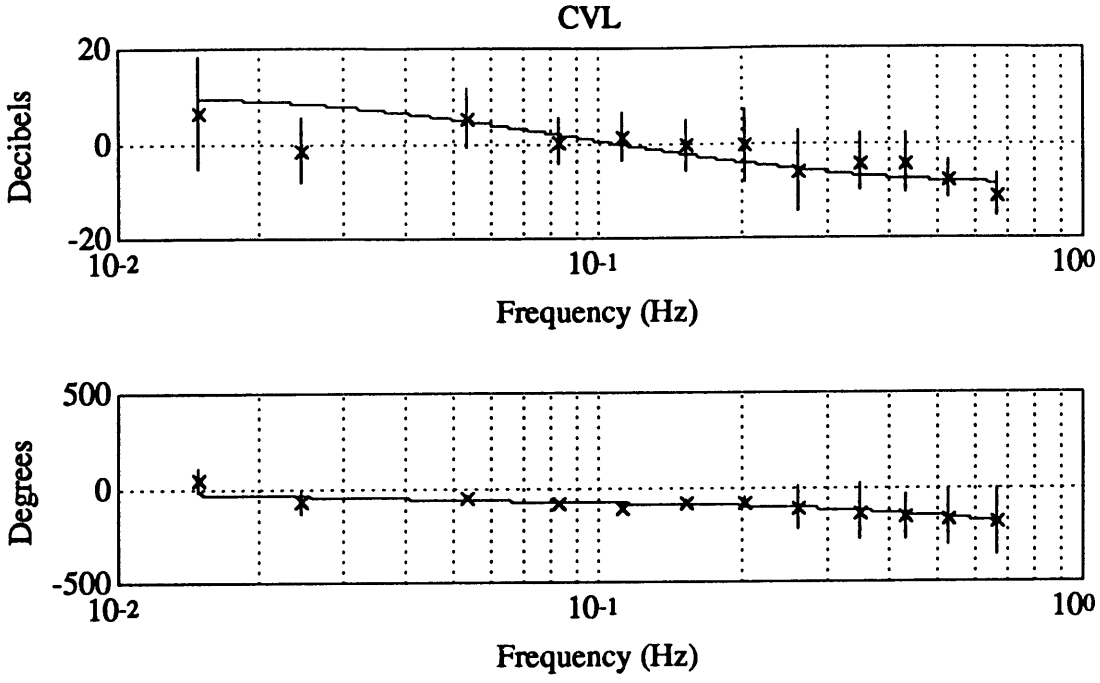


(a)

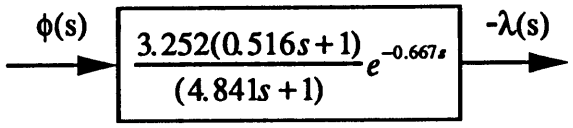


(b)

Figure 5.22: Population mean (± sd) open-loop transfer function for CON visual field (a) and transfer function fit shown by solid line (b).

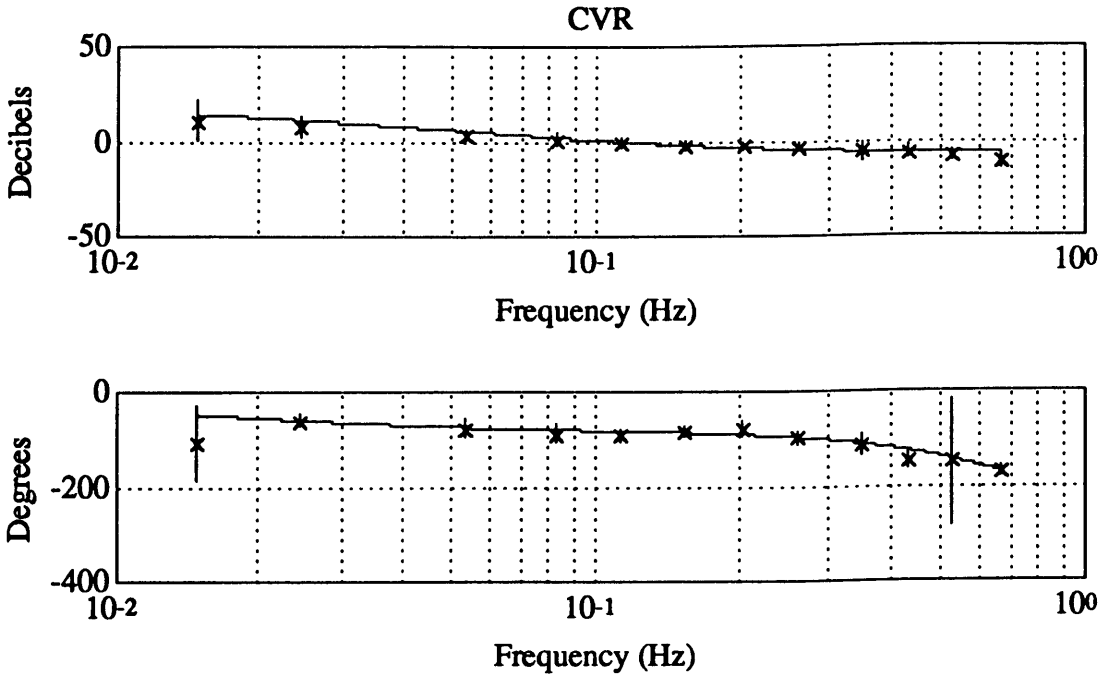


(a)

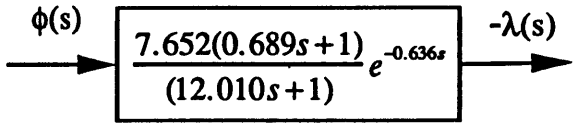


(b)

Figure 5.23: Population mean (± sd) open-loop transfer function for CVL visual field (a) and transfer function fit shown by solid line (b).

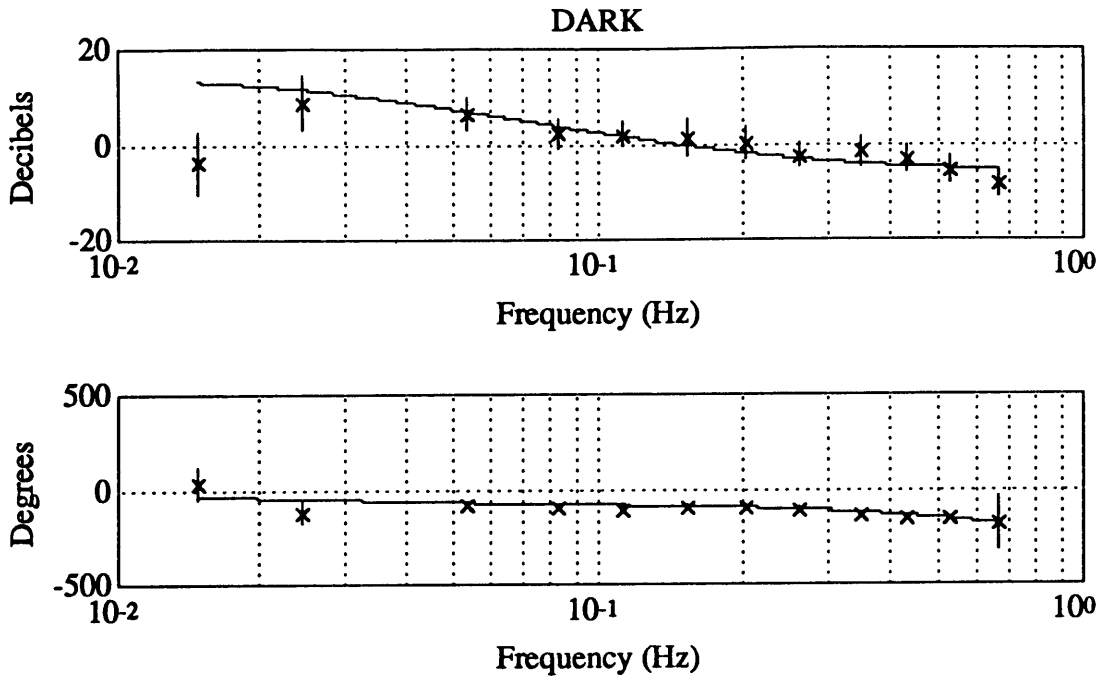


(a)

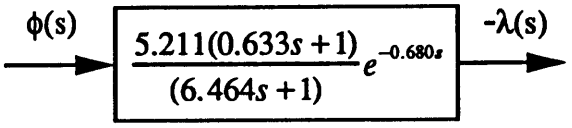


(b)

Figure 5.24: Population mean (± sd) open-loop transfer function for CVR visual field (a) and transfer function fit shown by solid line (b)

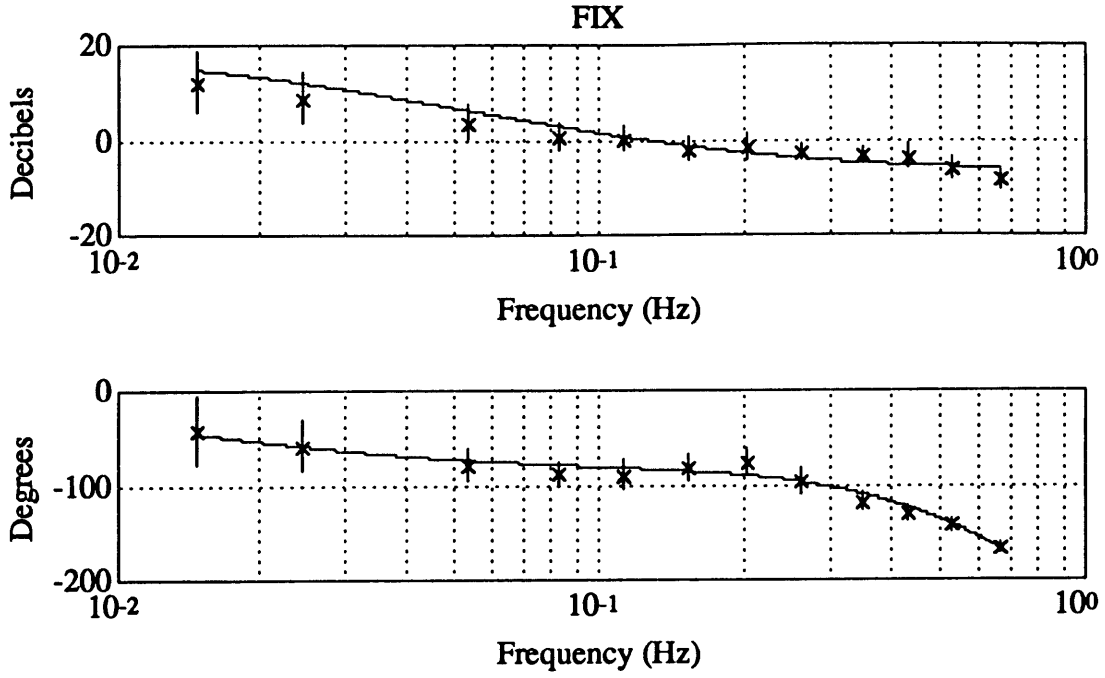


(a)

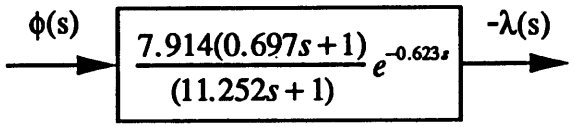


(b)

Figure 5.25: Population mean (± sd) open-loop transfer function for DARK visual field (a) and transfer function fit shown by solid line (b).

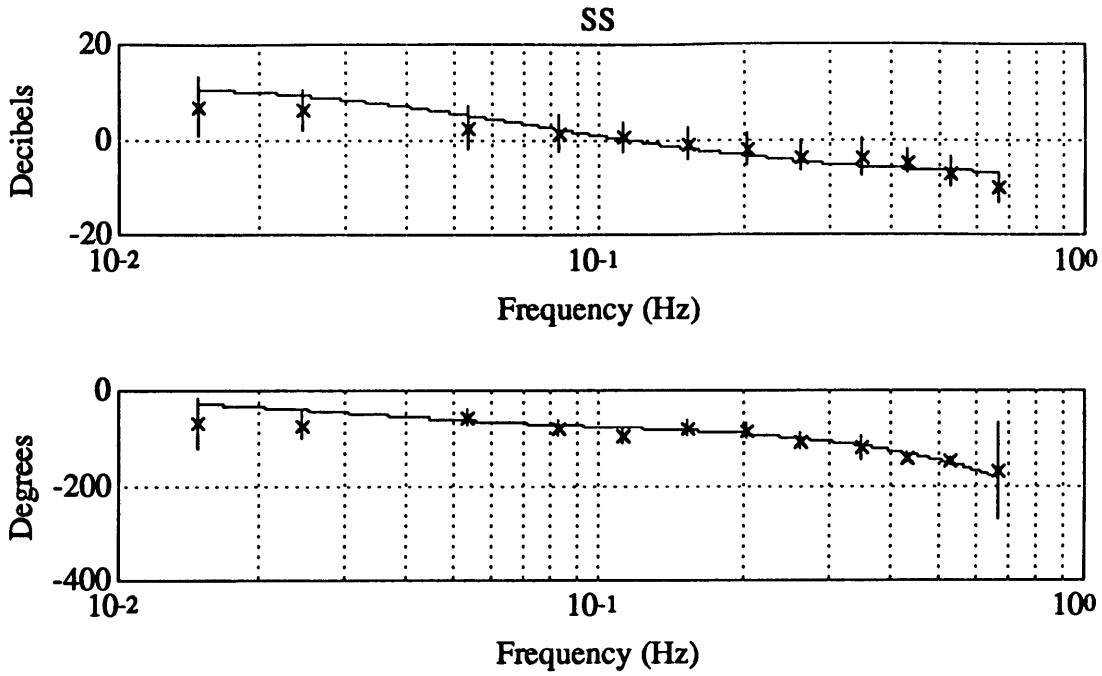


(a)

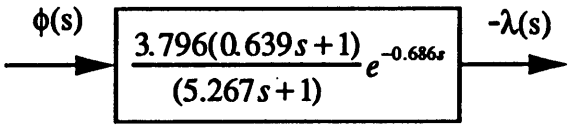


(b)

Figure 5.26: Population mean (± sd) open-loop transfer function for FIX visual field (a) and transfer function fit shown by solid line (b).



(a)



(b)

Figure 5.27: Population mean (± sd) open-loop transfer function for SS visual field (a) and transfer function fit shown by solid line (b).

PART THREE

INFLUENCE OF THE VISUAL FIELD ON MANUAL LATERAL STABILIZATION

6. EXPERIMENTAL METHOD FOR THE LATERAL STABILIZATION EXPERIMENTS

6.1. Experimental Apparatus

This section discusses the motion platform, windowshade, subject control, countermoving visual field circuitry, and supplementary equipment used in the manual lateral stabilization experiments.

6.1.1. Motion Platform

The US Laboratory Sled consists of a cushioned aluminum chair (ESA Space Sled Chair) and instrumented head restraint mounted on a cart which is guided along two cylindrical rails by four pillow blocks with recirculating ball bearing bushings. A cable attached to both sides of the cart is wound around a pulley at one end and a winch drum at the other. The cable was held under 600 lbs of tension and the winch drum was driven by a permanent magnet torque motor. The entire system is controlled by a velocity controller using tachometer feedback. Velocity commands to the controller are generated by a 386 IBM clone computer running a special software package developed for the MVL by Payload Systems, Inc. The velocity commands follow mathematical trajectories which can be combined with a joystick signal under the control of the subject. The sled is capable of controlled accelerations from 0.001 g to 0.7 g over an effective usable track length of 4.7 meters. Figure 6.1 shows a Bode plot of the sled closed-loop dynamics for velocity control.

To restrain the subject during linear accelerations, there is a 5 point strap restraint system. The subject's head is fixed within a molded housing attached to the sled which can be raised or lowered to fit the subject's torso. Under the moderate accelerations achieved in this experiment, these restraints kept the subject well fixed in the sled.

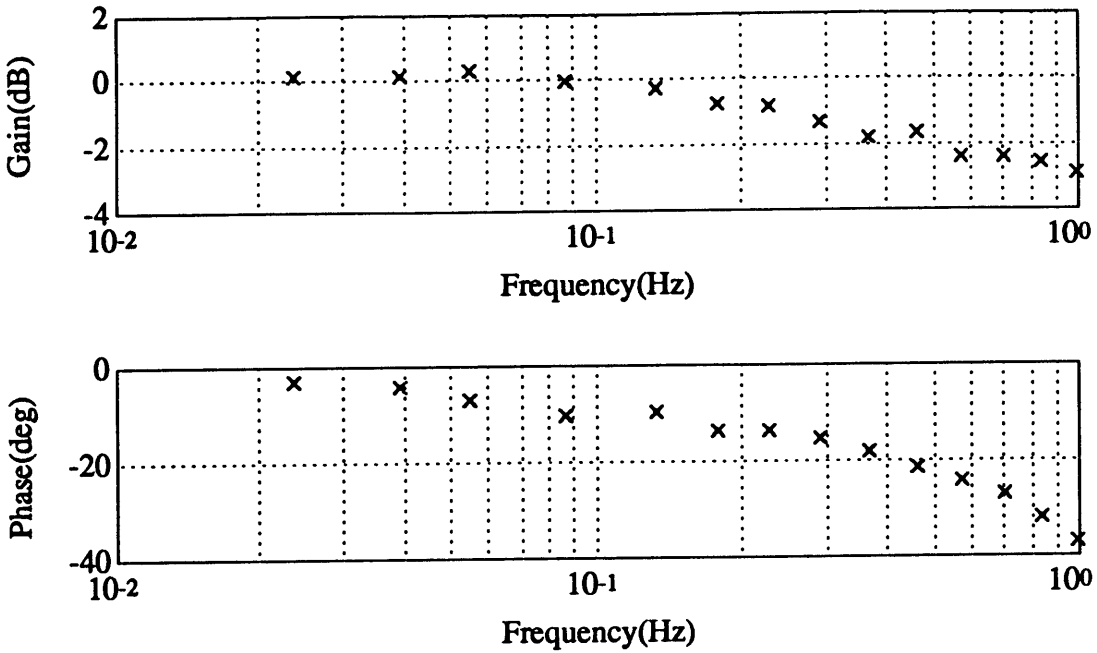


Figure 6.1: Sled closed-loop velocity dynamics.

6.1.2. The Windowshade

Visual stimuli are supplied by a infinite cloth loop called the windowshade which is mounted in front of the subject. Painted on the windowshade are alternating black and neon yellow stripes. When the subject is seated in the sled looking forward with his head restrained, the windowshade subtend approximately 88 degrees in the vertical direction (44 degrees up and 44 degrees down), and 88 degrees in the horizontal direction (44 degrees left and 44 degrees right). The alternating black and neon yellow stripes each subtend angles of approximately 4 degrees. The motor which moves the windowshade is driven closed-loop (velocity servo compensated). The velocity commands follow mathematical trajectories generated by the MVL Sled Software auxiliary channel. Figure 6.2 shows a Bode plot of the closed-loop projector dynamics for velocity control. Note that the dynamics of the shade are sufficiently fast to follow the computer commands, as well as sled velocity during countermotion mode.

6.1.3. Subject Control

A hand-held, spring-centered joystick was used for nulling sled velocity. The control fits comfortably in the subject's hand and sled velocity corresponds to the direction the joystick is moved.

6.1.4. Countermoving Visual Field Circuitry

In order to perform the trials with the countermoving visual field (confirming visual cue), a circuit had to be built which would allow the operator to choose between control of the visual field by the auxiliary channel, or by the sled tachometer signal (with appropriate scaling). To compute the appropriate scaling for the tach signal, we will need the following sled scalefactors:

1. Windowshade Command : 0.286 m/s/volt
2. Sled Tach Voltage : 0.73 volt/(m/s)

Now assume a 1 m/s sled velocity, which corresponds to a sled tach signal of 0.73 volts. Now the command required to generate a windowshade velocity of 1 m/s is 1/0.286 =

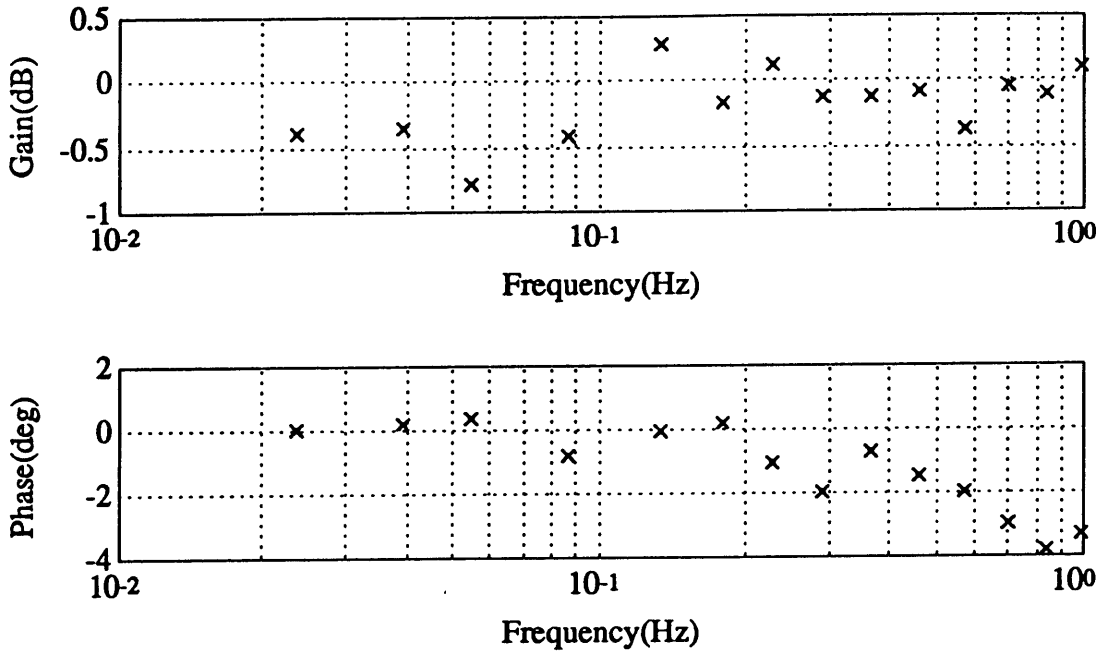


Figure 6.2: Windowshade closed-loop velocity dynamics.

3.497 volts. Thus the required gain on the sled tach signal is 3.497/0.73 = 4.790. A non-inverting circuit which can supply this gain and allow the operator to choose whether the windowshade is controlled by the sled computer or operating in countermotion mode is shown in figure 6.3.

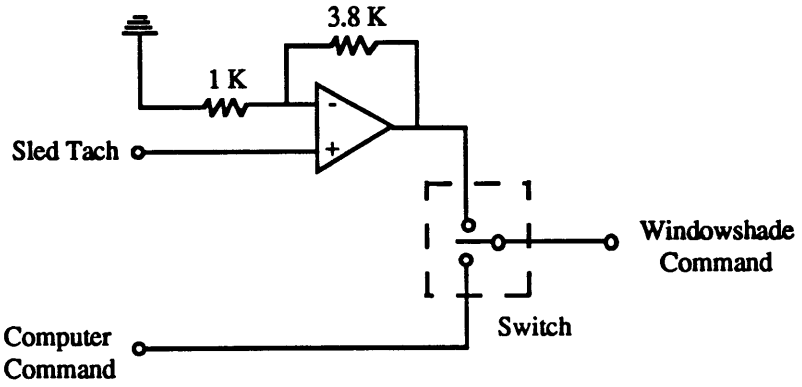


Figure 6.3 : Countermoving visual field circuitry.

6.1.5. Supplementary Equipment

A Macintosh II running LabView ™ was used for data acquisition, with data sampled at 64 Hertz. Sled position and velocity were recorded, as well as the subject's joystick velocity command and the visual field velocity. Each subject wore an earpiece, which together with a microphone mounted near the subject's mouth allowed two-way communication with the sled operator. Various steps were taken to reduce non-vestibular motion cues. Wind cues were eliminated by having the subject wear appropriate clothing (including gloves) so that no skin was exposed. Auditory cues were reduced by added white noise, and vision was eliminated, when appropriate, by a light-tight shroud.

6.2. Subjects and Stimulation

6.2.1. Subjects

Six subjects participated in the experiment, 4 males and 2 females, age 23 to 33, and in normal health. Three of the subjects from the manual roll stabilization experiment also took part in this experiment, while the other three were new subjects with no experience in manual lateral stabilization.

6.2.2. Vestibular Stimulation

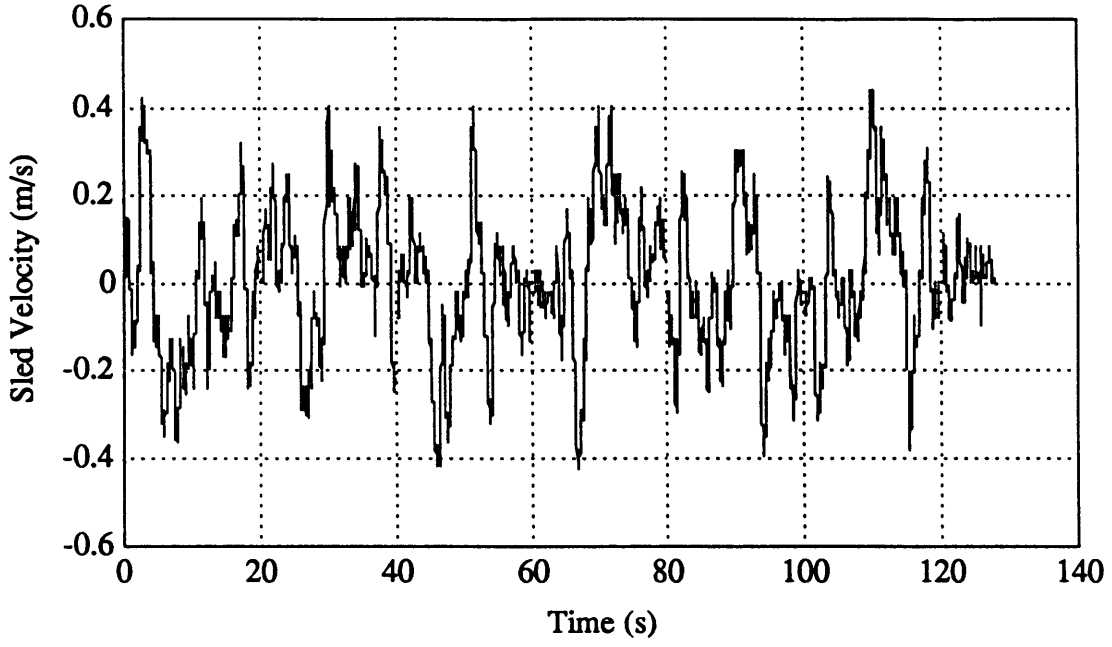
The motion disturbance sent to the sled is identical for all trials, and is a pseudo-random zero-mean sum of sines velocity command with the parameters shown in table 6.1. Guidelines for the disturbance design were provided by the work of Arrott and Huang [1,13]. A plot of sled velocity in response to the disturbance is shown in figure 6.4a.

6.2.3. Visual Stimulation

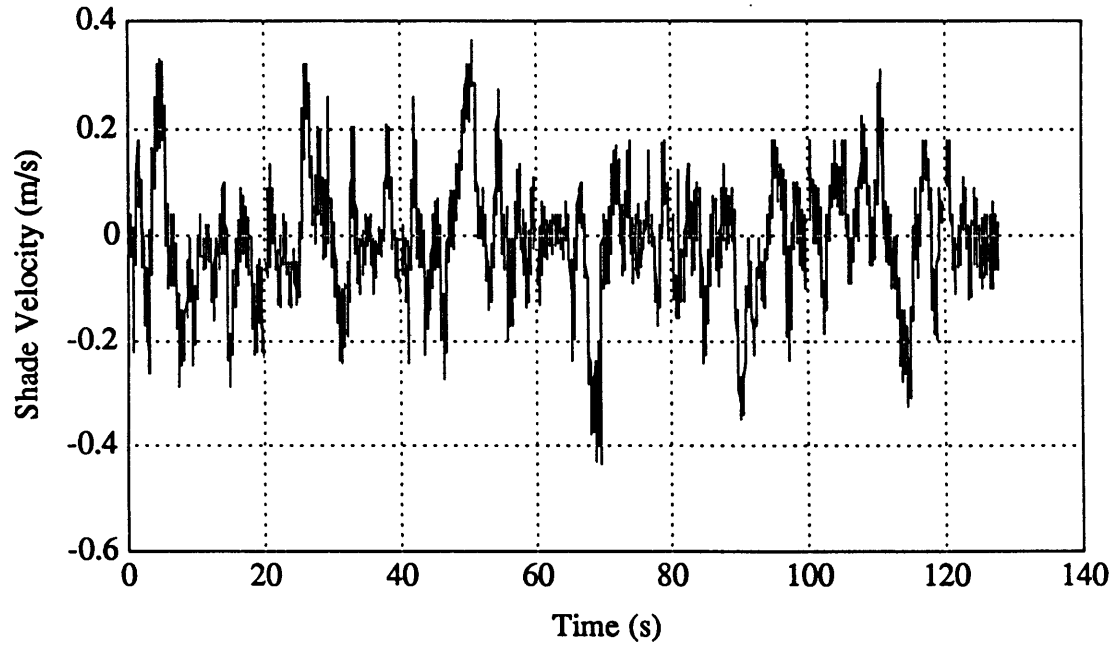
There were four different types of visual fields used in the experiments: DARK, FIX, CON, and SS. DARK trials had only a red fixation point (supplied by a laser) visible to the subject, requiring the subject to depend solely on vestibular cues. FIX supplied a stationary visual field with respect to the subject, again requiring the subject to depend on vestibular cues only. CON provided a visual field which moved in the direction opposite to that of the sled but with the same speed. These confirming visual cues mimic what we see in everyday life. And finally, SS was a pseudo-random zero-mean velocity command with the parameters shown in table 6.2. Guidelines for the disturbance design were provided by the work of Huang [13]. As shown in figure 6.5, the visual stimulus frequencies are interleaved among the movement stimulus frequencies; this again ensures that the stimuli are completely uncorrelated. A plot of visual field velocity in response to the disturbance is shown in figure 6.4b.

6.3. Experimental Procedure

All six subjects were instructed to keep the sled "as motionless as possible" by concentrating on their sensed velocity and providing appropriate compensatory commands using the joystick. Figure 6.6 shows the overall linearized loop model of the lateral velocity nulling task (modified from Huang [13]). Note that both visual and vestibular cues are assumed to be used by the subject to estimate his velocity. The subject was given four practice trials with CON, FIX, DARK, and SS visual fields. During this time, the white noise volume was adjusted to a level comfortable for the subject but sufficient to



(a)



(b)

Figure 6.4 : Response of sled (a) and visual field (b) to pseudo-random velocity disturbances.

Sum of Sines Vestibular Disturbance

---

1. Duration of profile: 128 seconds
2. Fundamental Frequency: 0.0078 HZ
3. Number of Sinusoids: 10
4. Successive Phase Angle: 247 degrees
5. Maximum Velocity: 0.445 m/s
6. Maximum Acceleration: 0.11 G

Frequency (Hz)	Velocity (m/s)	Acceleration (G)	Phase (degrees)
0.023	0.0651	0.0009	0
0.054	0.0869	0.0030	247
0.101	0.0941	0.0061	134
0.148	0.1013	0.0096	21
0.226	0.0941	0.0136	268
0.289	0.0796	0.0147	155
0.367	0.0724	0.0170	42
0.476	0.0724	0.0221	289
0.648	0.0579	0.0240	176
0.882	0.0434	0.0245	63

Table 6.1: Sum of sines vestibular disturbance used in sled manual lateral stabilization experiment.

Sum of Sines Visual Disturbance

---

1. Duration of profile : 128 seconds
2. Fundamental Frequency: 0.0078 HZ
3. Number of Sinusoids: 10
4. Successive Phase Angle : 247 degrees
5. Maximum Angular Velocity: 0.3163 m/s

Frequency (Hz)	Velocity (m/s)	Accel. (m/s/s)	Phase (degrees)
0.015	0.0348	0.0034	0
0.039	0.0523	0.0128	247
0.085	0.0639	0.0345	134
0.132	0.0581	0.0485	21
0.179	0.0581	0.0656	268
0.242	0.0581	0.0884	155
0.320	0.0523	0.1052	42
0.414	0.0465	0.1209	289
0.570	0.0406	0.1457	176
0.789	0.0290	0.1440	63

Table 6.2: Sum of sines visual disturbance used in sled manual lateral stabilization experiment.

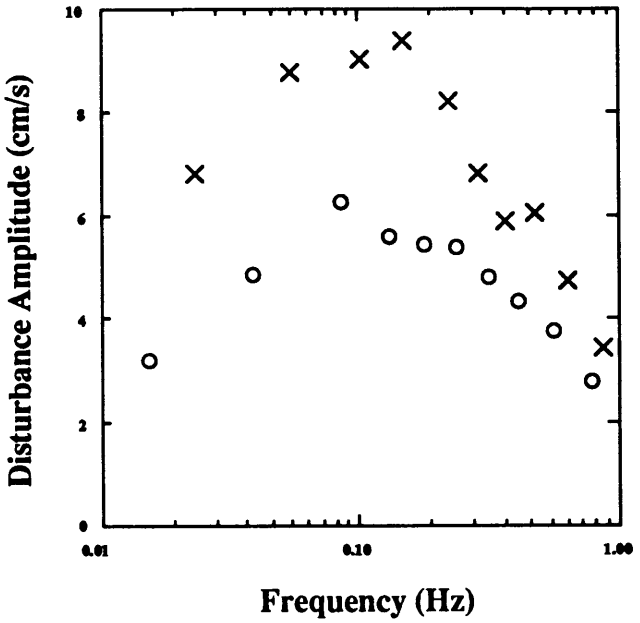


Figure 6.5: Interleaving of vestibular (x's) and visual disturbance components (o's).

mask external audio cues. If necessary, subjects were given additional practice sessions with the fields until they felt comfortable with the nulling task. There were 10 runs for each subject with the following order of visual presentations: DARK, SS, FIX, CON, SS, FIX, SS, DARK, SS, CON. Individual trials were repeated until the subject was able to complete the run. A complete experimental session took approximately 1 hour (more if many repeat trials were required).

6.4. Time and Frequency Domain Analysis Methods

The analytical and statistical methods developed in section 4.4 of chapter 4 were used to analyze subject performance in the manual lateral stabilization task with one small change: Link position signals were replaced with sled velocity signals, since the subject's task was to null velocity. Time domain measurements were mean and RMS sled velocity and frequency domain measurements were the scalar performance measure (SPM), visual response measure (VRM), and operator describing function. For the VRM, the visual disturbance was not integrated, since the subject's task was to null velocity. The derived

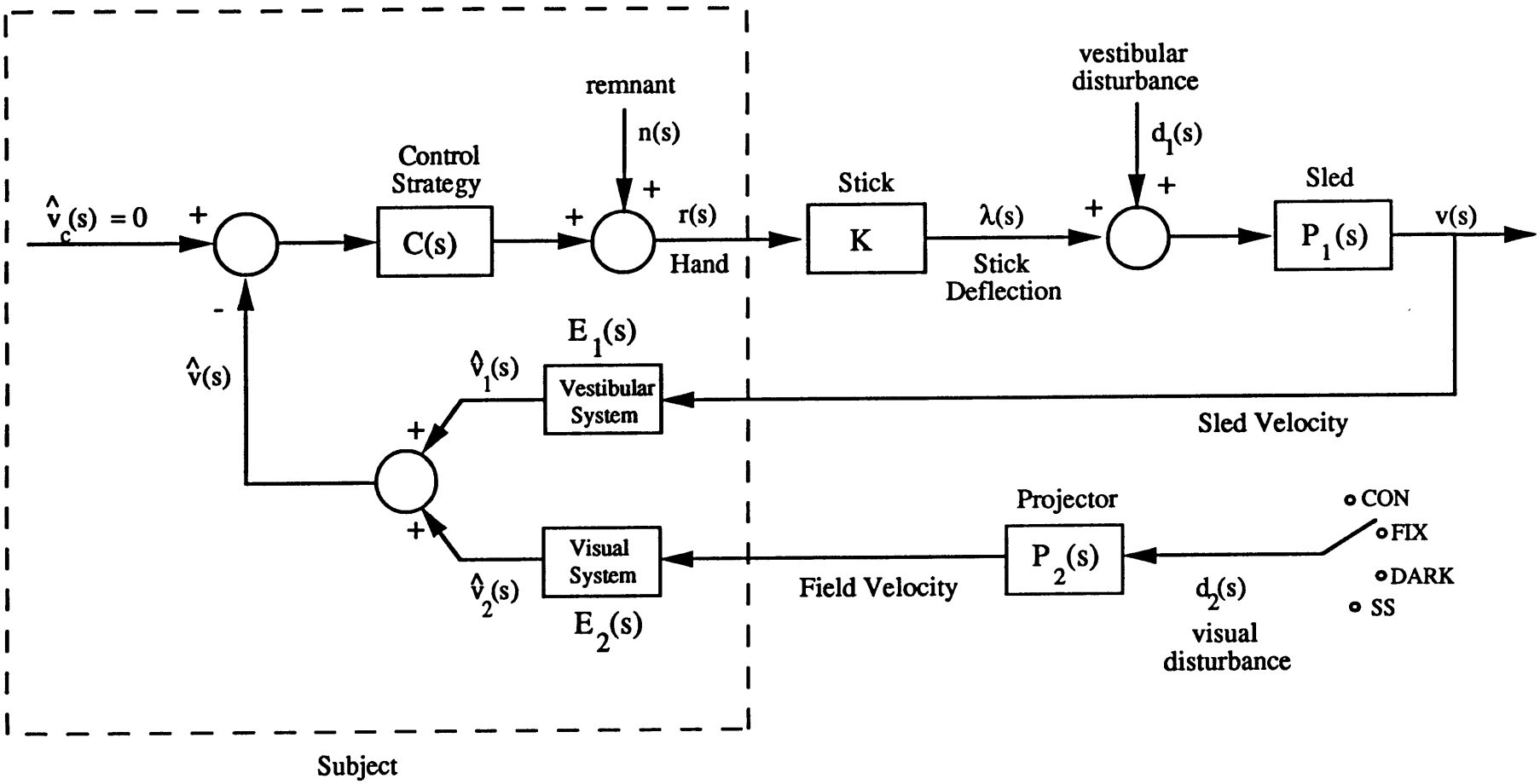


Figure 6.6: Linearized model of closed-loop velocity nulling.

result still applies, however, if $D_2(i)$ is interpreted as the velocity amplitude of the visual disturbance at the i^{th} visual disturbance frequency. This can be obtained using a simple PSD of the visual disturbance.

7. RESULTS OF THE LATERAL EXPERIMENTS

7.1. Time Domain Analysis of Subject Response

This section presents representative subject response time histories for each of the four different visual fields used in this study and summarizes the individual and population trends seen in the mean and RMS sled velocity.

7.1.1. Representative Time Histories

Figure 7.1 shows a set of typical time histories for the sled position, sled velocity, and subject's compensatory joystick response. Since the visual field is counterrotating (CON), operator performance in this situation results in a well-met task objective. The sled reached maximum velocities of 29.86 and -37.69 cm/s, and had a mean of -0.84 cm/s (note that positive velocities denote rightward sled motion, and negative velocities denote leftward sled motion). Figure 7.2 shows subject performance when the trial is done in the dark, with only a red fixation point (supplied by a laser) visible to the subject (DARK). Performance is slightly worse than with CON, with the sled reaching maximum velocities of 40.57 and -49.73 cm/s, and a mean of 0.07 cm/s. Similarly, figure 7.3 illustrates subject performance in the fixed visual field, where the visual field is held stationary with respect to the subject (FIX). Performance is worse than with CON or DARK, with the sled reaching maximum velocities of 55.28 and -56.42 cm/s, and a mean of 1.17 cm/s. And finally, figure 7.4 illustrates subject performance with the sum of sines pseudo-random visual field (SS). With this field, the sled reached maximum velocities of 50.60 and -59.77 cm/s, and had a mean of -0.41 cm/s. The runs presented here for subject F are representative of the runs for the other five subjects who took part in this experiment.

7.1.2. Individual Subject Results

This section presents the results for individual subjects for mean and RMS sled velocity.

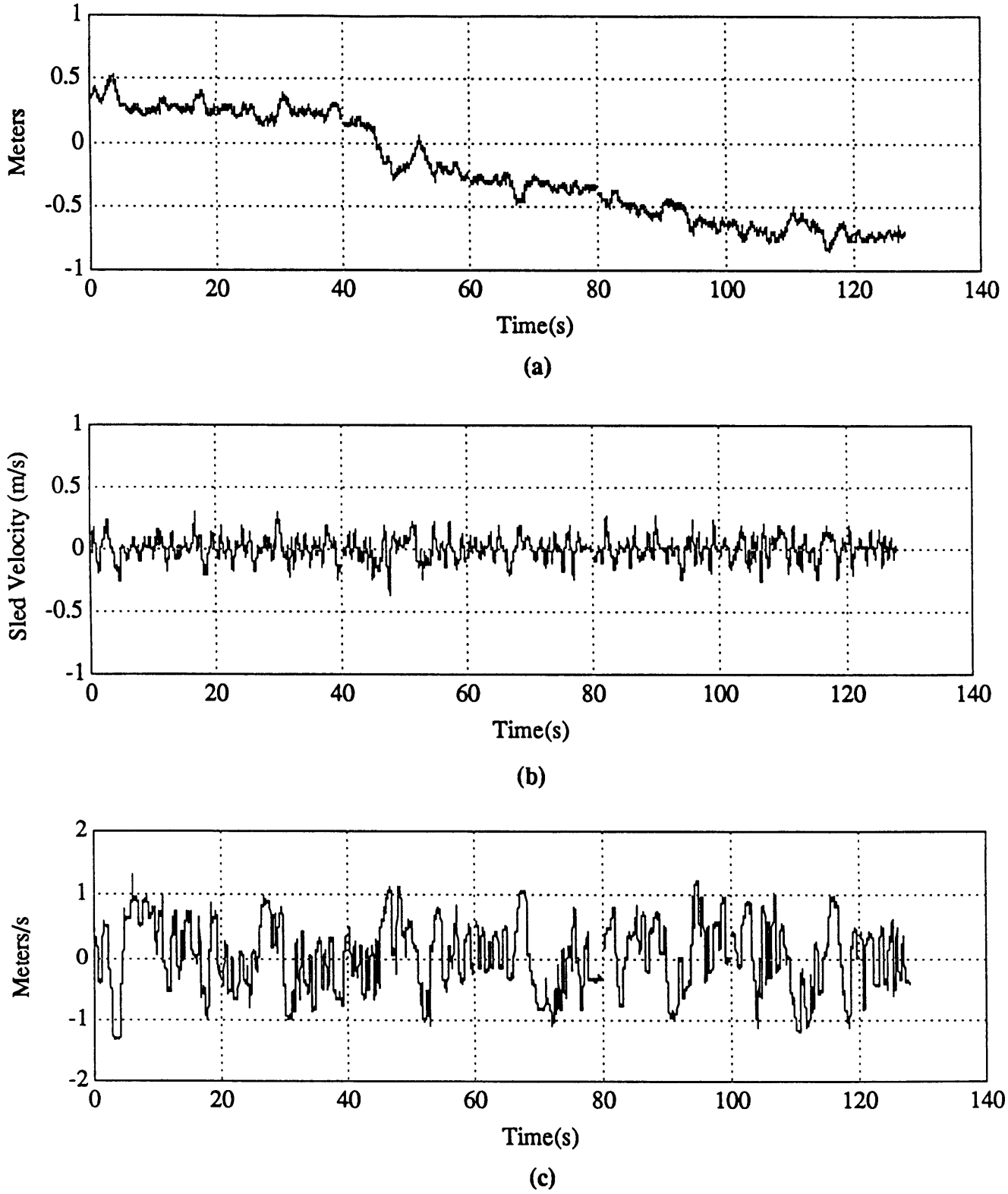


Figure 7.1 : Subject F time history of sled position (a), sled velocity (b), and joystick response (c) for counterrotating visual field (CON).

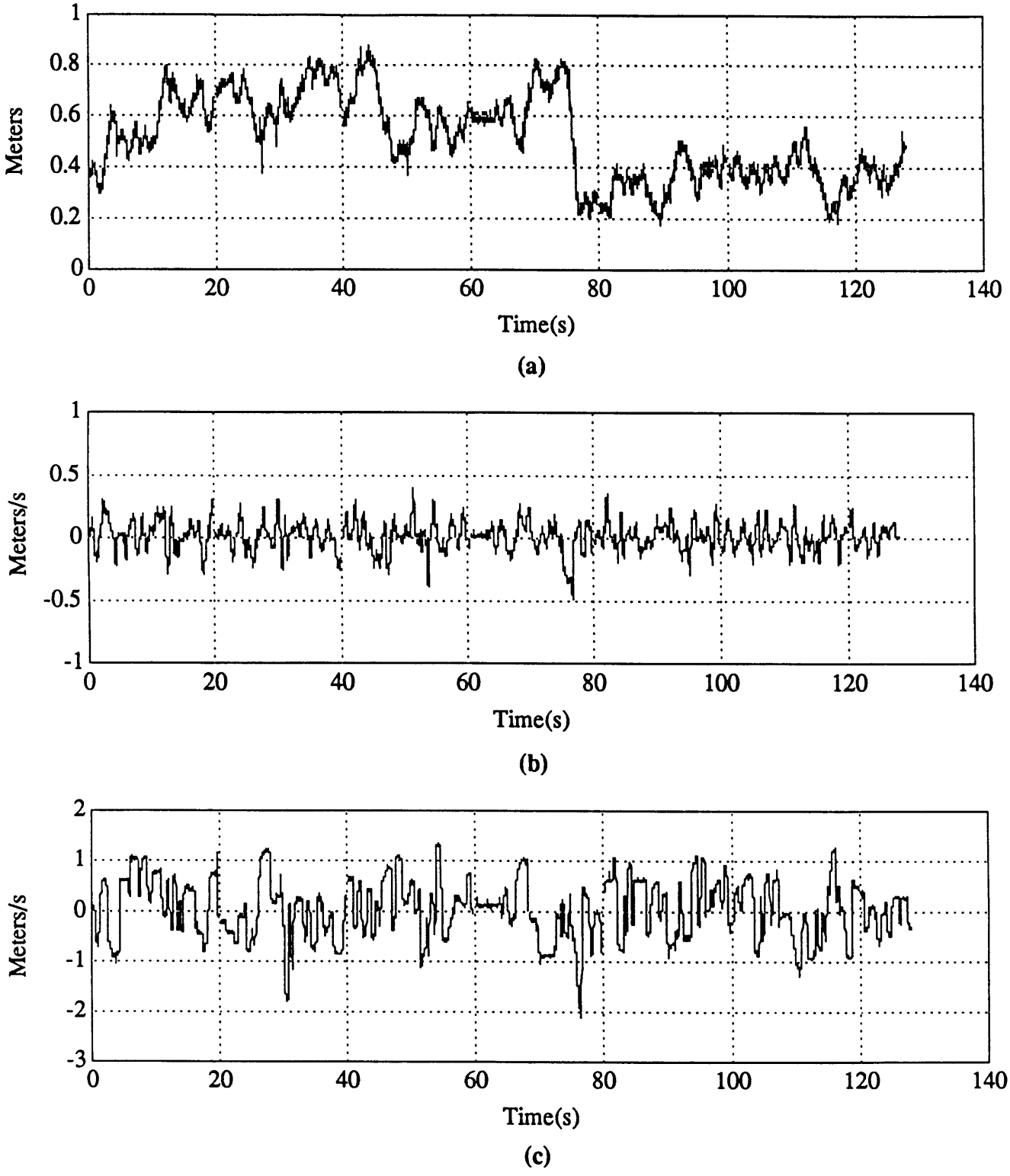


Figure 7.2 : Subject F time history of sled position (a), sled velocity (b), and joystick response (c) for dark visual field (DARK).

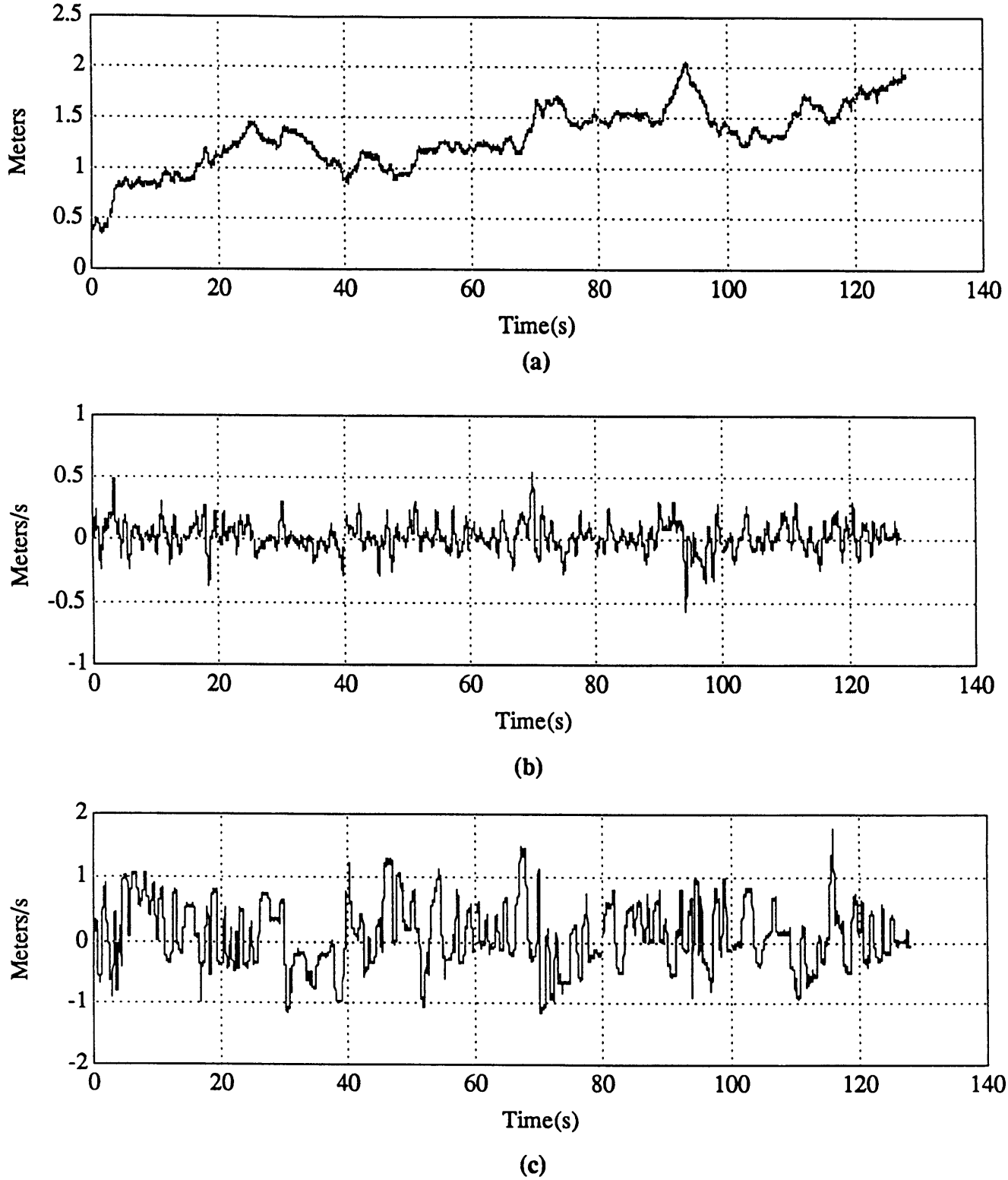


Figure 7.3 : Subject F time history of sled position (a), sled velocity (b), and joystick response (c) for fixed visual field (FIX).

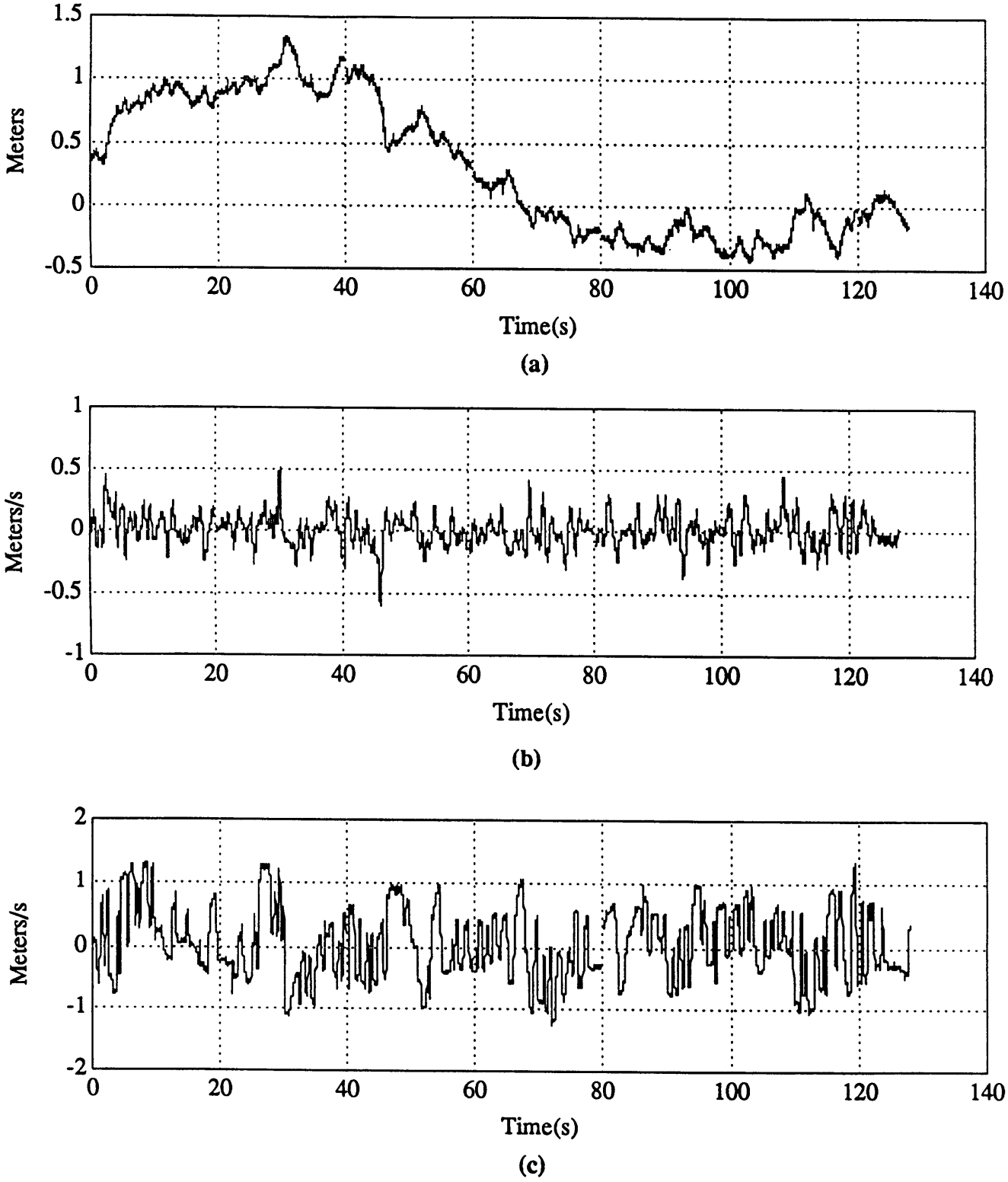


Figure 7.4 : Subject F time history of sled position (a), sled velocity (b), and joystick response (c) for sum of sines visual field (SS).

7.1.2.1. Mean Sled Velocity

The results for mean sled velocities for each subject and visual field condition are given in table 7.1, and in graphical form for easy comparison in figure 7.5. Note the large standard errors and differences between subjects for a given visual field. Two possible reasons for these discrepancies are that (1) too few trials were performed to obtain an accurate measure of the mean, or (2) subjects could not estimate zero velocity accurately enough to give a consistent mean across trials. Whatever the reason, these results show, not surprisingly, that mean sled velocity is a poor measure of subject performance in the nulling task.

	CON	DARK	FIX	SS
B	-1.500 ± 0.331	1.112 ± 0.196	0.062 ± 0.421	0.063 ± 0.143
D	-0.756 ± 0.225	-0.530 ± 0.923	-0.701 ± 0.124	-0.475 ± 0.216
F	-0.905 ± 0.069	-0.062 ± 0.127	1.212 ± 0.045	-0.242 ± 0.281
X	-1.424 ± 0.054	-0.165 ± 0.285	-0.247 ± 1.094	-0.198 ± 0.245
Y	-1.606 ± 0.167	-0.191 ± 0.681	0.320 ± 0.171	-0.514 ± 0.378
Z	-1.470 ± 0.198	-0.481 ± 0.289	-0.020 ± 0.347	-0.460 ± 0.414

Table 7.1: Subject mean sled velocity in cm/s (± se).

To investigate which of these mean velocities were significantly different from the desired zero cm/s (the subject was told to keep the sled "as motionless as possible"), we performed a χ^2 test, the results of which are summarized in table 7.2. All subjects had mean velocities which were statistically different from zero cm/s for CON. This was most likely due to a windowshade velocity drift caused by a slight asymmetry in the counterrotating visual field circuitry. Subject B had a significant difference for DARK, subject D for FIX and SS, and subject F for FIX. No other significant differences were seen.

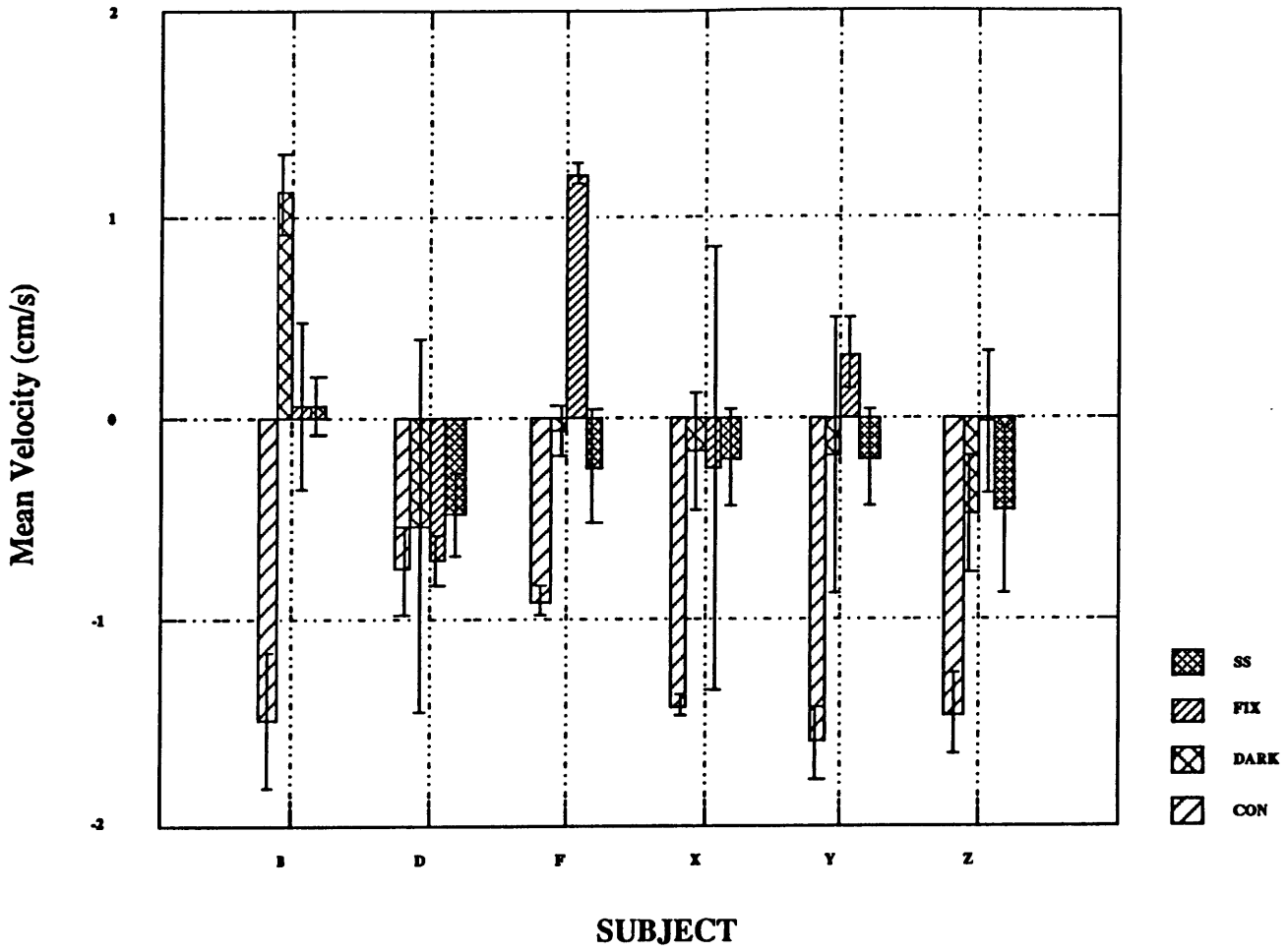


Figure 7.5: Individual subject mean sled velocity (± se) for each of the visual field conditions.

To investigate differences in the visual field effect on mean velocity for each subject, we performed an ANOVA (post-hoc Tukey test). For subjects B, significant differences were seen between CON and DARK ($p < 0.01$), CON and FIX ($p < 0.05$), and CON and SS ($p < 0.05$). For subject F, significant differences were seen between FIX and CON ($p < 0.01$) and FIX and SS ($p < 0.05$). No other significant differences were seen.

	CON	DARK	FIX	SS
B	<0.001	<0.001	0.600	0.203
D	<0.01	0.514	<0.001	<0.05
F	<0.001	0.539	<0.001	0.154
X	<0.001	0.512	0.591	0.161
Y	<0.001	0.583	0.105	0.089
Z	<0.001	0.151	0.606	0.120

Table 7.2: Results of χ^2 test on mean sled velocity.

7.1.2.2. RMS Sled Velocity

The results for RMS sled velocity for each subject and visual field condition are summarized in table 7.3, and in graphical form for easy comparison in figure 7.6. Only subject F had RMS sled velocities which were less than the disturbance RMS of 16.16 cm/s for all visual fields. The remaining subjects had RMS velocities greater than the disturbance for DARK, FIX, and SS (except subject D for DARK). Only subject B was unable to reduce the RMS velocity below the disturbance value for CON (although subject X's RMS velocity for CON was not significantly different from the disturbance). To investigate which of these RMS velocities were significantly different from 16.16 cm/s we performed a χ^2 test, the results of which are given in table 7.4. RMS velocities for subjects B and F were statistically different from the disturbance for all visual fields, with subject F

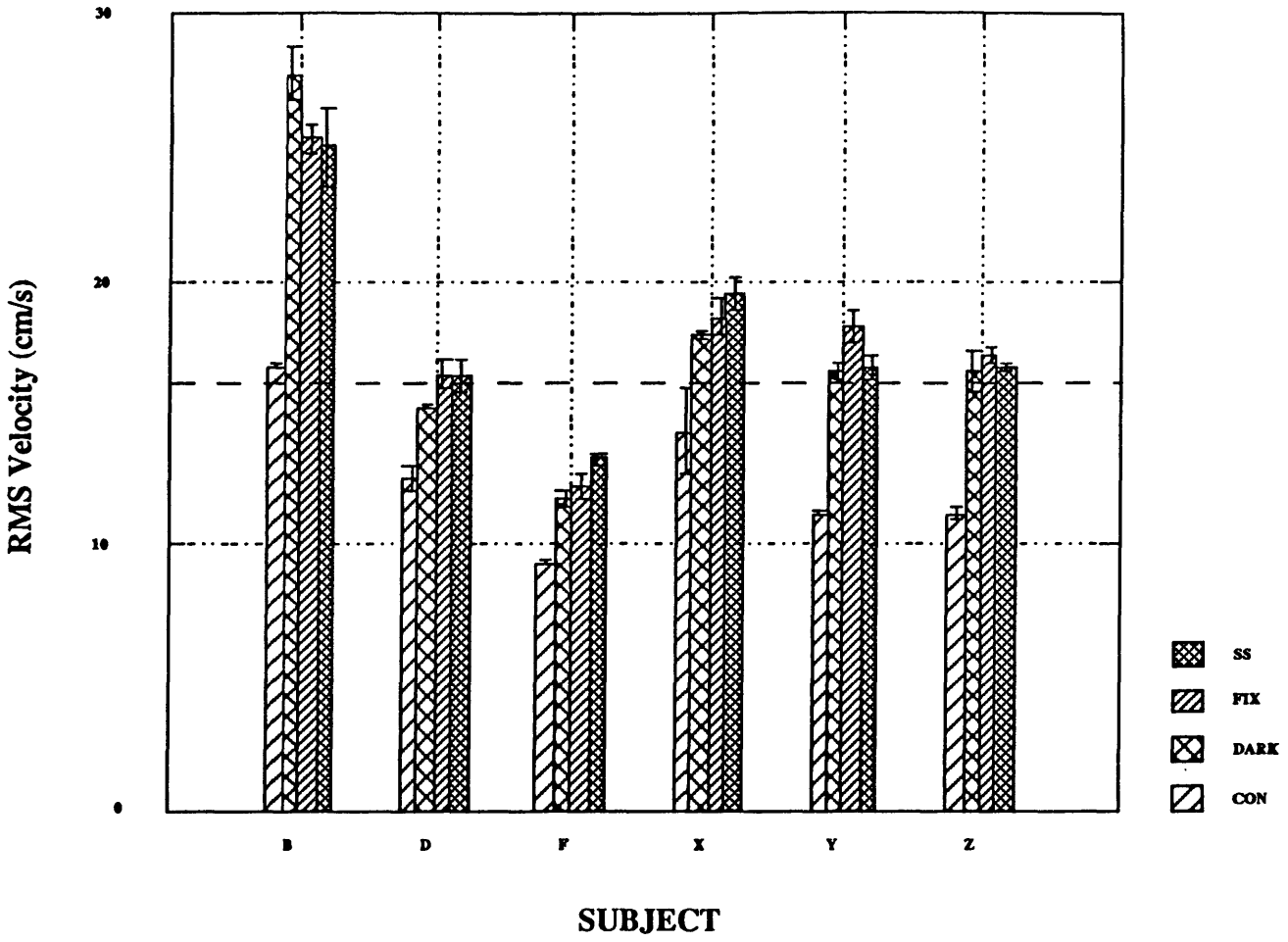


Figure 7.6: Individual subject RMS sled velocity (± se) for each of the visual field conditions. Dashed line at 16.2 cm/s represents the disturbance RMS.

consistently smaller, and subject B consistently larger. The remaining subjects showed differences which were statistically significant for at least 2 of 4 visual fields.

	CON	DARK	FIX	SS
B	16.786 ± 0.104	27.777 ± 1.035	25.370 ± 0.525	25.081 ± 1.492
D	12.496 ± 0.496	15.224 ± 0.053	16.464 ± 0.608	16.369 ± 0.636
F	9.287 ± 0.082	11.732 ± 0.327	12.158 ± 0.473	13.240 ± 0.279
X	14.279 ± 1.676	18.020 ± 0.147	18.614 ± 0.676	19.505 ± 0.653
Y	11.139 ± 0.070	16.616 ± 0.316	18.229 ± 0.643	16.700 ± 0.443
Z	11.186 ± 0.293	16.557 ± 0.791	17.158 ± 0.271	16.664 ± 0.140

Table 7.3: Subject RMS sled velocity in cm/s (± se).

	CON	DARK	FIX	SS
B	<0.001	<0.001	<0.001	<0.001
D	<0.001	<0.001	0.535	0.211
F	<0.001	<0.001	<0.001	<0.001
X	0.323	<0.001	<0.001	<0.001
Y	<0.001	0.213	<0.010	0.106
Z	<0.001	0.535	<0.001	<0.001

Table 7.4: Results of χ^2 test on RMS sled velocity.

Rankings for individual subject performance for the four visual field conditions are given in table 7.5. All subjects had their smallest RMS velocities with CON, and five subjects had their second smallest with DARK. All subjects had their maximum with either FIX or SS except subject B, who had it with DARK. These results suggest that any visual field other than CON was confusing to the subject and resulted in reduced performance in the manual lateral stabilization task. To investigate whether these differences were

statistically significant, we performed an ANOVA (post-hoc Tukey test) for each subject. CON was significantly different from DARK, FIX, and SS for subjects B, F, Y, and Z, from FIX and SS for subject D, and from SS for subject X. No other significant differences were seen.

	CON	DARK	FIX	SS
B	4	1	2	3
D	4	3	1	2
F	4	3	2	1
X	4	3	2	1
Y	4	3	1	2
Z	4	3	1	2

Table 7.5: Rankings of RMS sled velocity by test condition. (1=maximum, 4=minimum).

7.1.3. Population Results

Apparent population trends in subject nulling proficiency are presented in this section.

7.1.3.1. Mean Sled Velocity

Population results for mean sled velocity will not be presented for the following reasons:

1. Mean sled velocity was a poor measure of individual subject performance.
2. No apparent individual subject trends were seen to suggest the existence of interesting population trends.
3. Large differences in variance preclude pooling the data.

7.1.3.2. RMS Sled Velocity

Subject rankings for RMS sled velocity for a given visual field condition are tabulated in table 7.6. Inspection of this table shows that subject F consistently had the

smallest RMS velocity, subjects B and X the largest, and subject Z was consistently ranked fourth. Subjects Y and D were ranked third and fifth, respectively, for all visual fields except CON, where they were ranked fifth and third. To investigate whether these differences were statistically significant, we performed an ANOVA (post-hoc Tukey test) for each visual field (pooling the subject data). The results of the analysis are summarized in table 7.7. The intersection of two subjects gives the fields for which there was a significant difference in RMS velocity, with *all* indicating that a significant difference was found with all visual fields for that subject pair and *none* indicating that no significant difference was found for any visual field for that subject pair. All significant differences were at least 95% confident. In summary, only subjects B and F were statistically different from other subjects for any of the visual fields. This is not surprising, since subject B was the least proficient in the nulling task, and subject F was the most.

	B	D	F	X	Y	Z
CON	1	3	6	2	5	4
DARK	1	5	6	2	3	4
FIX	1	5	6	2	3	4
SS	1	5	6	2	3	4

Table 7.6: Rankings for subject RMS sled velocity (1=maximum, 6=minimum).

Since interesting individual subject trends were apparent in RMS sled velocity, we decided to look at population trends. The ability of the subject to reduce the sled RMS velocity through joystick compensation was affected by the visual fields. The results are tabulated in table 7.8, or in graphical form for easy comparison in figure 7.7. To investigate which of these RMS velocities were significantly different from the disturbance RMS velocity of 16.16 cm/s, we performed a χ^2 test. The pooled RMS velocity data for CON, DARK, FIX, and SS were statistically different from the disturbance ($p < 0.001$).

FIX had the largest RMS velocity, followed by SS, DARK, and CON. To investigate whether these differences were statistically significant, we performed an ANOVA (post-hoc Tukey test) on the pooled data for each visual field, the results of which are tabulated in table 7.9. CON was indeed significantly different from DARK, FIX, and SS, but no significant differences were seen for any other visual field pairs. This is not surprising, since DARK, FIX, and SS pooled RMS positions were very close. All significant differences were at least 95% confident.

	B	D	F	X	Y
B					
D	*all*				
F	*all*	DARK, FIX			
X	DARK, FIX, SS	*none*	*all*		
Y	*all*	*none*	DARK, FIX, SS	*none*	
Z	*all*	*none*	DARK, FIX, SS	*none*	*none*

Table 7.7: ANOVA results comparing subject RMS sled velocities for a given visual field.

CON	DARK	FIX	SS
12.529 ± 0.765	17.654 ± 1.496	18.010 ± 1.192	17.926 ± 0.813

Table 7.8: Population RMS sled velocity in cm/s (± se).

7.2. Frequency Domain Analysis of Subject Response

By use of the Fast Fourier Transform (FFT), we can investigate subject performance in the manual roll stabilization task using the subject's frequency response characteristics. In the following sections, we present the data using amplitude spectra,

scalar performance measures (SPM), visual response measures (VRM), and operator describing function analysis (see chapter 4 for a derivation of these measures).

	CON	DARK	FIX	SS
CON	1.000			
DARK	< 0.05	1.000		
FIX	< 0.01	0.997	1.000	
SS	< 0.01	0.998	1.000	1.000

Table 7.9: ANOVA results for RMS sled velocity (pooled).

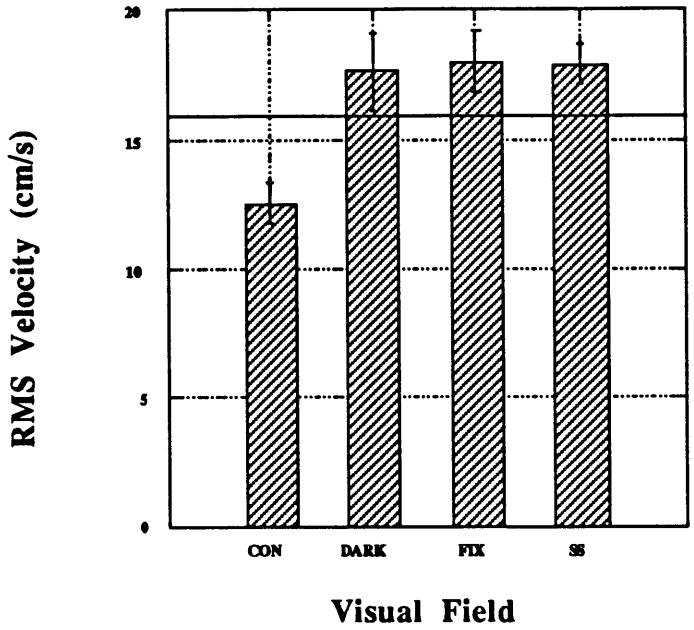


Figure 7.7: Population RMS sled velocity (± se) for each of the visual field conditions. Solid line at 16.2 cm/s represents the disturbance RMS.

7.2.1. Representative Frequency Domain Results

Figure 7.8 shows a set of mean velocity amplitude spectra for subject F, illustrating the improvement due to subject nulling for each of the visual fields. Nulling is best at low frequencies (an amplitude of zero cm/s would be perfect), and tends to worsens with increasing frequency. This general trend is seen for all the subjects who took part in the

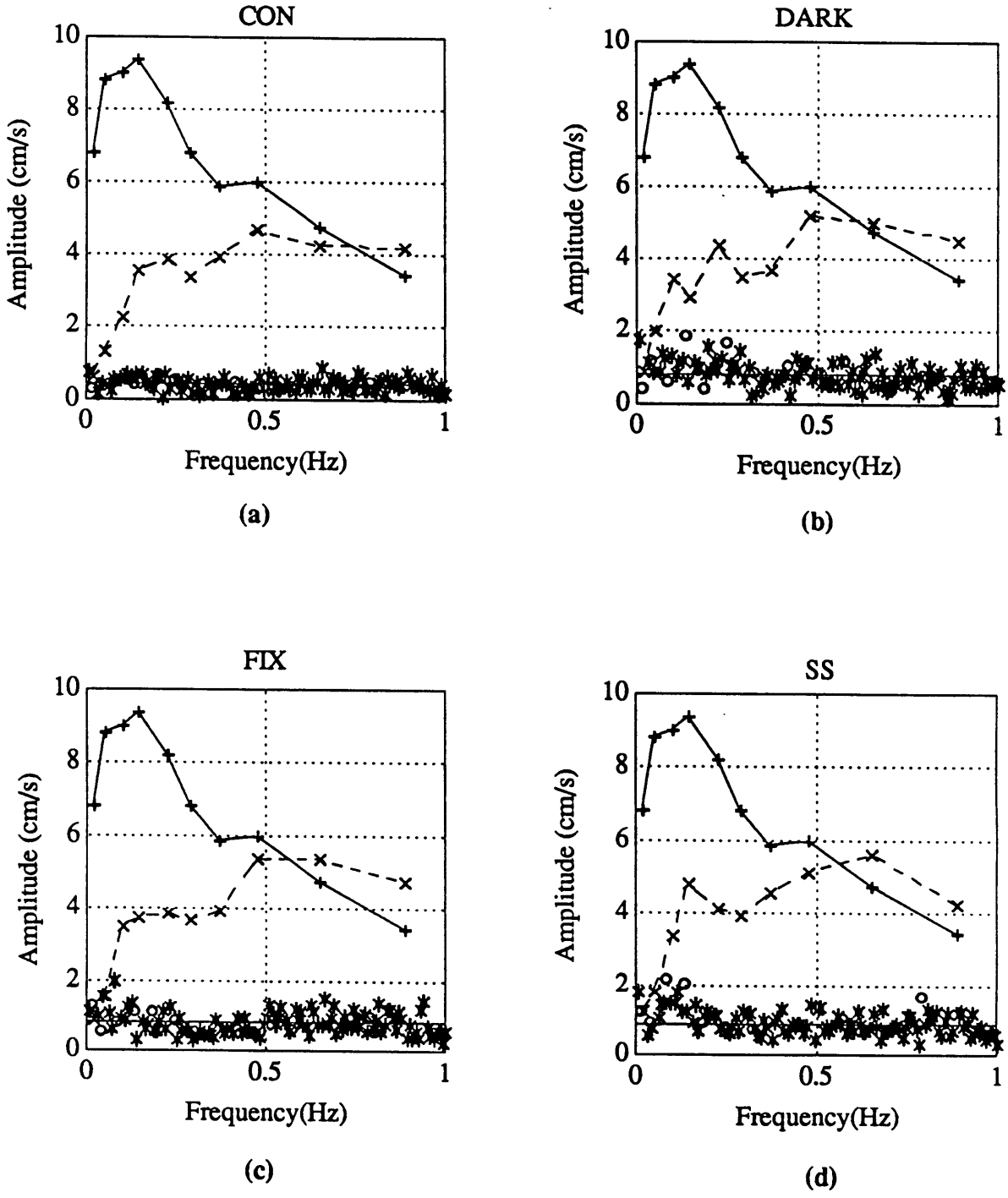


Figure 7.8 : Subject F mean velocity amplitude spectra. +'s are amplitudes if subject did no nulling, X's are at the vestibular frequencies, O's are at the visual frequencies, and *'s are subject remant. Area between curves represents change due to nulling.

experiment. Figure 7.9 shows a set of mean operator describing functions and associated fits for subject F for each of the visual fields used in the experiment. The high gain at low frequencies decays with increasing frequency and appears to level off at high frequencies; phase remains nearly constant.

7.2.2. Individual Subject Results

This section presents the results for individual subjects for SPM, VRM, and operator describing functions.

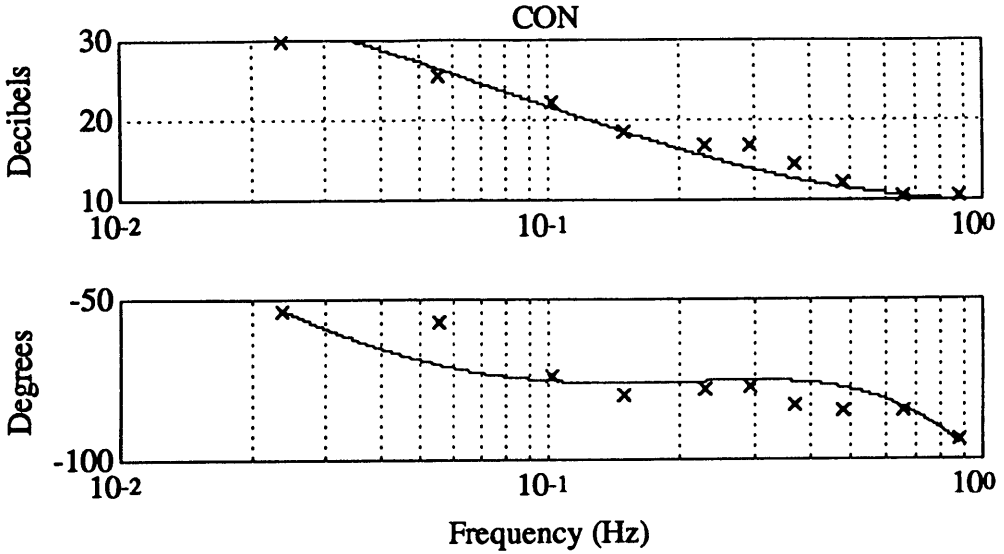
7.2.2.1. SPM

The SPM results for each individual subject and visual field condition are summarized in table 7.10, and in graphical form for easy comparison in figure 7.10. SPM rankings for individual subject performance for the four visual field conditions are given in table 7.11. All subjects had their largest SPM (and thus best nulling proficiency) with CON, and five had their second largest with DARK. The smallest SPMs were seen with FIX and SS.

	CON	DARK	FIX	SS
B	0.417 ± 0.087	0.178 ± 0.064	0.143 ± 0.022	0.161 ± 0.051
D	0.450 ± 0.014	0.201 ± 0.011	0.177 ± 0.053	0.175 ± 0.028
F	0.534 ± 0.007	0.484 ± 0.041	0.467 ± 0.011	0.435 ± 0.035
X	0.407 ± 0.003	0.296 ± 0.084	0.266 ± 0.020	0.231 ± 0.024
Y	0.466 ± 0.001	0.180 ± 0.044	0.111 ± 0.009	0.220 ± 0.020
Z	0.500 ± 0.013	0.130 ± 0.001	0.029 ± 0.005	0.087 ± 0.015

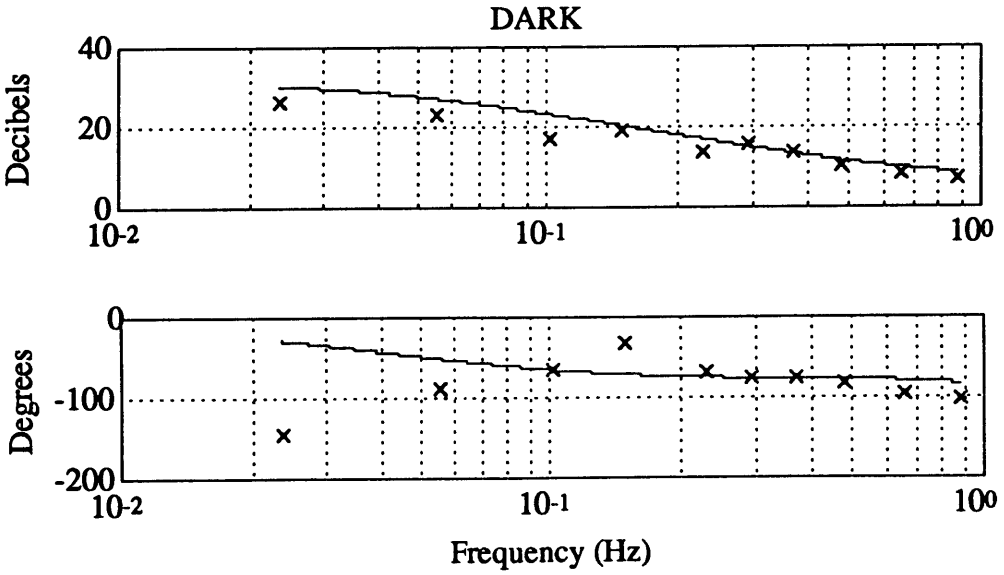
Table 7.10: Subject mean SPM (± se) for each visual field.

To investigate whether these differences were statistically significant, we performed an ANOVA (post-hoc Tukey test) for each subject. No significant differences were seen for subjects B, F, or X. CON was significantly different from DARK, FIX, and SS for



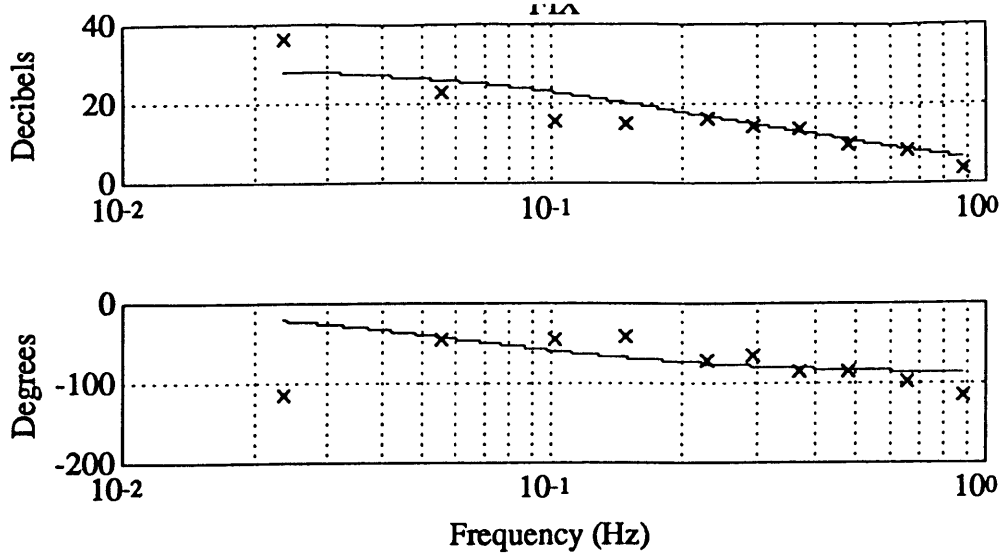
(a)

$$v(s) \rightarrow \frac{72.037(0.384s+1)}{(9.522s+1)} e^{-0.222s} \rightarrow -\lambda(s)$$

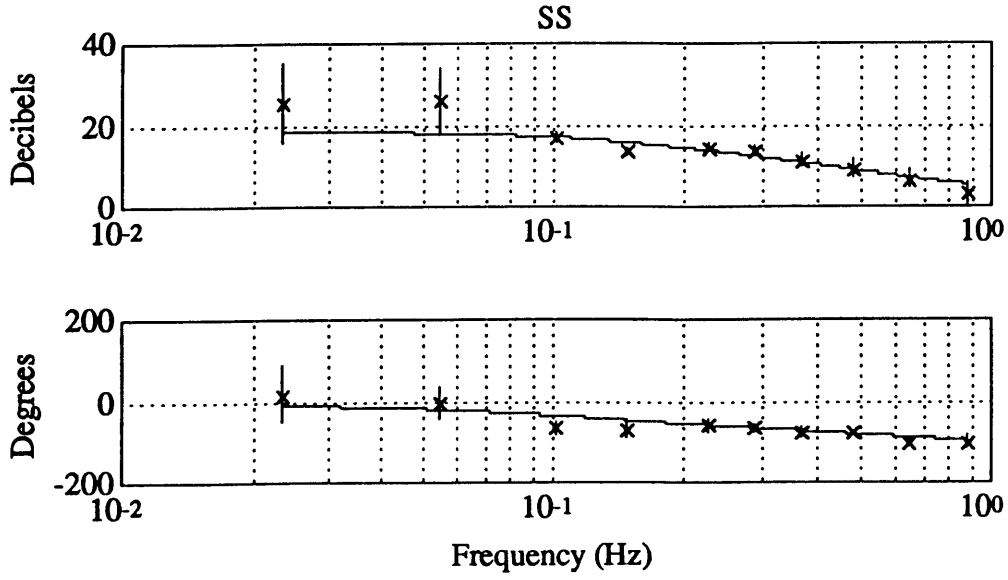


(b)

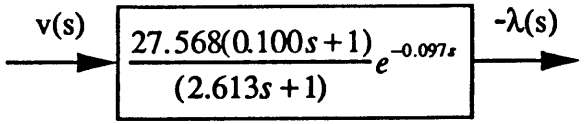
$$v(s) \rightarrow \frac{37.058(0.209s+1)}{(3.708s+1)} e^{-0.138s} \rightarrow -\lambda(s)$$



(c)



(d)



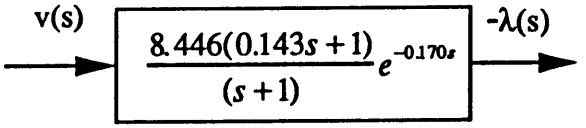


Figure 7.9 : Subject F mean operator describing function for each visual field. Boxes contain transfer function fit shown by solid line. Error bars for figure (d) are sd.

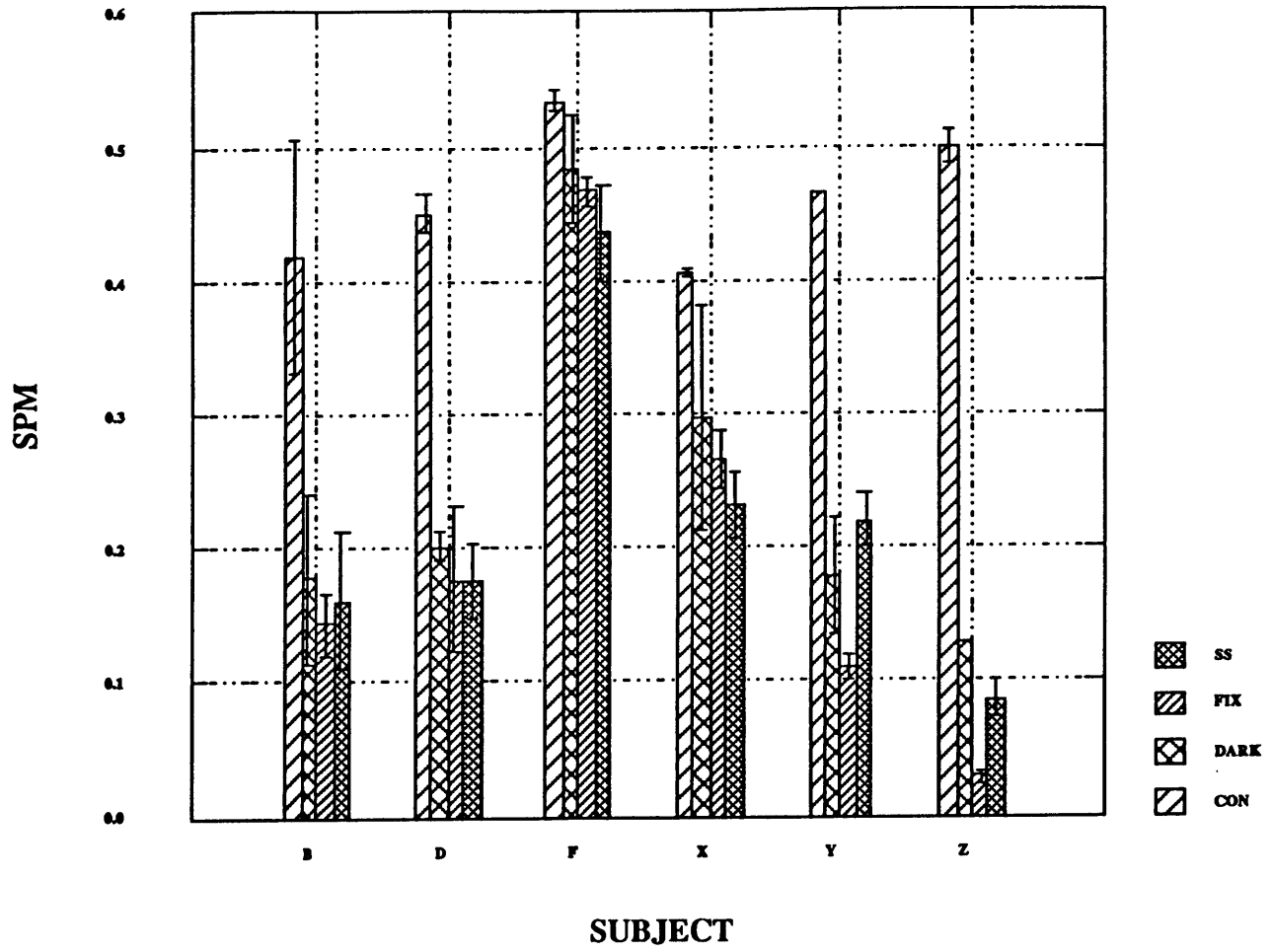


Figure 7.10: Individual subject SPM (± se) for each of the visual field conditions.

subjects D, Y, and Z, and subject Z had a significant difference between FIX and DARK. No other significant differences were seen ($p > 0.05$).

	CON	DARK	FIX	SS
B	1	2	4	3
D	1	2	3	4
F	1	2	3	4
X	1	2	3	4
Y	1	3	4	2
Z	1	2	4	3

Table 7.11: Rankings for visual field effect on SPM (1=maximum, 4=minimum).

7.2.2.2. VRM

The VRM results for each individual subject and visual field condition are summarized in table 7.12, and in graphical form for easy comparison in figure 7.11. VRM rankings for the four visual field conditions are given in table 7.13. Again to investigate the statistical significance of these differences, we performed an ANOVA (post-hoc Tukey test) for each subject. No significant differences were seen for subject D. CON was significantly different from DARK for subject B, X, Y, and Z. Subject F, X, Y, and Z had a significant difference between SS and CON, and subjects X, Y, and Z had a significant difference between FIX and CON. No other significant differences were seen ($p > 0.05$). That the SS VRM was not significantly larger than for the other visual fields suggests that the subjects were not responding to the pseudo-random visual stimulus. However, since the VRM (and thus the subject remnant) is large for CON, FIX, and DARK, it is would require a large visual effect in order to see the change in the data. Thus without further experimentation, we cannot reject the possibility that the subject had a slightly enhanced response at the visual disturbance frequencies with SS.

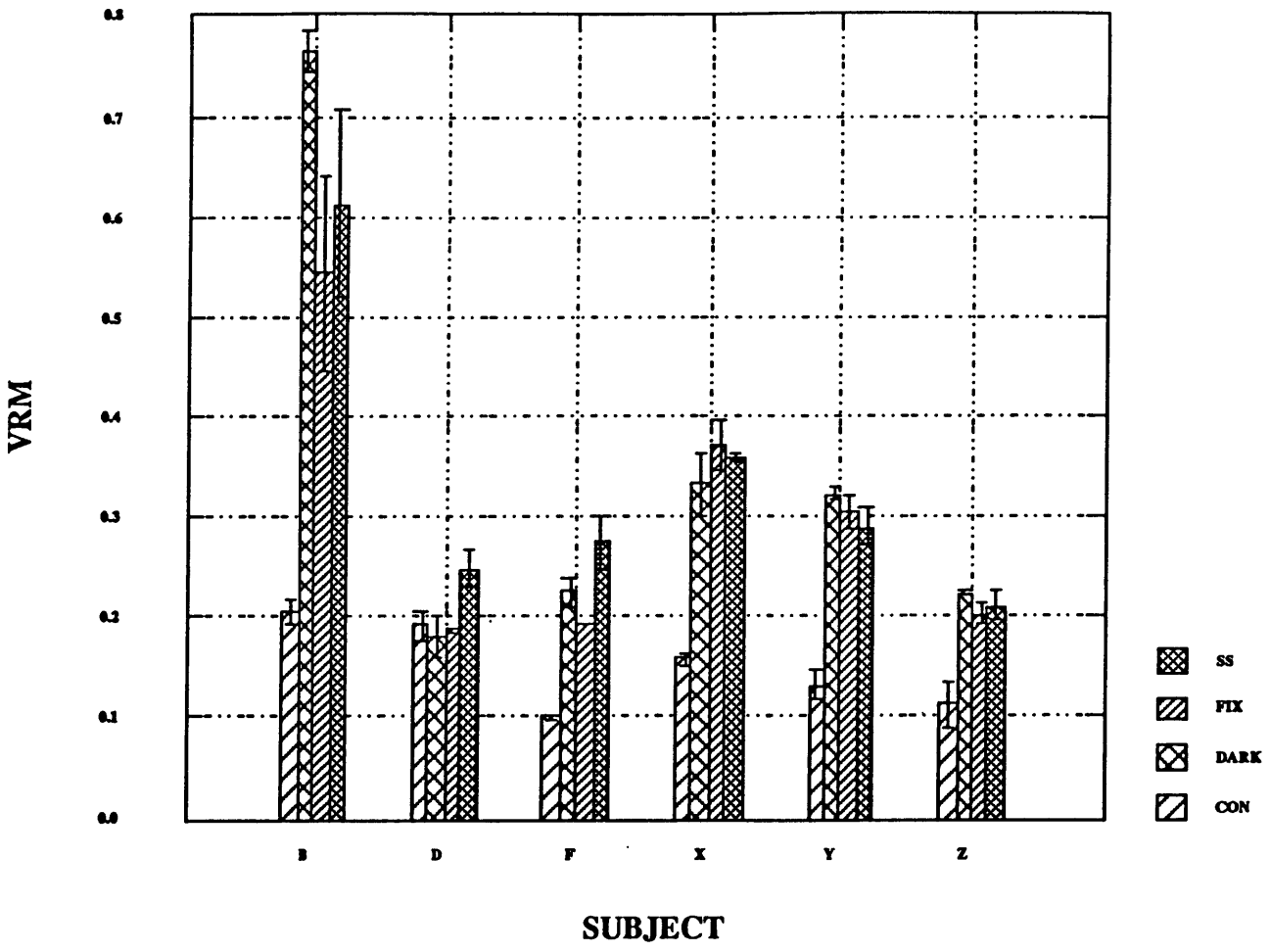


Figure 7.11: Individual subject VRM (± se) for each of the visual field conditions.

	CON	DARK	FIX	SS
B	0.206 ± 0.014	0.766 ± 0.020	0.544 ± 0.098	0.613 ± 0.093
D	0.191 ± 0.013	0.182 ± 0.021	0.187 ± 0.003	0.248 ± 0.018
F	0.101 ± 0.002	0.225 ± 0.016	0.191 ± 0.000	0.275 ± 0.027
X	0.159 ± 0.006	0.332 ± 0.031	0.372 ± 0.025	0.359 ± 0.003
Y	0.131 ± 0.015	0.323 ± 0.006	0.304 ± 0.017	0.290 ± 0.019
Z	0.112 ± 0.021	0.223 ± 0.002	0.201 ± 0.011	0.211 ± 0.015

Table 7.12: Subject mean VRM (± se) for each visual field.

	SS	DARK	FIX	CON
B	2	1	3	4
D	1	4	3	2
F	1	2	3	4
X	2	3	1	4
Y	3	1	2	4
Z	2	1	3	4

Table 7.13: Rankings for visual field effect on VRM (1=maximum, 4=minimum).

7.2.2.3. Operator Describing Functions

Transfer function fits were made to each of the individual operator describing functions using the method developed in chapter four. Table 7.14 specifies the constraints on the fit parameters and the initial value. In the interest of space, plots of these fits will not be shown for each subject. Instead the transfer function fit parameters are given in table 7.15. Inspection of this table shows that the best fits (smallest parameter variances) were obtained for the CON visual field for all subjects. Large variances in the parameters

were seen for the subjects for DARK, FIX, and SS, showing that fit quality for these fields was poor. Better fits were seen for the pooled operator describing functions, and are presented in section 7.2.3.4.

	K	τ_1	τ_2	τ_d
initial value	10.00	10.00	100.00	0.01
maximum	100.00	100.00	200.00	1.00
minimum	1.00	0.10	1.00	0.00

Table 7.14: Transfer function fit parameter constraints and initial values.

7.2.3. Population Results

Apparent population trends in the subject frequency response are presented in this section using amplitude spectra, SPM, VRM, operator describing functions, and open-loop transfer functions.

7.2.3.1. Amplitude Spectra

Figure 7.12 shows the population mean amplitude spectra of the sled velocity (with subject compensation) for each of the visual field conditions used in the experiment. Inspection of this figure shows that, for all visual fields, subjects responded to the frequencies in the vestibular disturbance. The most efficient nulling was seen at low frequencies, while at high frequencies the subject made little improvement, or in some cases even added energy.

7.2.3.2. SPM

Subject rankings for SPM for a given visual field condition are tabulated in table 7.16. Inspection of this table shows that subject F had the largest SPMs (and thus best nulling proficiency) for all visual fields, and subject Z had the worst for all visual fields except CON, where he had the second best. Subject X had the second best nulling proficiency for all visual fields except CON, where he had the worst. Subject B had the fifth worst proficiency for all fields except FIX, where she had the fourth worst. Rankings

Subject	CON				DARK			
	K	τ_1	τ_2	τ_d	K	τ_1	τ_2	τ_d
B	61.41 ± 10.60	0.185 ± 0.009	6.362 ± 1.135	0.359 ± 0.006	100.0 ± 0.0 [2]	0.323 ± 0.223	15.547 ± 3.322	0.390 ± 0.115
D	3.661 ± 1.245	43.083 ± 0.858	50.98 ± 18.95	0.340 ± 0.005	3.078 ± 1.023	9.174 ± 2.611	31.096 ± 1.993	0.384 ± 0.004
F	71.46 ± 5.53	0.379 ± 0.032	9.356 ± 0.843	0.221 ± 0.010	46.01 ± 23.94	0.197 ± 0.094	4.814 ± 2.855	0.132 ± 0.034
X	100.0 ± 0.0 [2]	0.164 ± 0.000	12.250 ± 0.027	0.294 ± 0.001	12.427 ± 0.403	0.214 ± 0.002	2.947 ± 0.045	0.223 ± 0.014
Y	100.0 ± 0.0 [2]	0.536 ± 0.002	17.619 ± 0.032	0.219 ± 0.000	16.07 ± 11.09	2.851 ± 1.426	38.71 ± 34.01	0.495 ± 0.038
Z	78.4 ± 21.7 [1]	0.845 ± 0.063	17.229 ± 5.541	0.331 ± 0.005	6.341 ± 1.441	40.284 ± 9.426	200.0 ± 0.0 [2]	0.450 ± 0.004

Table 7.15a: Operator describing function fit parameters (± se) for individual subjects for CON and DARK. Superscript indicates number of parameters included in calculation of mean that were at constraint.

Subject	FIX				SS			
	K	τ_1	τ_2	τ_d	K	τ_1	τ_2	τ_d
B	50.52 ± 17.02	9.35 ± 9.25 [1]	106.8 ± 93.3 [1]	0.493 ± 0.227	8.869 ± 2.080	0.20 ± 0.05 [1]	2.64 ± 0.93 [1]	0.234 ± 0.042
D	2.24 ± 1.24 [1]	3.126 ± 1.004	9.154 ± 7.657	0.218 ± 0.033	13.34 ± 11.57	0.927 ± 0.660	50.8 ± 49.8 [4]	0.06 ± 0.04 [1]
F	27.814 ± 0.866	0.10 ± 0.00 [2]	2.619 ± 0.040	0.096 ± 0.004	31.6 ± 22.8 [1]	0.24 ± 0.07 [2]	5.47 ± 4.42 [2]	0.206 ± 0.020
X	96.20 ± 3.80 [1]	0.760 ± 0.454	32.62 ± 12.27	0.391 ± 0.125	8.195 ± 0.336	0.209 ± 0.062	1.803 ± 0.117	0.394 ± 0.023
Y	12.528 ± 3.228	0.539 ± 0.134	8.344 ± 2.926	0.379 ± 0.022	6.157 ± 1.079	0.525 ± 0.187	2.79 ± 0.88 [1]	0.254 ± 0.048
Z	25.039 ± 1.736	6.270 ± 0.464	200.0 ± 0.0 [2]	0.555 ± 0.004	16.829 ± 4.426	12.135 ± 6.162	169.9 ± 30.1 [3]	0.490 ± 0.047

Table 7.15b: Operator describing function fit parameters (± se) for individual subjects for FIX and SS. Superscript indicates number of parameters included in calculation of parameter mean that were at a constraint.

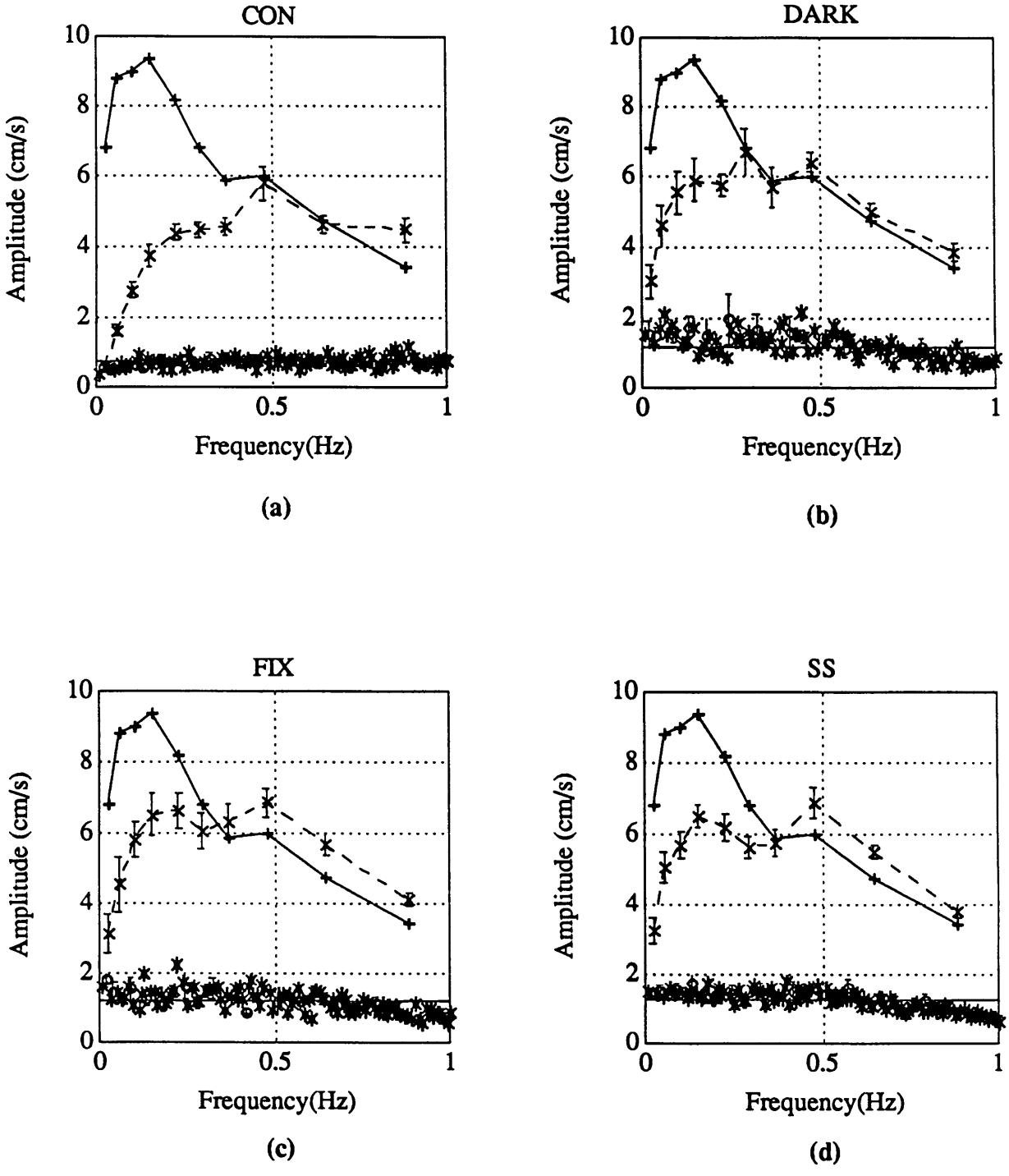


Figure 7.12: Population mean velocity amplitude spectra. +'s are amplitudes if subject did no nulling, X's are at the vestibular frequencies, O's are at the visual frequencies, and *'s are subject remant. Area between curves represents change due to nulling.

for the other subjects were mixed, with no apparent trends. To investigate whether these differences were statistically significant, we performed an ANOVA (post-hoc Tukey test) for each subject. The results of the analysis are summarized in table 7.17. An intersection of two subjects gives the fields for which there was a significant difference in SPM, with *none* indicating that no significant difference was found for any visual field for that subject pair ($p > 0.05$).

	B	D	F	X	Y	Z
CON	5	4	1	6	3	2
DARK	5	3	1	2	4	6
FIX	4	3	1	2	5	6
SS	5	4	1	2	3	6

Table 7.16: Rankings for subject SPM (1=maximum, 6=minimum).

	B	D	F	X	Y
B					
D	*none*				
F	DARK, FIX, SS	DARK, FIX, SS			
X	*none*	*none*	FIX, SS		
Y	*none*	*none*	DARK, FIX, SS	FIX	
Z	*none*	FIX	FIX, SS	FIX, SS	*none*

Table 7.17: ANOVA results comparing subject SPM for each visual field.

Since interesting SPM trends were seen for the individual subjects, we decided to look at population trends. Subject performance, as measured by the SPM, was affected by the visual fields. The SPM order from smallest to largest (and thus increasing nulling proficiency), was FIX, SS, DARK, CON. These results are given in table 7.18, or in

graphical form for easy comparison in figure 7.13a. To investigate whether this trend was significant, we performed an ANOVA (post-hoc Tukey test) on the pooled data, the results of which are given in table 7.19. In summary, CON was significantly different from DARK, FIX, and SS, but no other significant differences were seen ($p > 0.05$).

CON	DARK	FIX	SS
0.462 ± 0.017	0.245 ± 0.039	0.199 ± 0.043	0.218 ± 0.025

Table 7.18: Population SPM (± se).

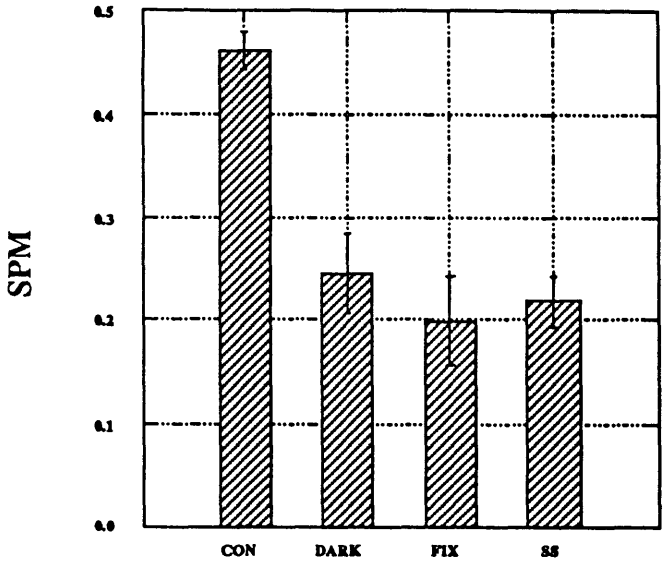
	CON	DARK	FIX	SS
CON	1.000			
DARK	< 0.001	1.000		
FIX	< 0.001	0.793	1.000	
SS	< 0.001	0.927	0.969	1.000

Table 7.19: ANOVA results for SPM (pooled).

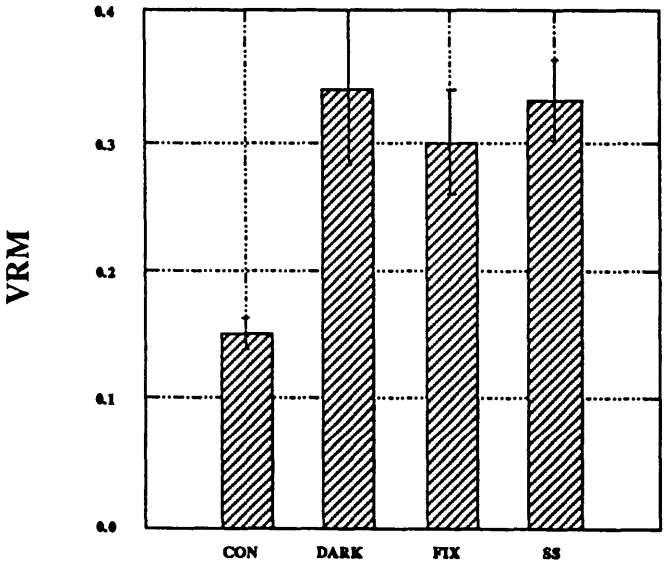
7.2.3.3. VRM

Subject rankings for VRM for a given visual field condition are tabulated in table 7.20. Inspection of this table shows that subject B had the largest remnant for the three visual fields. To investigate whether these differences were statistically significant, we performed an ANOVA (post-hoc Tukey test) for each subject. The results of the analysis are summarized in table 7.21. Again, an intersection of two subjects gives the fields for which there was a significant difference in VRM, with *none* indicating that no significant difference was found for any visual field for that subject pair ($p > 0.05$).

Turning to population trends, the VRM order from smallest to largest (and thus increasing response at the visual disturbance frequencies) for the subject population was



(a)



(b)

Figure 7.13: Population mean (+/- se) SPM (a) and VRM (b).

CON, FIX, SS, and DARK. These results are given in table 7.22, or in graphical form for easy comparison in figure 7.13b. Again, we performed an ANOVA (post-hoc Tukey test) on the pooled data, the results of which are given in table 7.23. In summary, CON was significantly different from DARK and SS, but no other significant differences were seen. That SS was not significantly different from DARK and FIX shows that the subject was not responding to the pseudo-random visual stimulus. This is not surprising, since none of the individual subjects showed a significant response at the visual disturbance frequencies.

	B	D	F	X	Y	Z
CON	1	2	6	3	4	5
DARK	1	6	4	2	3	5
FIX	1	6	5	2	3	4
SS	1	5	4	2	3	6

Table 7.20: Subject rankings for VRM (1=maximum, 6=minimum).

	B	D	F	X	Y
B					
D	DARK, FIX, SS				
F	*all*	CON			
X	DARK, SS	DARK	DARK		
Y	DARK, FIX, SS	DARK	*none*	*none*	
Z	CON, FIX, SS	CON	*none*	DARK	*none*

Table 7.21: ANOVA results comparing subject VRM for each visual field.

7.2.3.4. Operator Describing Functions

Figures 7.14 - 7.17 are Bode plots of the population mean operator describing functions for the six different visual fields used in this experiment. The same fit parameter

constraints and initial values given in table 7.14 were used for the fits for the pooled operator describing functions. A transfer function fit to the data is shown by the solid line in each figure (see section 4.4.2.2 in chapter 4 for a discussion of the fitting process); the parameters for each of the fits are given in table 7.24. Fit quality was very good for all visual fields.

CON	DARK	FIX	SS
0.150 ± 0.012	0.342 ± 0.060	0.300 ± 0.041	0.333 ± 0.032

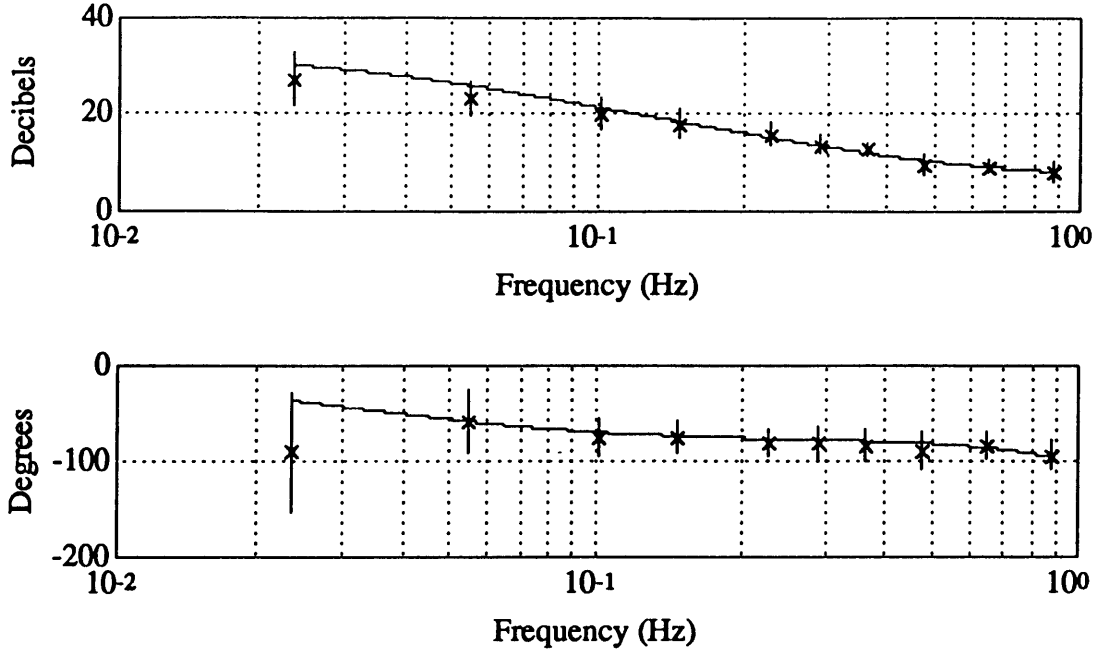
Table 7.22: Population VRM (± se).

	CON	DARK	FIX	SS
CON	1.000			
DARK	< 0.05	1.000		
FIX	0.079	0.903	1.000	
SS	< 0.01	0.998	0.927	1.000

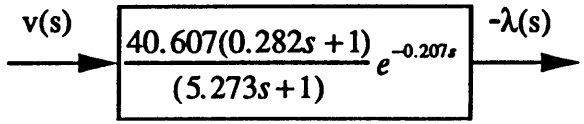
Table 7.23: ANOVA results for VRM (pooled).

	CON	DARK	FIX	SS
K	40.607	100.000 *	7.086	5.303
τ_1	0.282	0.615	0.165	0.100 *
τ_2	5.273	43.997	1.871	1.424
τ_d	0.207	0.310	0.232	0.124

Table 7.24: Summary of mean operator describing function fit parameters for each visual field. Superscript * indicates a parameter constraint.

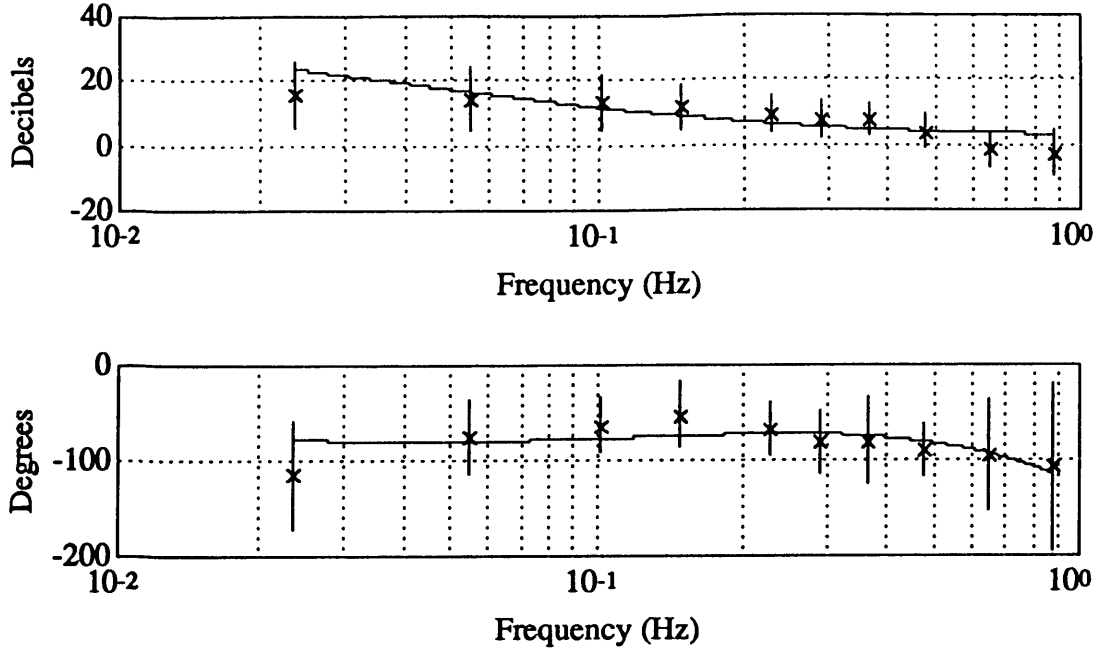


(a)

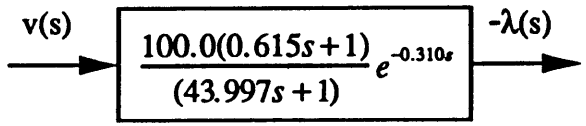


(b)

Figure 7.14: Population mean (± sd) operator describing function for CON visual field (a) and transfer function fit shown by solid line (b).

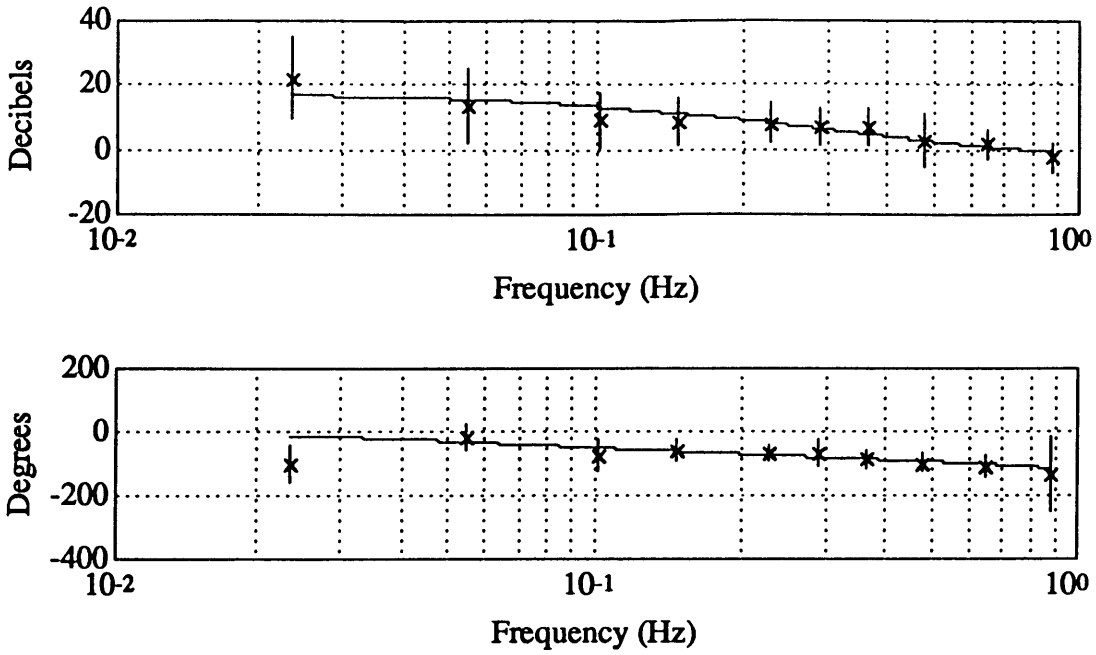


(a)

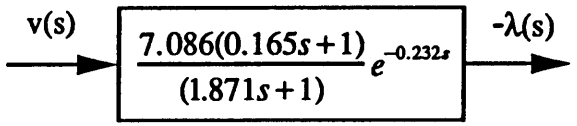


(b)

Figure 7.15: Population mean (± sd) operator describing function for DARK visual field (a) and transfer function fit shown by solid line (b).

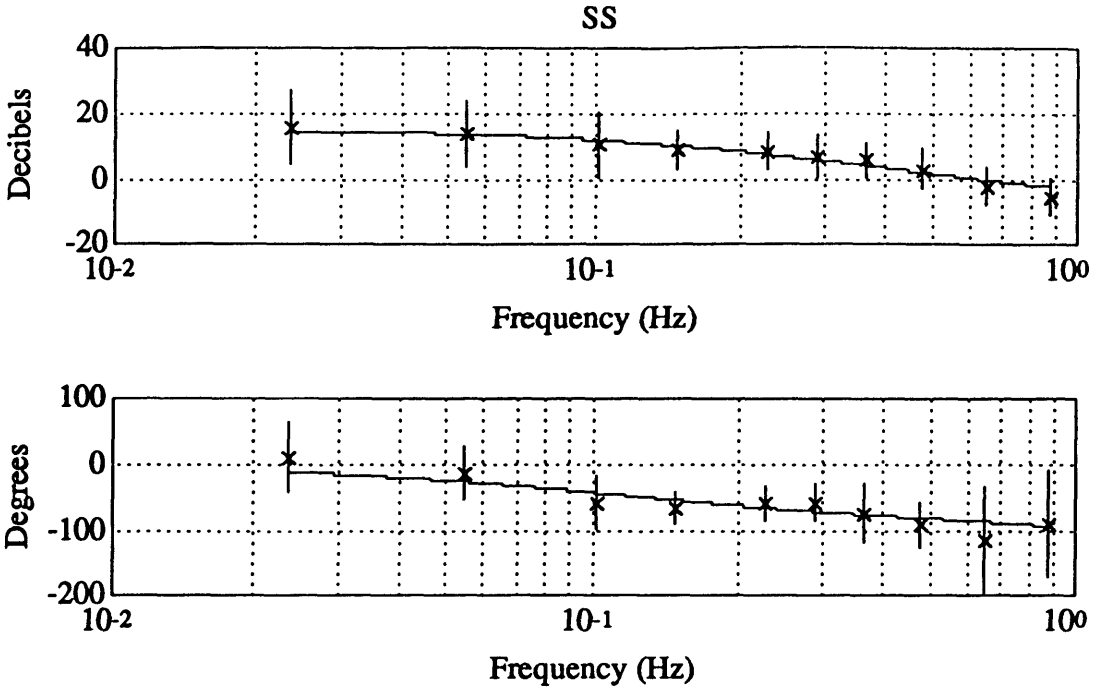


(a)

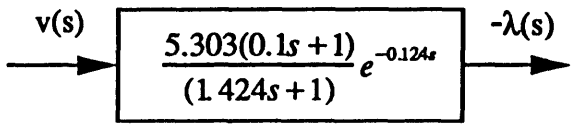


(b)

Figure 7.16: Population mean (± sd) operator describing function for FIX visual field (a) and transfer function fit shown by solid line (b).



(a)



(b)

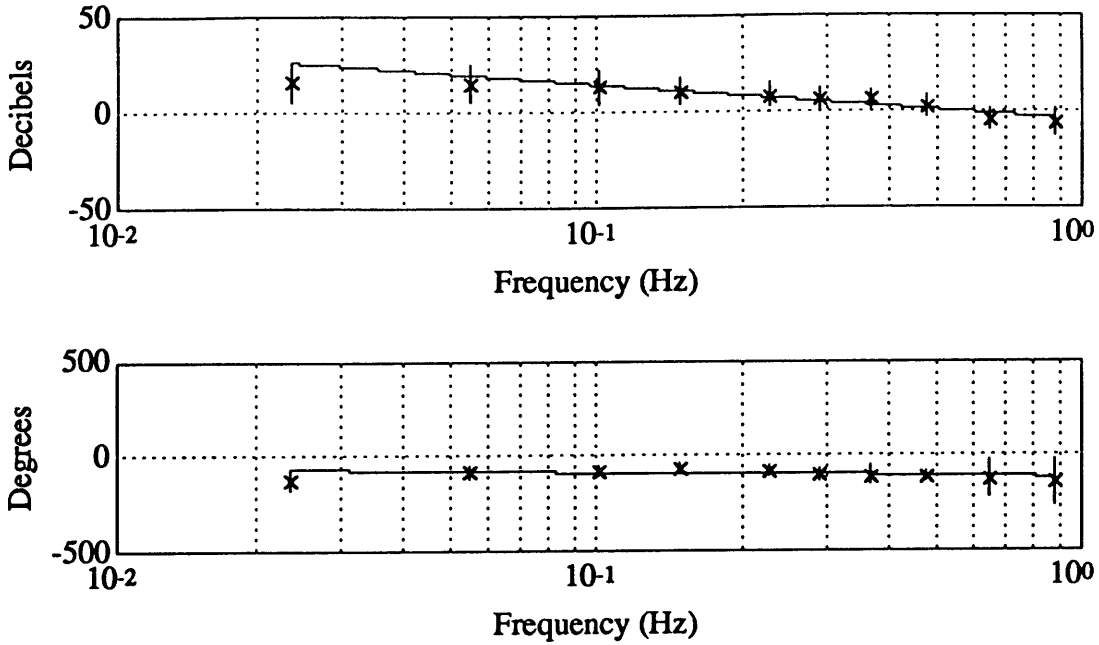
Figure 7.17: Population mean (± sd) operator describing function for SS visual field (a) and transfer function fit shown by solid line (b).

7.2.3.5. Open-Loop Transfer Functions

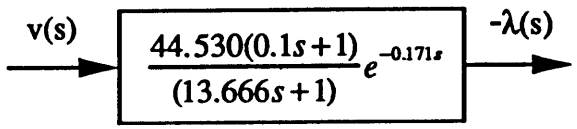
Figures 7.18 - 7.21 are Bode plots of the population mean open-loop transfer functions (product of the operator describing function and Link dynamics) for the four different visual fields used in this experiment. A transfer function fit to the data is shown by the solid line in each figure (see section 4.4.2.2 in chapter 4 for a discussion of the fitting process); the parameters for each of the fits are summarized in table 7.25. As with the pooled operator describing function, the fit quality was good for all visual fields. In chapter eight, we compare the control strategy of the pooled subjects with that predicted by the McRuer Crossover Model.

	CON	DARK	FIX	SS
K	45.530	12.360	29.764	4.614
τ_1	0.100 *	0.100 *	0.562	0.100 *
τ_2	13.666	2.494	16.318	1.469
τ_d	0.171	0.289	0.488	0.226

Table 7.25: Summary of mean open-loop transfer function fit parameters for each visual field. Superscript * indicates a parameter constraint.

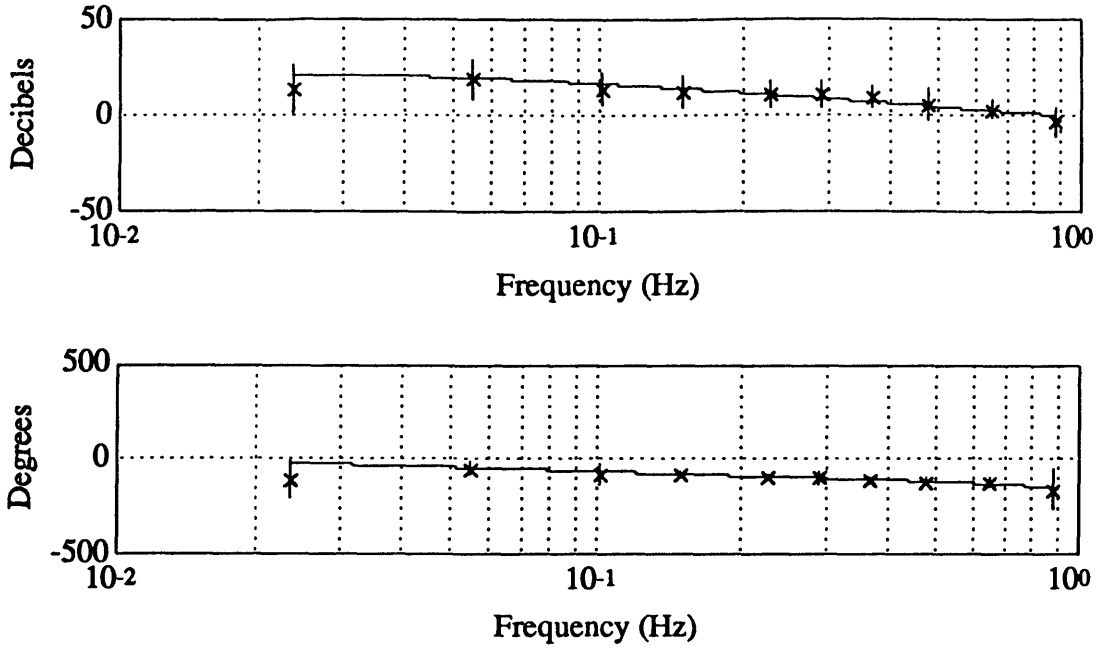


(a)

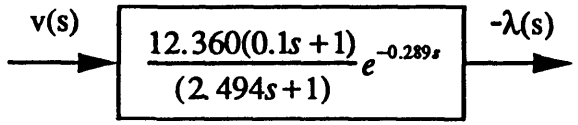


(b)

Figure 7.18: Population mean (± sd) open-loop transfer function for CON visual field (a) and transfer function fit shown by solid line (b).

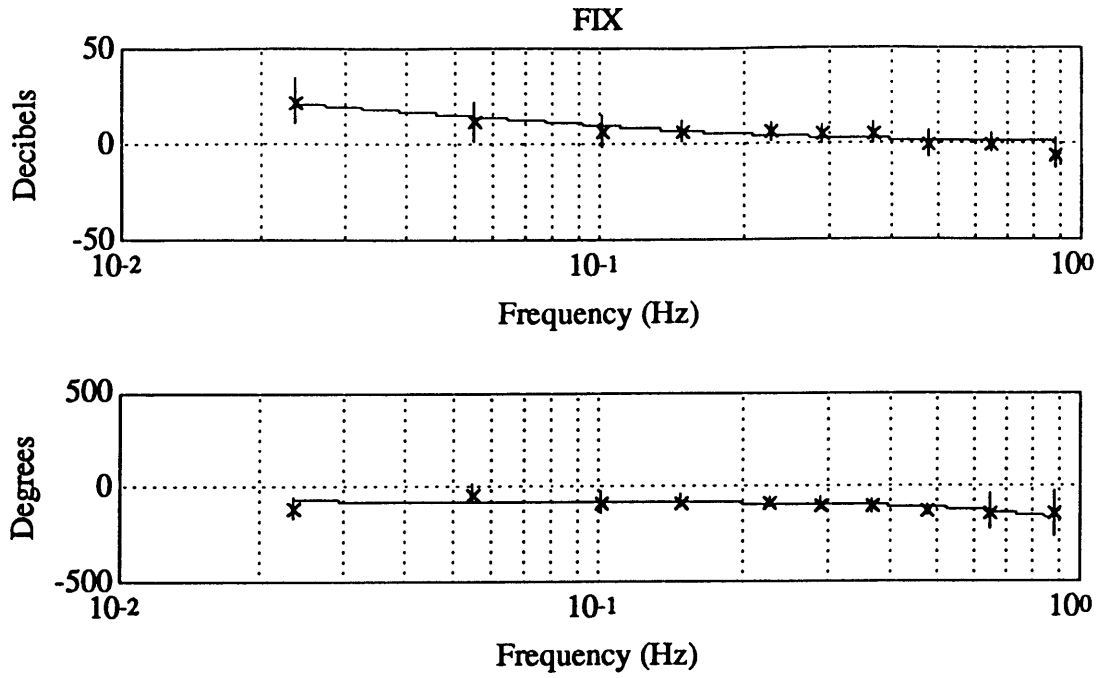


(a)

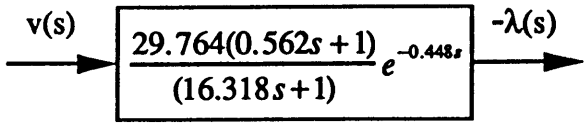


(b)

Figure 7.19: Population mean (± sd) open-loop transfer function for DARK visual field (a) and transfer function fit shown by solid line (b).

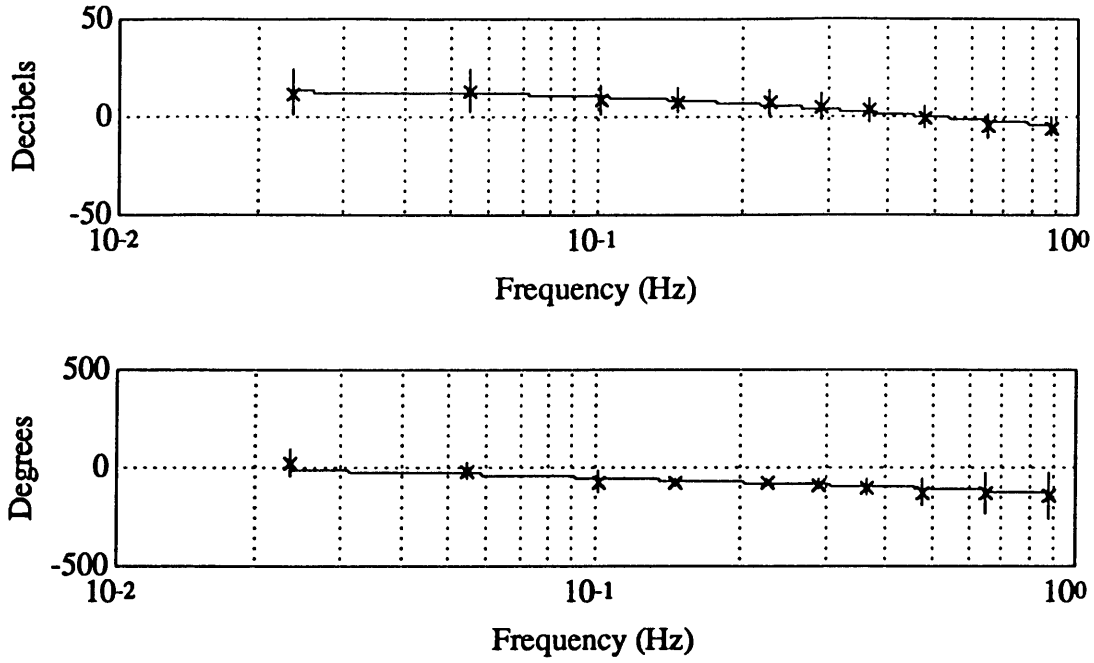


(a)

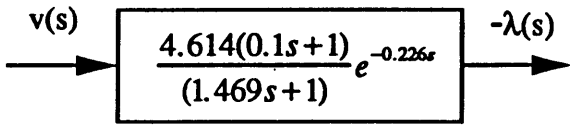


(b)

Figure 7.20: Population mean (± sd) open-loop transfer function for FIX visual field (a) and transfer function fit shown by solid line (b).



(a)



(b)

Figure 7.21: Population mean (± sd) open-loop transfer function for SS visual field (a) and transfer function fit shown by solid line (b).

PART FOUR

8. DISCUSSION OF RESULTS IN THE MANUAL ROLL AND LATERAL STABILIZATION EXPERIMENTS

8.1. Otolith-Canal-Visual Interaction

It is well known that motion cues aid human operator response in compensatory tracking tasks. The roll and lateral experiments discussed in this thesis were directed at investigating otolith-canal-visual interactions; more specifically, how the signals from these systems are combined by the nervous system to yield a wideband sensory system. The extent of otolith contributions to motion sensation was studied by Dinsdale [5]. In his thesis, he concluded that the addition of otolith stimulation to semicircular canal stimulation allowed the human operator to increase his gain and phase lead over the middle-to-high frequency range ($0.15 \text{ Hz} < \omega < 0.25 \text{ Hz}$). For the low frequency range ($\omega < 0.15 \text{ Hz}$), the human operator appears to depend on a combination of vision and otoliths. In the high frequency range ($\omega > 0.25 \text{ Hz}$), dependence appears to be primarily on the semicircular canals. In the roll position nulling task, the combination of the strong low frequency otolith cue (the otolith organs are very sensitive to tilt) together with the semicircular canal cue in the middle-to-high and high frequency range should result in good subject roll position nulling proficiency across a wide range of frequencies. In interaural accelerations the otoliths act as very effective linear acceleration sensors. Thus, in the absence of a visual field, we would expect good velocity nulling proficiency in the low and mid-to-high frequency range. However, no complementary high frequency canal cues are available for linear accelerations, and therefore we might hypothesize that performance in this frequency range could be worse than in the manual roll stabilization experiment. In both manual roll and lateral stabilization, low frequency performance should be enhanced with confirming visual cues (CON), and reduced with non-confirming cues (DARK, FIX, SS, CV), while

high frequency performance should be relatively unaffected by the visual field. These hypotheses are discussed in the sequel.

8.2. Manual Roll Stabilization

This section discusses the population time and frequency domain results for the manual roll stabilization experiment.

8.2.1. Time Domain

This section discusses the results of the manual roll stabilization experiments for mean and RMS position, as well as an apparent limit on realizable subject performance.

8.2.1.1. Mean Trainer Position

As expected, no consistent mean position bias was seen for the CON, DARK, FIX, and SS visual fields. In the case of DARK and FIX where dependence was primarily on vestibular cues, 4 of 6 subjects (A, C, E, and F) had mean positions within approximately ±1 degree of zero. Of these subjects, only subject E's DARK mean position was significantly different from zero degrees. Similarly, in the case of CON and SS, 4 of 6 subjects (A, C, E, and F) had mean positions within ±1.3 degree of zero, and 3 of 6 subjects (A, E, and F) were within ±1 degree of zero. Certainly we would expect a nearly zero mean position for CON, since the confirming visual cues gave the subject an "absolute" reference with which to estimate (and thus correct for) his roll position error. That the mean trainer position for CON was significantly different from zero and leftward biased for all subjects was most likely a result of the subjects using the fixation LED mounting post as a vertical reference (when in fact it was tilted slightly to the right with respect to the trainer). In the case of SS, the visual disturbance was zero-mean velocity (and thus position), and thus any response by the subject to the visual field would not induce a mean position bias. Of the remaining subjects, subject B had a statistically significant leftward mean position bias for all fields, as did subject D for CON and DARK, however these subjects were also among the least proficient at the nulling task. Due to large differences in variance between individual subjects, the data were not pooled to obtain

population mean trainer positions for the four visual fields. The results for CON and FIX agree with Huang, who saw a mean position of nearly zero degrees for both visual fields, however no comparison is possible for DARK and SS, since these fields were not used in Huang's experiment [13]. That the majority of the subjects were able to maintain the trainer with a nearly zero mean position (in the absence of a non-confirming visual field) supports the dominance of the otolith cue at low frequencies.

Trials with the CV visual fields, on the other hand, had a mean position bias when compared to the other visual field conditions. This bias (in the direction of the field) was caused by roll vection. Four of 6 subjects had their most negative mean positions with CVL, and 6 of 6 subjects had their most positive mean position with CVR. These trends were preserved for the subject population. The CVR results are in agreement with Huang, who saw an approximately two degree rightward population mean position bias with CVR (referred to as CVT in his experiment). Unfortunately, no comparison of the latter result (and therefore of the apparent left/right asymmetry) is possible, since Huang performed no trials with the CVL field [13]. The fact that the visual field induced a mean position bias supports the hypothesis that vision plays a key role at low frequencies.

8.2.1.2. RMS Trainer Position

Consistent RMS position trends were seen for the six visual fields used in the manual roll stabilization experiment. Five of 6 subjects had their largest RMS positions with CV, and 4 of 6 subjects also had their second largest . This enhanced RMS was due primarily to two factors: (1) reduced nulling proficiency at low frequencies due to the disorienting nature of the visual field, and the (2) position bias induced by the subject (due to the roll vection). These results agree with Huang, who saw an RMS position of 3.5 degrees for the CVR field, compared to approximately 3 degrees for CON and FIX (his disturbance had an RMS position of 5 degrees) [13]. Since Huang did not do trials with a CVL field, no comparison of results is possible. The fact that the visual field induced a mean position bias again supports the hypothesis that vision plays a key role at low

frequencies. Four of the six subjects (A, B, C, and F) had their next largest RMS trainer position with SS, followed by DARK, FIX, and CON. Although subject D had her least RMS position with FIX rather than CON, she otherwise agreed with the trend. Subject E was relatively unaffected by the visual field. Of the six subjects, B and D had the largest RMS positions (and thus worst nulling proficiency), and only subject B ever had an RMS position larger than the disturbance (DARK and SS). That trials with the SS field had RMS positions larger than CON, DARK, and FIX supports the hypothesis that low frequency visual cues are used by the subject to complement otolith and canal information to effect a wideband sensory system. This enhanced RMS is due primarily to response of the subject to the visual disturbance (particularly the low frequency components) and thus reduced response to the vestibular disturbance. That CON should have the smallest RMS position (and thus the best nulling proficiency) is intuitive, since the confirming visual cues (especially the strong horizontal reference visible to the subject through the trainer front window) provided the subject with an "absolute" reference with which to estimate his roll position error. In the case of FIX and DARK trials, the subject was depending solely on vestibular cues which should result in a larger RMS position than CON. Finally, FIX performance was better than DARK. Although in both cases the subject was depending entirely on vestibular cues, the enhanced performance with FIX was most likely due to an interaction (without conflict) of low frequency otolith and visual cues. This agrees with the results of Huang, who saw comparable nulling proficiency with FIX and CON in his roll experiment [13]. This RMS position trend was preserved in the subject population.

8.2.1.3. Limit on Realizable Performance

The fact that mean and RMS position were not statistically different for CON, DARK, and FIX suggests the existence of an upper bound on realizable subject performance. This could be caused by a limit in the subject's accuracy in estimating zero tilt or in his capability to control the trainer. The former would cause the subject to maintain the trainer position within a small error region on either side of zero degrees, most

likely oscillating back-and-forth through zero. This oscillatory behavior could vary from trial to trial, thus altering the mean and RMS position. The latter could be caused by the inability of the subjects to turn the control wheel quickly and accurately enough for fine control, or by too much system delay between the control wheel command and the trainer response. Either of these possible causes (or a combination of the two) would limit how tightly the subject could control the system, especially at middle-to-high and high frequencies, which would place a natural upper bound on how effectively the subject could perform the manual roll stabilization task.

8.2.2. Frequency Domain

This section discusses the results seen in the manual roll stabilization experiment for the frequency response of the subjects.

8.2.2.1. Amplitude Spectra

Close inspection of figure 5.14 reveals several general trends: (1) subjects tended to undercompensate for the vestibular disturbance; (2) nulling was best at low frequencies, while at high frequencies only small reductions in the roll motion were achieved; (3) subjects did respond to the sum of sines pseudo-random visual stimulus at frequencies less than approximately 0.16 Hz; and (4) remnant response was small in all cases, but was slightly enhanced at low frequencies with CVL, CVR, and SS. Quantitative comparison of nulling proficiency with the four visual fields is discussed in section 8.2.2.3.

8.2.2.2. Operator Describing Functions

Since the quality of the fits to the individual operator describing functions were less than satisfactory (see section 5.2.2.3 of chapter 5), this section only discusses the effect of the visual field on the population mean operator describing functions. Figure 8.1 shows asymptotic approximations to the frequency domain transfer function fits to the population mean operator describing functions for each of the visual fields using the values for K (renamed K_{low} to distiniguish it from the high frequency gain K_{high}), τ_1, and τ_2 given in table 5.25 of chapter 5. Table 8.1 summarizes the values for low frequency gain (K_{low}),

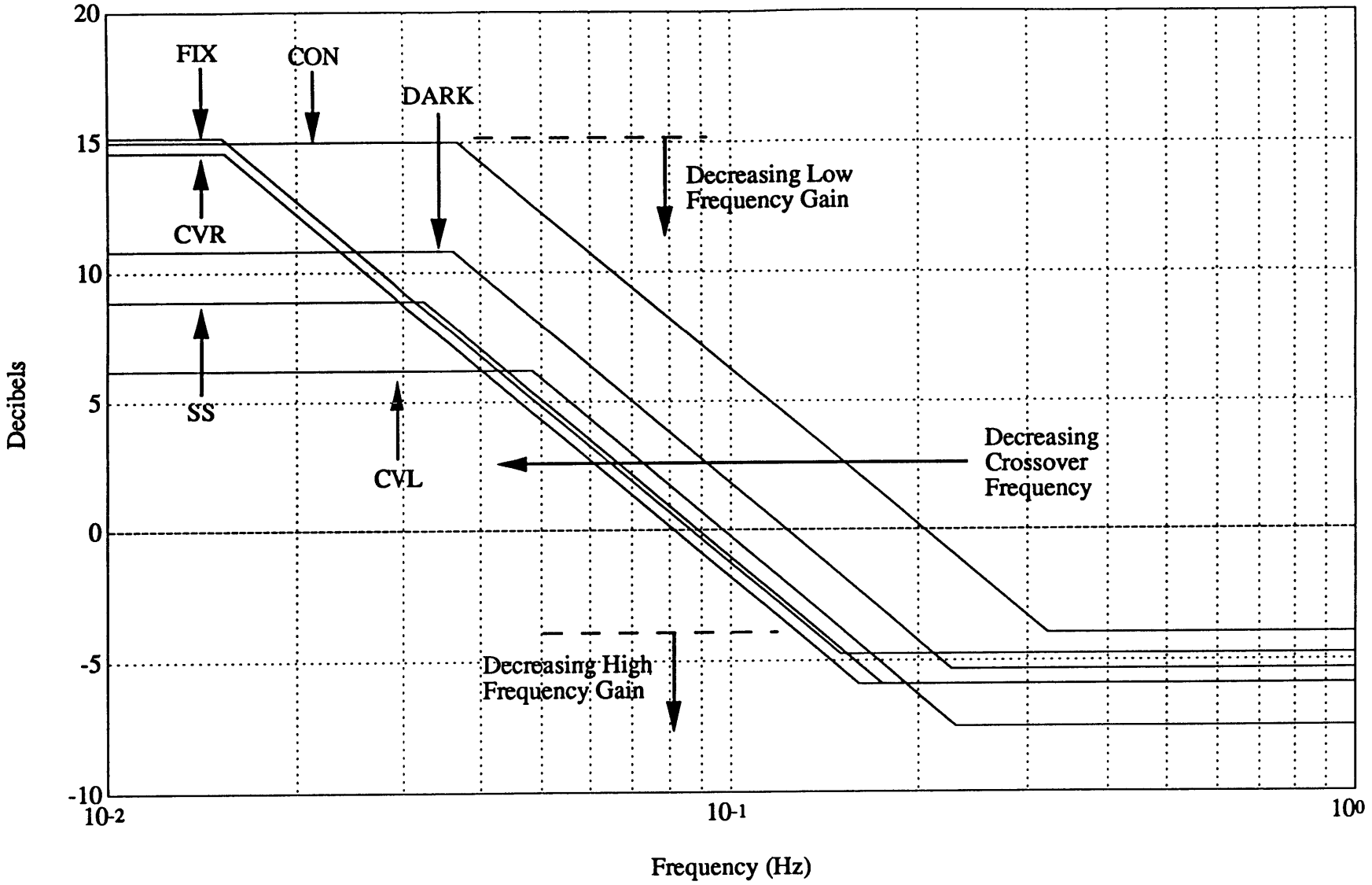


Figure 8.1 : Asymptotic approximations to population mean operator describing function fits for the six different visual fields in the manual roll stabilization experiment.

lead time constant (τ_1), lag time constant (τ_2), high frequency gain ($K_{high} \equiv K_{low}\tau_1/\tau_2$), and zero dB crossover frequency (ω_c) for each of the visual fields used in the experiment.

	CON	CVL	CVR	DARK	FIX	SS
K_{low}	14.975	6.197	14.575	10.794	15.193	8.887
τ_1	0.494	0.692	0.992	0.703	1.061	0.906
τ_2	4.327	3.270	10.414	4.413	10.510	4.895
K_{high}	-3.931	-7.482	-5.886	-5.270	-4.768	-5.912
ω_c	0.203	0.087	0.080	0.120	0.086	0.084

Table 8.1: Summary of low frequency gain (dB), lead time constant (s), lag time constant (s), high frequency gain (dB), and crossover frequency for the asymptotic approximations shown in figure 8.1.

Low frequency gain (measured in dB) was noticeably affected by the visual field, illustrating that low frequency sensation is driven primarily by the visual field. In the case of confirming visual fields (CON), the strong visual tilt angle error cue provided the subject with tilt error magnitude and direction information at all frequencies in the disturbance and allowed the subject to use a high gain control strategy at low frequencies. Non-confirming, non-disorienting visual fields (FIX and DARK), provided the subject with no additional information on position error magnitude or direction; this forced the subject to depend entirely on vestibular cues to perform the nulling task. In the case of DARK, this resulted in a reduced low frequency gain. For FIX, however, the strong low frequency otolith cue interacted with the visual cue and produced a gain comparable to CON. In the case of disorienting visual fields (CVL, CVR, and SS), the effect of increased subject confusion about what was "upright" as well as the enhanced response of the subject to the visual disturbance (particularly at low frequencies) resulted in a decreased low frequency gain. Although CVR had the third largest low frequency gain, it was due to a measurement artifact rather than a real effect, and thus CVR will not be included in the following

discussion. Finally, SS had a larger low frequency gain than CVL due to the position bias induced by CVL (this results in a larger ϕ, and thus a smaller $-\lambda/\phi$).

Lead time constant, lag time constant, high frequency gain, and crossover frequency were also affected by the visual field. The different values of the lag time constant seem to provide compensation at the mid-to-high and high frequencies for the different levels of gain. This was further illustrated by the crossover frequencies for each of the visual field conditions. All visual fields except CON had crossover frequencies in the narrow range 0.08 - 0.12 Hz. This shows that providing the subject with non-confirming or disorienting visual cues resulted in a similar region in which the subject used the high gain control strategy characteristic of good position nulling. In the case of CON, the crossover frequency was boosted to 0.2 Hz. This increase in the frequency range over which the subject employed a high gain strategy was caused by the presence of the confirming visual cues. The high frequency gain is comparable among the six presentations, supporting the relative indifference to visual inputs at high frequencies (dependence is primarily on the canals) [13]. That the largest high frequency gain was seen with CON is intuitive, since the confirming visual cues boosted performance at all frequencies in the disturbance. The smallest high frequency gain was, as expected, seen with CVL, showing that the provocative low frequency visual cues reduce performance slightly in the high frequency range. The remaining visual fields had very similar high frequency gains, showing that providing the subject with non-confirming or pseudo-random visual cues had little effect on high frequency gain, and thus high frequency nulling proficiency. This further supports the hypothesis that the subject does not make use of high frequency visual cues in the nulling task.

8.2.2.3. SPM

Consistent SPM trends were seen for the six visual fields used in the manual roll stabilization experiment. The largest SPM (and thus best nulling proficiency) was seen, as expected, with CON for all subjects. This was due to the confirming visual cues, which

provided the subjects with an absolute reference against which to estimate (and thus correct for) his roll position error. Four of the 6 subjects (A, C, D, and F) had their next best performance with DARK, followed closely by FIX. Although FIX had an enhanced low frequency gain (as discussed in section 8.2.2.2), roll-off occurs much sooner than with DARK (low frequency gains are equal at 0.035 Hz but DARK is constant and FIX is falling off at -20 dB/decade) which results in comparable SPMs (or even slightly better for DARK). This does not contradict the result that RMS trainer position was less for FIX than DARK since the subject's remnant was smaller for FIX than DARK. Six of 6 subjects had their smallest SPMs with either SS or CV. It is certainly intuitive that DARK and FIX should be better than with these fields, since in the former dependence is on vestibular cues only, while in the latter the subject is provided with a disorienting visual field. In the case of SS, the subject responds to the low frequency components of the visual stimulus (below approximately 0.16 Hz), which results in a reduced low frequency gain at the vestibular frequencies, and thus a reduced SPM. With CV, the roll vection generated by the field causes the subject to use a low gain control strategy as well as to bias the trainer in the direction of the visual field, both of which combine to reduce the SPM slightly below that for SS. These results were preserved for the subject population, although only CON was statistically different from any of the other visual fields.

8.2.2.4. VRM

The VRM with SS was 2-3 times larger than with any of the other fields, showing that the subject indeed responded to the sum of sines visual field disturbance. For CON, FIX, DARK, CVR, and CVL, the VRMs were small. This is expected, since for these fields the VRM is a measure of subject remnant at the visual disturbance frequencies. As mentioned in section 8.2.2.1, CVR and CVL have a slightly larger remnant due to the strong low frequency roll vection. This explains why they had the largest VRM next to SS. Similarly, trials with CON had the best subject performance and smallest remnant, so it is not surprising that they had the smallest VRM. That FIX had a slightly smaller VRM

(due to the otolith-visual interaction) than DARK supports the assertion in section 8.2.2.3 that FIX had a smaller remnant.

8.2.2.5. McRuer Crossover Models

The crossover model of McRuer et al. states that in the region of the crossover frequency (ω_c), the open-loop transfer function (product of the operator describing function and the Link dynamics) behaves as an integrator with a pure delay, i.e.:

$$C(s)E(s)P(s) \approx \frac{\omega_c e^{-\tau_e s}}{s} \qquad (8.1)$$

where the effective delay (τ_e) is due both to reaction time and high-frequency neuromuscular dynamics and the crossover frequency ω_c is equivalent to the human operator's gain compensation [21]. Thus, in the region of crossover, we would expect a phase of approximately ($-\pi/2-\omega_c\tau_e$) radians.

The open-loop transfer function for the manual roll stabilization task for each of the six visual fields are shown in figures 5.22-5.27. Inspection of these figures reveals two important facts: (1) all the open-loop transfer functions have the large gain at low frequency characteristic of good "command" following (the "command" is to maintain upright as shown in figure 4.3 of chapter 4); and (2) in the region of crossover, the slope was nearly the -20 dB/decade predicted by the Crossover Model. Table 8.2 summarizes the crossover frequencies for each of the visual fields, as well as the actual phase at crossover and that predicted by the Crossover Model. All visual fields except CON had crossover frequencies in the narrow frequency band of 0.11-0.15 Hz, placing crossover in a region of approximately constant phase. Due to the confirming visual cues, the crossover frequency (i.e. the region of good command following) for CON was pushed out to approximately 0.26 Hz, placing crossover in a region of decreasing phase (and thus a less stable regime). In all cases, the predicted phase lag was larger than the actual phase lag, implying the existence of subject generated phase lead which increased the phase margin, and thus enhanced the closed-loop stability of the system. The semicircular canals are a likely

source of this phase lead, since they are good velocity sensors in the frequency range of this experiment. Due to the flattening of the gain at high frequencies and the rapidly decreasing phase, the subjects could not increase their gain significantly without driving the system unstable. Thus in all cases, the subject population chose a very good control strategy for the manual roll stabilization task.

	ω_c (Hz)	Actual Phase (degrees)	Predicted Phase (degrees)
CON	0.261	-112.210	-152.724
CVL	0.108	-79.929	-116.045
CVR	0.112	-83.127	-115.621
DARK	0.146	-86.091	-125.843
FIX	0.127	-83.188	-118.576
SS	0.125	-80.857	-120.788

Table 8.2: Summary of manual roll stabilization crossover frequency, actual crossover phase, and crossover phase predicted by McRuer Crossover Model.

8.3. Manual Lateral Stabilization

This section discusses the population time and frequency domain results for the manual lateral stabilization experiments.

8.3.1. Time Domain

This section discusses the results seen in the manual lateral stabilization experiments for mean and RMS sled velocity.

8.3.1.1. Mean Sled Velocity

Each of the visual fields used in this experiment had a very small effect on mean

sled velocity. The existence of a non-zero mean sled velocity is not surprising, since any dc velocity component is below the otolith acceleration threshold, and thus cannot be detected. Interestingly, CON consistently had large mean sled velocities. This is result of the much more aggressive control strategy employed by the subject as well a windowshade velocity drift caused by the counterrotating visual field circuitry. No other general trends were seen.

8.3.1.2. RMS Sled Velocity

Similarly, each of the visual fields had an effect on RMS sled velocity. The best performance in the velocity nulling task was, as expected, seen with CON. That CON should have the smallest RMS velocity (and thus the best nulling proficiency) is intuitive, since the confirming visual cues provided by the countermoving visual field provide the subject with an "absolute" reference with which to estimate his velocity error. Most subjects said they were nulling windowshade velocity rather than sled velocity, and thus were depending on visual, rather than vestibular, cues. Trials with the FIX field had the largest RMS velocity, followed by SS and DARK, however these differences were not statistically significant. In the case of FIX and DARK trials, the subject was depending solely on vestibular cues, and therefore we would expect trials with these visual fields to have a larger RMS velocity than CON. That DARK performance was better than FIX is in disagreement with Huang, who saw the opposite trend in his lateral stabilization experiments (his differences were also not statistically significant, however) [14]. Additional trials would be required to determine which result is correct. That SS performance was not significantly different from DARK or FIX is again in disagreement with Huang, who saw RMS velocities for SS that were at least twice as large [14]. This is most likely due to the fact that in Huang's dual input experiment, the subject's joystick also controlled the windowshade velocity, which would make it more difficult for the subject to ignore the visual field. Further discussion of subject performance with SS is included in section 8.3.2.3.

8.3.2. Frequency Domain

This section discusses the results seen in the manual lateral stabilization experiments for the frequency response of the subjects.

8.3.2.1. Amplitude Spectra

Close inspection of figures 7.12 reveals several general trends: (1) subjects tended to undercompensate for the vestibular disturbance; (2) nulling was best at low frequencies, while at high frequencies energy was consistently added to the system; (3) subjects did not respond to the sum of sines pseudo-random visual stimulus since the vision response is within the remnant; and (4) remnant responses were smallest with CON, and were similar for DARK, FIX, and SS. Comparison of nulling proficiency with the four visual fields is discussed in section 8.3.2.4.

8.3.2.2. Operator Describing Functions

Since the quality of the fits to the individual operator describing functions were less than satisfactory (see section 7.2.2.3 of chapter 5), this section only discusses the effect of the visual field on the population mean operator describing functions. Figure 8.2 shows asymptotic approximations to the frequency domain transfer function fits to the mean operator describing functions for each of the visual fields using the values given in table 7.24 of chapter 7. Table 8.3 summarizes the values for low frequency gain (K_{low}), lead time constant (τ_1), lag time constant (τ_2), and high frequency gain (K_{high}), all of which were defined in section 8.2.2.2.

Low frequency gain (measured in dB) was affected by the visual field, illustrating that low frequency sensation is driven primarily by the visual field. In the case of confirming visual fields (CON), the strong velocity error cue provided the subject with velocity error magnitude and direction information at all frequencies in the disturbance and allowed the subject to use a high gain control strategy at low frequencies. Non-confirming, non-disorienting visual fields (FIX and DARK), provided the subject with no additional information on velocity error magnitude or direction; this forced the subject to depend

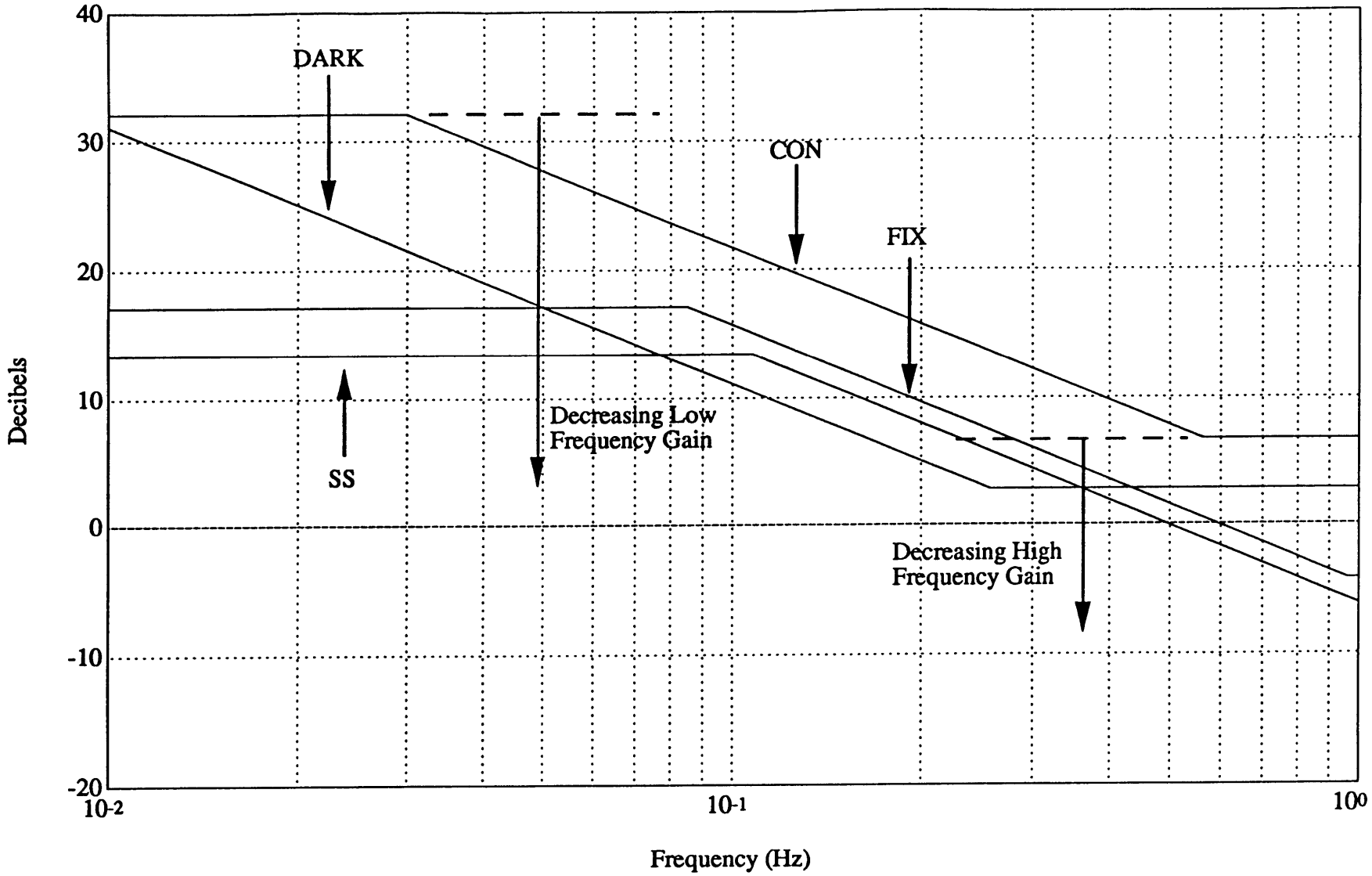


Figure 8.2: Asymptotic approximations to population mean operator describing function fits for the four different visual fields in the manual lateral stabilization experiments.

entirely on vestibular cues to perform the nulling task. In the case of DARK, although the low frequency gain was large, roll-off began early such that the gain was less than for FIX above approximately 0.06 Hz. For FIX, interaction of the strong low frequency otolith and visual cue produced a gain comparable to DARK and pushed the onset of roll-off to higher frequencies. In the case of disorienting visual fields (SS), the confusing visual information caused the subject to use the least low frequency gain, supporting the influence of the visual field at low frequencies.

	CON	DARK	FIX	SS
K_{low}	32.172	40.000	17.008	13.282
τ_1	0.282	0.615	0.165	0.100
τ_2	5.273	43.997	1.871	1.469
K_{high}	6.723	2.908	-4.117	-10.079

Table 8.3: Summary of low frequency gain (dB), lead time constant (s), lag time constant (s), and high frequency gain (dB) for the asymptotic approximations shown in figure 8.2.

Lead time constant, lag time constant, and high frequency gain were also affected by the visual field. The different values of the lag time constant seem to provide compensation at the mid and high frequencies for the different levels of gain. That the largest high frequency gain was seen with CON was intuitive, since the confirming visual cues boosted performance at all frequencies in the disturbance. DARK, FIX, and SS high frequency gain were comparable, again showing the relative indifference to visual inputs at high frequencies. It therefore appears that in the manual lateral stabilization task, the confusing nature of SS caused the subject to attempt to ignore the visual field and to adopt a control strategy more similar to that seen with FIX.

8.3.2.3. SPM

Each of the visual fields used in this experiment had a different affect on the mean SPMs. CON, not surprisingly, had the largest SPM (and was significantly different from

the other visual field conditions), since the confirming visual cues provided the subject with an "absolute" reference against which to estimate his velocity error. The next best performance was seen with DARK, followed closely by SS and FIX, however these differences were not significant. Although roll-off occurred much sooner with DARK than FIX, the large low frequency gain in DARK as well as the addition of energy by the subject at high frequencies (> 0.4 Hz) in FIX resulted in a smaller SPM. This supports the result that the RMS sled velocity was less with DARK than FIX. SS and FIX performance were similar, as discussed in section 8.3.2.2.

8.3.2.4. VRM

The VRMs for SS were not significantly different from those for CON, DARK, and FIX implying that the subjects in the manual lateral stabilization experiment did not respond to the pseudo-random visual disturbance. In fact, the majority (if not all) of the visual responses were within the remnant. These results agree with Huang, who saw an enhanced RMS velocity with SS, but no consistent response at the frequencies in the visual disturbance [14]. The smallest VRM was again seen with CON where subject performance was best. DARK and FIX were comparable, with FIX having a slightly larger VRM (and thus remnant). This in no way contradicts the results that DARK RMS sled velocity was smaller than FIX, since the VRM only gives an estimate of the remnant based on 10 frequencies. This results should not be interpreted to limit the use of full field visually induced motion in simulation, but is limited only to this experimental setup.

8.3.2.5. McRuer Crossover Models

The open-loop transfer function for the manual roll stabilization task for each of the four visual fields are shown in figures 7.18-7.21. Inspection of these figures reveals two important facts which apply to all visual fields except FIX: (1) all the open-loop transfer functions have the large gain at low frequency characteristic of good "command" following (the "command" is to maintain zero velocity as shown in figure 6.6 of chapter 6); and (2) in the region of crossover, the slope was nearly the -20 dB/decade predicted by the Crossover

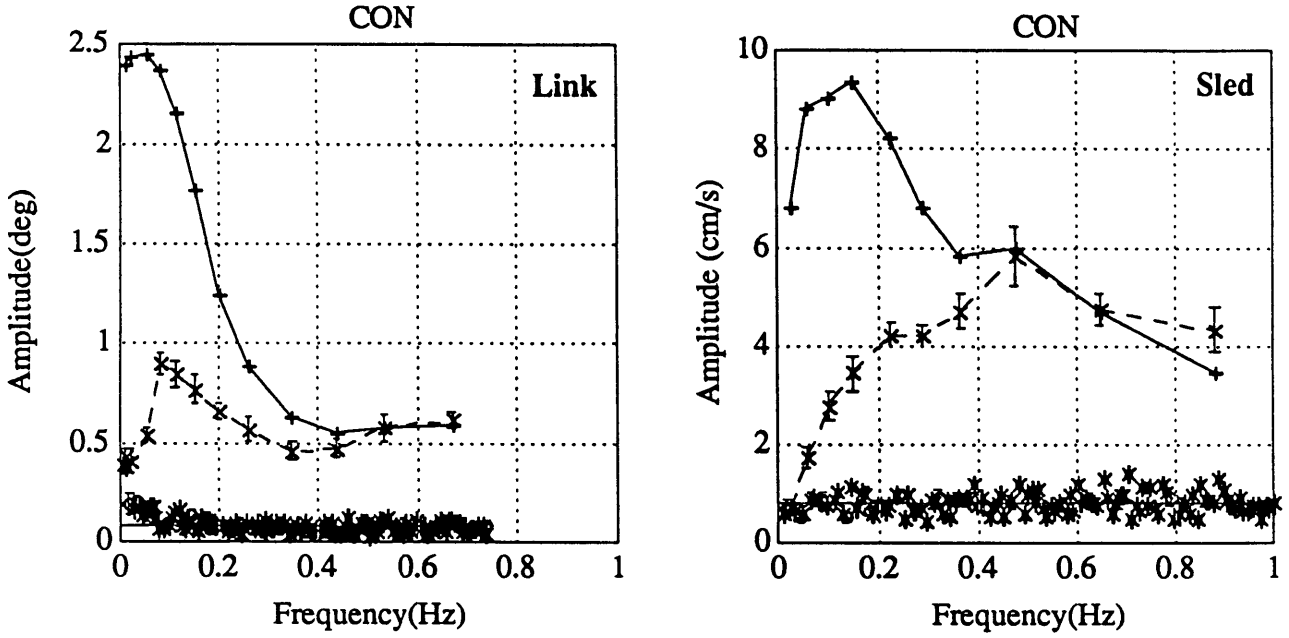
Model. Table 8.4 summarizes the crossover frequencies for each of the visual fields, as well as the actual phase at crossover and that predicted by the Crossover Model. Values for FIX could not be calculated since the model fit did not crossover. However, inspection of the open-loop transfer function for FIX (see figure 7.20 in chapter 7) shows that the actual data did indeed crossover with a slope near -20 dB/decade. Note that the crossover frequencies in manual lateral stabilization are much higher than their counterparts in the roll experiments. This is not surprising, since the dynamics of the sled are inherently more stable than those of the Link (see figure 6.1 in chapter 6). In all cases, the predicted phase lag was larger than the actual phase lag, implying the existence of subject generated phase lead which increased the phase margin and thus enhanced the closed-loop stability of the system. That the magnitude of the phase lead is less than with the manual roll stabilization experiments is most likely due to the absence of the velocity cue provided by the semicircular canals.

	ω_c (Hz)	Actual Phase (degrees)	Predicted Phase (degrees)
CON	0.562	-104.009	-124.613
DARK	0.905	-150.955	-184.179
FIX	----	----	----
SS	0.514	-102.687	-131.821

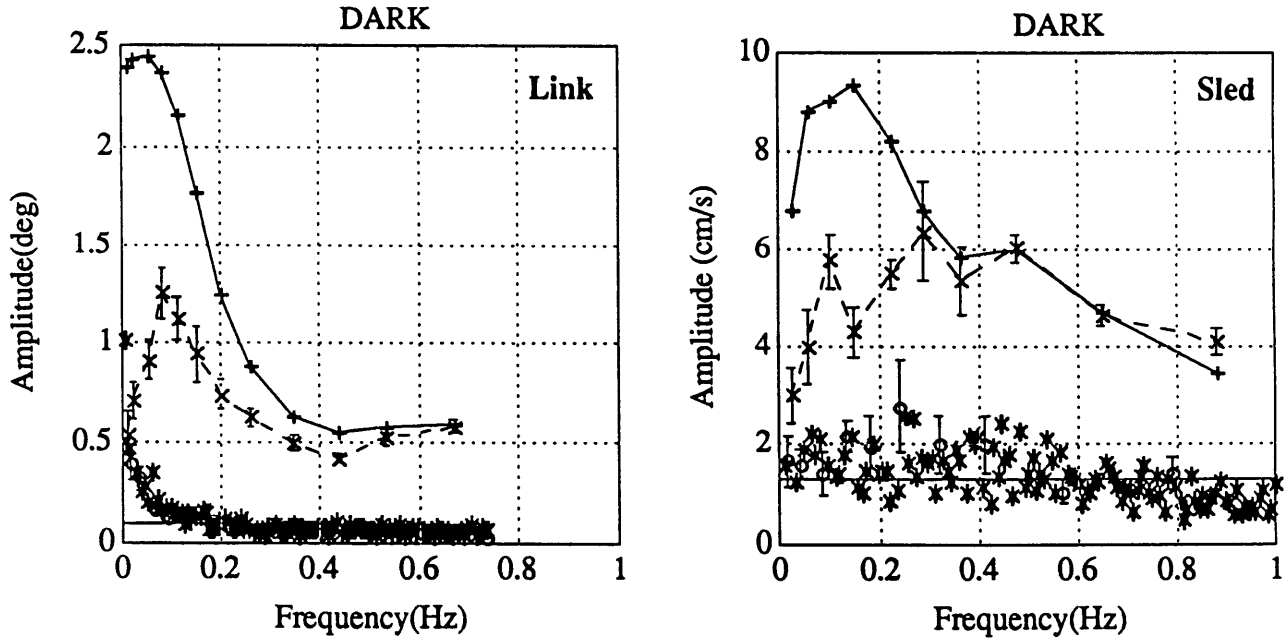
Table 8.4: Summary of manual lateral stabilization crossover frequency, actual crossover phase, and crossover phase predicted by McRuer Crossover Model.

8.4. Comparison of Manual Roll and Lateral Stabilization

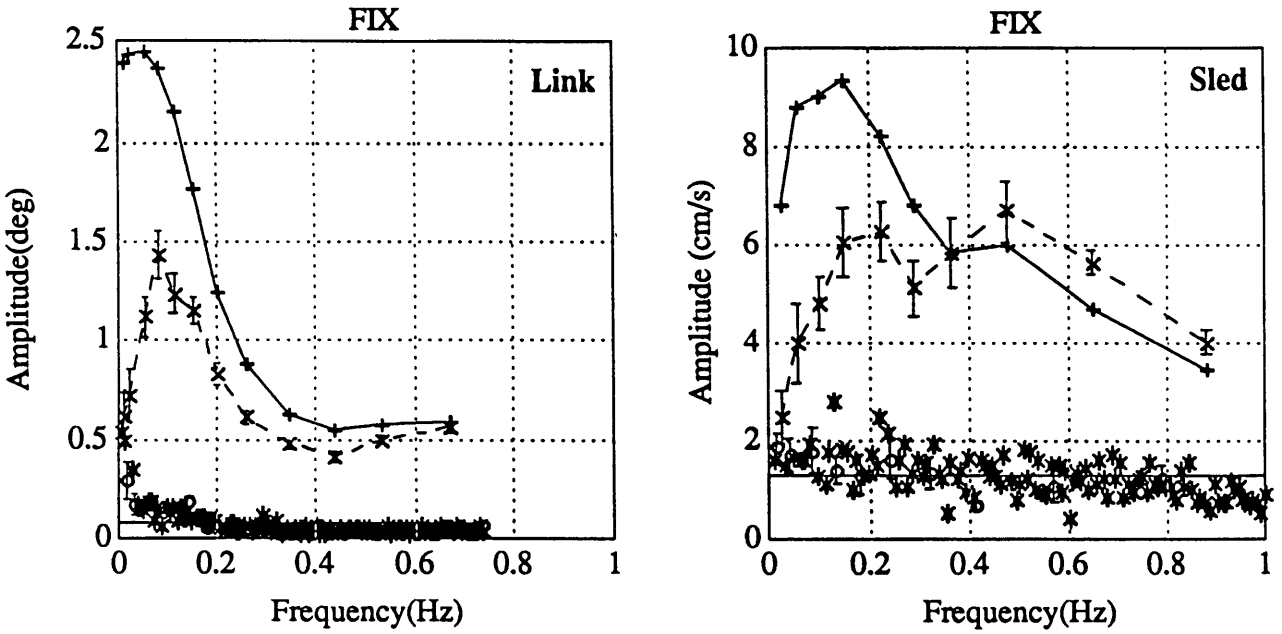
Figure 8.3 shows the amplitude spectra for the pooled results for subjects B, D, and F (who took part in both the manual roll (MRS) and lateral stabilization (MLS)


CON
Link
Amplitude(deg)
Frequency(Hz)
CON
Sled
Amplitude (cm/s)
Frequency(Hz)

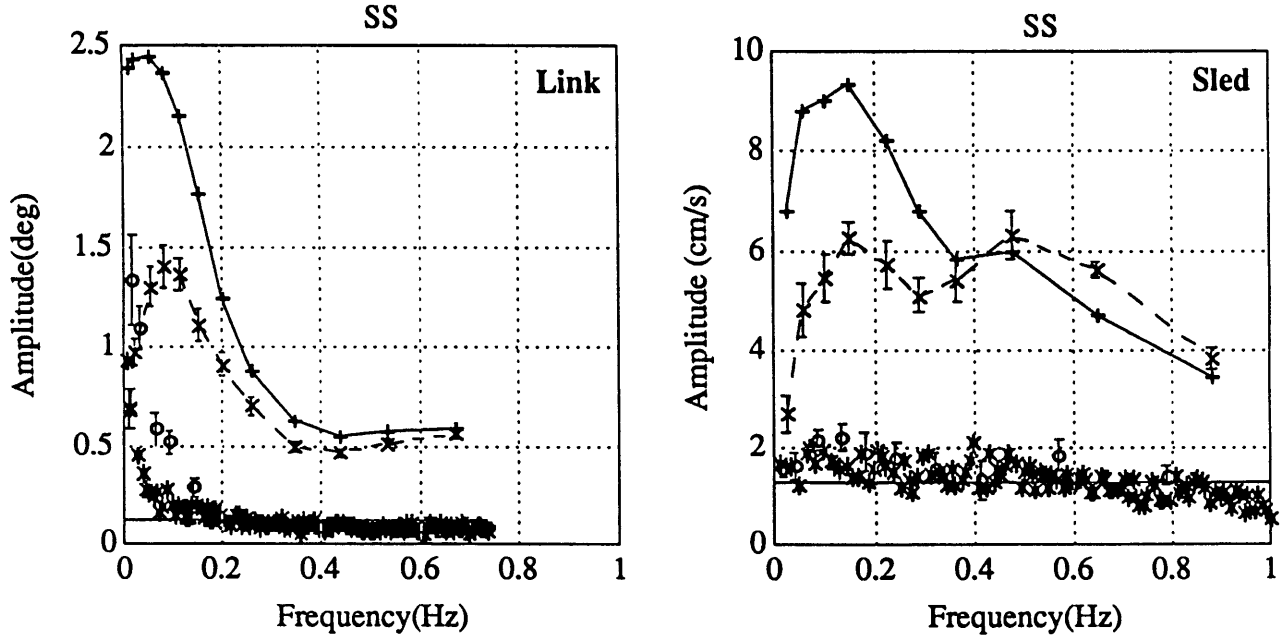
(a)


DARK
Link
Amplitude(deg)
Frequency(Hz)
DARK
Sled
Amplitude (cm/s)
Frequency(Hz)

(b)



(c)



(d)

Figure 8.3: Comparison of population mean (+/- se) position and velocity amplitude spectra for manual roll and lateral stabilization, respectively. x's are power at vestibular disturbance frequencies, o's at visual disturbance frequencies, and *'s are subject remnant for each of the visual fields .

experiments) for the CON, DARK, FIX, and SS visual fields. A few general points are noticeable from inspection of the figure: (1) the mean remnant was much larger (when compared to the vestibular disturbance) and more "scattered" in MLS than MRS; (2) subjects consistently added energy at the high frequencies in MLS, but only did so once with MRS (at the highest frequency in CON); (3) low frequency nulling proficiency was good in both MRS and MLS, and tended to worsen with increasing frequency; (4) subjects responded to the SS visual field in MRS, but did not in MLS; (5) error bars were much smaller in MRS than MLS (particularly at high frequencies), implying a higher repeatability in MRS; and (6) SPMs were higher for all subjects for all visual fields in MRS than MLS, as summarized in table 8.5.

MRS	CON	DARK	FIX	SS
B	0.535 ± 0.008	0.376 ± 0.015	0.401 ± 0.008	0.311 ± 0.025
D	0.628 ± 0.017	0.543 ± 0.010	0.410 ± 0.002	0.436 ± 0.028
F	0.641 ± 0.021	0.608 ± 0.020	0.596 ± 0.017	0.506 ± 0.012
MLS	**CON**	**DARK**	**FIX**	**SS**
B	0.417 ± 0.087	0.178 ± 0.064	0.143 ± 0.022	0.161 ± 0.051
D	0.450 ± 0.014	0.201 ± 0.011	0.177 ± 0.053	0.175 ± 0.028
F	0.534 ± 0.007	0.484 ± 0.041	0.467 ± 0.011	0.435 ± 0.035

Table 8.5: Comparison of MRS and MLS subject mean SPMs (± se) for each visual field.

In general, performance differences between MRS and MLS were due primarily to the high frequency components of the disturbance. At low frequencies (in the absence of a visual field), the strong otolith cue resulted in a well-met task objective for both MRS and MLS. For MRS, the subjects were able to reduce trainer position for frequencies out through approximately 0.7 Hz for all visual fields, adding very little (if any) energy. On the other hand, for MLS, at frequencies above approximately 0.4 Hz for DARK, FIX, and

SS visual fields, the subject's consistently added energy. This same behavior was seen with CON, although the confirming visual cues increased the frequency above which the subjects added energy to approximately 0.6 Hz. Similar high frequency behavior to CON was seen with DARK, where the subject added energy above 0.6 Hz. This increased frequency could be due to a reduction in the subject's acceleration threshold due to an oculogyral illusion created by the fixation point (see Huang [13]), or tentativeness by the subject since the trials were done in the DARK.

8.5. Recommendations for Further Research

Natural continuations of the work begun in this thesis would be to: (1) perform the complementary manual stabilization experiments in the pitch axis; (2) attempt to model visual-vestibular interaction (VVI) in roll and pitch; (3) perform the sled velocity nulling experiments in the z (head-to-foot) axis and compare with performance in the y axis documented in this thesis; (4) quantify changes in vection strength due to spaceflight; and (5) investigate the Otolith Tilt Translation Reinterpretation Hypothesis.

8.5.1. Manual Pitch Stabilization

A similar effect of the visual field is expected on manual pitch stabilization and would be a valuable study. Huang has performed the study for constant velocity visual fields [14], but not for a pseudo-random uncorrelated visual disturbance. The experimental procedure used in the roll experiment discussed in this thesis could easily be extended to pitch.

8.5.2. Modeling VVI in Roll and Pitch

An interesting study would be to attempt to model VVI in the roll and pitch axes. The conflict model developed by Zacharias in yaw or existing optimal control models could potentially be extended to roll and pitch [27]. This would provide a means of functionally describing the otolith-canal interactions, which would eventually allow for combination with visual motion cues.

8.5.3. Manual Stabilization in the Z Axis

Another interesting study would be to investigate subject sled velocity nulling proficiency in the z axis. Preliminary studies indicate that subject performance is worse than in the y axis, but no attempt has been made to quantify these differences. The methods developed in this thesis could easily be extended to this study.

8.5.4. Quantifying Changes in Vection Strength Due to Spaceflight

Another interesting study would be to attempt to quantify changes in circularvection strength as a result of spaceflight. Currently, one measure of vection strength is obtained in the dome experiment by having the subject indicate his level of vection using a joystick. This measure is quite subjective, however, since the subject is asked to scale his current vection to levels experienced in the past. What is needed is a more objective measure of vection strength so that hypothesized changes due to spaceflight can be studied with less confounding variables. One simple study would be to have the subject null his perceived velocity about the earth-vertical axis (yaw) in the presence of constant velocity or pseudo-random visual stimuli. Subject frequency response information could be derived from these measurements. This experiment could easily be performed using the equipment developed in this thesis as a preflight/posflight protocol.

8.5.5. Otolith Tilt Translation Reinterpretation (OTTR)

8.5.5.1. What is it?

When an astronaut enters the microgravity environment, the otolith organs which usually supply the brain with information on the orientation of the head with respect to gravity, can no longer sense gravity (the astronaut is in free-fall). The brain therefore receives an incorrect motion cue from the otoliths which does not agree with the information coming from the other senses (primarily vision). The OTTR hypothesis says that during spaceflight, the brain learns to reinterpret the "faulty" otolith cue as a linear acceleration of the head (or body) rather than a change in head (or body) orientation with respect to gravity.

8.5.5.2. How can we investigate it?

The tools developed in this thesis provide a good method to investigate the OTTR hypothesis. If OTTR is correct, we would expect that the astronauts will become more sensitive to linear accelerations of the head (or body) and less sensitive to tilt with respect to gravity. In other words, we can hypothesize that the performance of the astronauts on the manual lateral stabilization task should improve, and their performance on the manual roll stabilization task should worsen. Thus if the postflight performance (compared to preflight) indicates the above trends, this would be strong evidence supporting OTTR.

At the writing of this thesis, these exact experiments are underway as a part of the E-072 experiment series for the Spacelab Life Sciences (SLS)-2 shuttle mission scheduled to be launched in late 1993. The results should provide evidence for or against the OTTR hypothesis.

REFERENCES

[1] Arrott, A. P.and L. R. Young, "M.I.T./Canadian vestibular experiments on the Spacelab-1 mission: 6. Vestibular reactions to lateral accelerations following ten days of weightlessness", Experimental Brain Research (1986) 64 : pp. 347-357.

[2] Arrott, A. P., L. R. Young, and D. M. Merfeld, "Perception of Linear Acceleration in Weightlessness", Aviation Space and Environmental Medicine (April, 1990) : pp. 319-326.

[3] Breuer, J., *Ueber die Funktion der Bogengange de Ohrlabyrinths Med. Jahrbucher.* (Wien) 72-124.

[4] Dichgans, J., R. Held, L. R. Young, and T. Brandt, "Moving Visual Scenes Influence the Apparent Direction of Gravity", Science 178 : 1217-1219, 1972.

[5] Dinsdale, P. B., Relative Effects of Roll and Yaw Motion Cues in Manual Control, S. M. Thesis, Massachusetts Institute of Technology, Cambridge, MA 1968.

[6] Flourens, P., *Experiences sur les canoux semi-circulaires de l'oreille, dans les oiseaux.* Mem. Acad. R. Sci Med, Paris 9: 455-477 (1830).

[7] Flourens, P., R*echerches Experimentales sure les Proprietes et les Fonctions du Systeme Nerveux dans les Animaux Vertebres.* Paris: J-B Bailliere (1842).

[8] Graybiel, A. "The Vestibular System", Bioastronautics Data Book, Second Edition : pp. 533-609, 1973.

[9] Held, R., J. Dichgans, and J. Bauer, "Characteristics of moving visual scenes influencing spatial orientation", Vision Res 14: 1-9, 1974.

[10] Henn, V., B. Cohen, and L. R. Young, Visual Vestibular Interaction in Motion Perception and the Generation of Nystagmus, Neurosciences Research Program Bulletin, MIT Press, Volume 18, Number 4, September, 1980.

[11] Hiltner, D. W., A Closed-Loop Otolith System Assessment Procedure, S. M. Thesis, MIT, Cambridge, MA, 1983.

[12] Huang, J. K. and L. R. Young, "Sensation of Rotation About a Vertical Axis with a Fixed Visual field in Different Illuminations and in the Dark", Experimental Brain Research (1981) 41 : pp. 172-183.

[13] Huang, J. K..Visual Field Influence on Motion Sensation in Yaw and on Manual Roll Stabilization, S. M. Thesis, MIT, Cambridge, MA, 1979.

[14] Huang, J. K., Visual and Motion Cues in Lateral and Pitch Simulator Stabilization, Ph. D. Thesis, MIT, Cambridge, MA, 1983.

[15] Huang, J. K. and L. R. Young, "Influence of Visual and Motion Cues on Manual Lateral Stabilization", Aviation, Space and Environmental Medicine (December, 1987) : pp. 1197-1204.

[16] Huang, J. K. and L. R. Young, "Visual Field Influence on Manual Roll and Pitch

Stabilization", Aviation, Space and Environmental Medicine (July, 1988) : pp. 611-619.

[17] Mach, E., *Physikalische Versuche uber den Gleichgewichtsinn des Menschen. 1. Mitteilung. Akad. Wiss. Wien* 68 : 124-140, Abt. 3 (1873).

[18] Mach, E., *Grundlinien der Lehre von den Bewegungsempfindungen.* Leipzig: Wilhelm Engelmann (1875).

[19] McRuer, D., "Development of Pilot-in-the-Loop Analysis", Journal of Aircraft (September, 1973), Vol. 10, No. 9 : pp. 515-524.

[20] Purkinje, J., *Betrage zur naheren Kenntnis des Schwindels aus heautognostischen Daten. Medizinisches Jahrbuch des K. K. Oest. Staates* Vienna 6: 79-125 (1820).

[21] Sheridan, T. B. and W. R. Ferrell. Man-Machine Systems : Information, Control, and Decision Models of Human Performance (MIT Press, 1974).

[22] Shirley, R. S. and L. R. Young, "Motion Cues in Man-Vehicle Control : Effects of Roll-Motion Cues on Human Operator's Behavior in Compensatory Systems with Disturbance Inputs", IEEE Transactions on Man-Machine Systems (December, 1968), Vol. MMS-9, No. 4 : pp. 121-128.

[23] Taylor, J. H., "Vision", Bioastronautics Data Book, Second Edition : pp. 611-665, 1973.

[24] Young, L. R., "Human Control Capabilities", Bioastronautics Data Book, Second Edition : pp. 751-806, 1973.

[25] Young, L. R., J. Dichgans, R. Murphy, and Th. Brandt, "Interaction of Optokinetic and Vestibular Stimuli in Motion Perception", Acta Otolaryng (1973) 76 : pp. 24-31.

[26] Young, L. R., C. M. Oman, and J. M. Dichgans, "Influence of Head Orientation on Visually Induced Pitch and Roll Sensation", Aviation, Space and Environmental Medicine (March, 1975) : pp. 264-268.

[27] Zacharias, G. L..Motion Sensation Dependence On Visual and Vestibular Cues, Ph. D Thesis, MIT, Cambridge, MA, 1977.

[28] Zacharias, G. L. and L. R. Young, "Influence of Combined Visual and Vestibular Cues on Human Perception and Control of Horizontal Rotation", Experimental Brain Research (1981) 41 : pp. 159-171.

APPENDIX A

PROGRAM LISTINGS FOR LINK SUM OF SINES PROFILE GENERATORS

SMSSINTG.CPP

SMSSINTG.HPP

SSTG.CPP

SSTG.HPP

A.1. Introduction

This appendix contains program listings of a subset of the routines used to generate the pseudo-random sum of sines profiles used within this thesis for the Link experiments. There are four total profile generators, two for the auxilliary axis, and two for the main Link axis. Each routine is compiled and linked into the existing MVL Link software, and thus they are not intended as stand-alone routines. Similar routines were used for the sled experiments, with minor changes in window formatting since the sled uses linear (rather than angular) units. Since the sled routines were so similar, no listings are included.

A.1. AXSMSSTG.CPP

This routine generates a filtered sum of sines profile for the auxilliary channel. A double lag-lead filter (implemented in discrete time) of the following form is used:

$$\frac{\omega_2^2}{\omega_1^2}\frac{s^2+2\zeta_1\omega_1 s+\omega_1^2}{s^2+2\zeta_2\omega_2 s+\omega_2^2} \qquad \text{(A.1a)}$$

with the following default values:

$(\omega_1,\omega_2) = (0.350, 0.150)$ Hz

$(\zeta_1, \zeta_2) = (0.707, 0.707)$ (A.1b)

In addition to items 1-6 of the list in section A.2, the user also has control over the lag and lead break frequencies of the filter. This allows a profile to be created with a more gradual transition between the large low frequency amplitudes and the small high frequency amplitudes. As before, a maximum of 20 sines is allowed. Again no listing is included, since it is similar to the Link channel routine SMSSINTG.CPP.

A.2. AXSMSSTG.HPP

This is the C++ header file for the auxiliary channel routine AXSMSSTG.CPP and is used during compilation and linking. Again a listing is not included, since it is similar to the Link channel header file SMSSINTG.HPP.

A.3. AXSSTG.CPP

This routine generates a sum of sines profile for the auxilliary channel. The user has control over the following trajectory parameters:

1. Period
2. Number of Sines
3. Equal amplitude domain (position, velocity, or acceleration)
4. Successive phase angle
5. Maximum command (in volts)
6. Frequency of each sine wave.
7. Amplitude of each sine wave.

A maximum of 20 sines is allowed. A listing of this routine is not included (in the interest of saving paper) since it is very similar to the Link channel routine SSTG.CPP.

A.4. AXSSTG.HPP

This is the C++ header file for the auxiliary channel routine AXSSTG.CPP and is used during compilation and linking. A listing is not included (in the interest of saving trees) since it is very similar to the Link axis header file SSTG.HPP.

A.5. SMSSINTG.CPP

This routine generates the a filtered sum of sines profile for the auxilliary channel. A double lag-lead filter (implemented in discrete time) of the form given in A.1a is used for filtering, and with the same default values given in A.1b. In addition to items 1-6 of the list in section A.6, the user also has control over the lag and lead break frequencies of the filter.

This allows a profile to be created with a more gradual transition between the large low frequency amplitudes and the small high frequency amplitudes. As before, a maximum of 20 sines is allowed.

A.6. SMSSINTG.HPP

This is the C++ header file for the routine SMSSINTG.CPP and is used during compilation and linking.

A.7. SSTG.CPP

This routine generates a sum of sines profile for the auxilliary channel. The user has control over the following trajectory parameters:

1. Period
2. Number of Sines
3. Equal amplitude domain (position, velocity, or acceleration)
4. Successive phase angle
5. Maximum command (in volts)
6. Frequency of each sine wave.
7. Amplitude of each sine wave.

A maximum of 20 sines is allowed. A listing of this routine is not included (in the interest of saving paper) since it is very similar to the routine SSTG.CPP.

A.8. SSTG.HPP

This is the C++ header file for the routine SSTG.CPP and is used during compilation and linking.

```
////////////////////////////////////////////////////////////////
// Title:          SMSSINTG.CPP
// Author:         Scott B. Stephenson
// Date:           September, 1993
// $Revision:      1.9.1.6
////////////////////////////////////////////////////////////////

// Interface Dependencies -------------------------------------------------

#ifndef  SMSSINTG_HPP
#include "smssintg.hpp"
#endif

// End Interface Dependencies ---------------------------------------------

// Implementation Dependencies --------------------------------------------

#ifndef __MATH_H
#include <math.h>
#endif

#ifndef __STDIO_H
#include <stdio.h>
#endif

#ifndef __STRING_H
#include <string.h>
#endif

#ifndef  DISPVARS_HPP
#include "dispvars.hpp"
#endif

#ifndef  LINKCONV_HPP
#include "linkconv.hpp"
#endif

#ifndef  TRAJEDIT_HPP
#include "trajedit.hpp"
#endif

// End Implementation Dependencies ----------------------------------------

// Begin global variable declaration
int smprimes[smssMaxSines];
double w13,w23;
double z13,z23;
float phaseAngle3;
float usableTrack3;
float allowedDeriv3;
// End global variable declaration
```

```
class SmoothSumSineEditForm : public TrajEditForm {
public:
        SmoothSumSineEditForm(SmoothSumSineTG *traj,int rate,int flag);

        static int validateDuration(void *item, int ccode);
        static int validateFrequency(void *item, int ccode);
        static int validateNumberSines(void *item, int ccode);
        static int validatephaseAngle3(void *item, int ccode);
        static int validateusableTrack3(void *item, int ccode);
        static int validateAllowedAccel(void *item, int ccode);
        static int validateAllowedVel(void *item, int ccode);
        static int validateEqualDomain(void *item, int ccode);
        static int validatez13(void *item, int ccode);
        static int validatez23(void *item, int ccode);
        static int validatew13(void *item, int ccode);
        static int validatew23(void *item, int ccode);
        static void getPrimes(void *item, UI_EVENT &event);

private:
        int doValidateDuration(void *item, int ccode);
        int doValidateFrequency(void *item, int ccode);
        int doValidateNumberSines(void *item, int ccode);
        int doValidatephaseAngle3(void *item, int ccode);
        int doValidateusableTrack3(void *item, int ccode);
        int doValidateAllowedAccel(void *item, int ccode);
        int doValidateAllowedVel(void *item, int ccode);
        int doValidateEqualDomain(void *item, int ccode);
        int doValidatez13(void *item, int ccode);
        int doValidatez23(void *item, int ccode);
        int doValidatew13(void *item, int ccode);
        int doValidatew23(void *item, int ccode);
        void dogetPrimes(void *item, UI_EVENT &event);

        };

SmoothSumSineEditForm::SmoothSumSineEditForm(SmoothSumSineTG *traj,int rate,int
flag) :
                TrajEditForm(traj,rate,3,3,51,16,flag,0) {
        }
int SmoothSumSineEditForm::validateDuration(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SmoothSumSineEditForm *)number->parent)->doValidateDuration(item,
ccode));
        }

int SmoothSumSineEditForm::doValidateDuration(void *item,int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();
        SmoothSumSineEditForm *me = (SmoothSumSineEditForm *)(((UIW_NUMBER
*)item)->parent);
        SmoothSumSineTG *mine = (SmoothSumSineTG *)me->myTraj;
```

```
        if (mine->verifyDuration()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%f is not valid. The value must be greater than %f", value,0.0);
                return -1;
                }
        else
                return 0;
        }

int SmoothSumSineEditForm::validateFrequency(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SmoothSumSineEditForm *)number->parent)-
>doValidateFrequency(item, ccode));
        }

int SmoothSumSineEditForm::doValidateFrequency(void *item,int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();
        SmoothSumSineEditForm *me = (SmoothSumSineEditForm *)(((UIW_NUMBER
*)item)->parent);
        SmoothSumSineTG *mine = (SmoothSumSineTG *)me->myTraj;

        if (mine->verifyFrequency()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%f is not valid. The value must be greater than %f, but less than"
                        " %f", value,0.0,getMaximumFrequency());
                return -1;
                }
        else
                return 0;
        }

int SmoothSumSineEditForm::validateNumberSines(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SmoothSumSineEditForm *)number->parent)-
>doValidateNumberSines(item, ccode));
        }

int SmoothSumSineEditForm::doValidateNumberSines(void *item,int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        int value = *(int *)field->DataGet();
        SmoothSumSineEditForm *me = (SmoothSumSineEditForm *)(((UIW_NUMBER
*)item)->parent);
        SmoothSumSineTG *mine = (SmoothSumSineTG *)me->myTraj;

        if (mine->verifyNumberSines()) {
                _errorSystem->ReportError(field->windowManager, -1,
```

```
                    "%d is not valid. The value must be greater than 0, but less than"
                    " %d", value,smssMaxSines);
            return -1;
            }
        else
            return 0;
        }

void SmoothSumSineEditForm::dogetPrimes(void *object,UI_EVENT &event) {

        SmoothSumSineEditForm *me = (SmoothSumSineEditForm *)(((UIW_NUMBER
*)object)->parent);
        SmoothSumSineTG *mine = (SmoothSumSineTG *)me->myTraj;

        mine->getFilt();

        }

void SmoothSumSineEditForm::getPrimes(void *object,UI_EVENT &event) {

        SmoothSumSineEditForm *form = (SmoothSumSineEditForm
*)(((UIW_BUTTON *)object)->parent);
        form->dogetPrimes(object,event);
        }

int SmoothSumSineEditForm::validatephaseAngle3(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SmoothSumSineEditForm *)number->parent)-
>doValidatephaseAngle3(item, ccode));
        }

int SmoothSumSineEditForm::doValidatephaseAngle3(void *item,int ccode) {
        if (ccode == S_CURRENT)
            return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();
        SmoothSumSineEditForm *me = (SmoothSumSineEditForm *)(((UIW_NUMBER
*)item)->parent);
        SmoothSumSineTG *mine = (SmoothSumSineTG *)me->myTraj;

        if (mine->verifyPhaseAngle3()) {
            _errorSystem->ReportError(field->windowManager, -1,
                    "%f is not valid. The value must be greater than or equal to 0, "
                    "but less than 360", value);
            return -1;
            }
        else
            return 0;
        }

int SmoothSumSineEditForm::validateusableTrack3(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
```

```
        return (((SmoothSumSineEditForm *)number->parent)-
>doValidateusableTrack3(item, ccode));
        }

int SmoothSumSineEditForm::doValidateusableTrack3(void *item,int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();
        SmoothSumSineEditForm *me = (SmoothSumSineEditForm *)(((UIW_NUMBER
*)item)->parent);
        SmoothSumSineTG *mine = (SmoothSumSineTG *)me->myTraj;

        if (mine->verifyUsableTrack3()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%f is not valid. The value must be greater than %f, but less than"
                        " %f", value,0.0,getTrackLength());
                return -1;
                }
        else
                return 0;
        }

int SmoothSumSineEditForm::validateAllowedAccel(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SmoothSumSineEditForm *)number->parent)-
>doValidateAllowedAccel(item, ccode));
        }

int SmoothSumSineEditForm::doValidateAllowedAccel(void *item,int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();
        SmoothSumSineEditForm *me = (SmoothSumSineEditForm *)(((UIW_NUMBER
*)item)->parent);
        SmoothSumSineTG *mine = (SmoothSumSineTG *)me->myTraj;

        if (mine->verifyAllowedAccel()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%f is not valid. The value must be greater than %f, but less than"
                        " %f", value,0.0,getMaximumAccel());
                return -1;
                }
        else
                return 0;
        }

int SmoothSumSineEditForm::validateAllowedVel(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SmoothSumSineEditForm *)number->parent)-
>doValidateAllowedVel(item, ccode));
```

```
	}

int SmoothSumSineEditForm::doValidateAllowedVel(void *item,int ccode) {
	if (ccode == S_CURRENT)
		return (0);

	UIW_NUMBER *field = (UIW_NUMBER *)item;
	float value = *(float *)field->DataGet();
	SmoothSumSineEditForm *me = (SmoothSumSineEditForm *)(((UIW_NUMBER
*)item)->parent);
	SmoothSumSineTG *mine = (SmoothSumSineTG *)me->myTraj;

	if (mine->verifyAllowedVel()) {
		_errorSystem->ReportError(field->windowManager, -1,
			"%f is not valid. The value must be greater than %f, but less than"
			" %f", value,0.0,getMaximumVelocity());
		return -1;
		}
	else
		return 0;
	}


int SmoothSumSineEditForm::validateEqualDomain(void *item,int ccode) {
	UIW_NUMBER *number = (UIW_NUMBER *)item;
	return (((SmoothSumSineEditForm *)number->parent)-
>doValidateEqualDomain(item, ccode));
	}

int SmoothSumSineEditForm::doValidateEqualDomain(void *item,int ccode) {
	if (ccode == S_CURRENT)
		return (0);

	UIW_NUMBER *field = (UIW_NUMBER *)item;
	int value = *(int *)field->DataGet();
	SmoothSumSineEditForm *me = (SmoothSumSineEditForm *)(((UIW_NUMBER
*)item)->parent);
	SmoothSumSineTG *mine = (SmoothSumSineTG *)me->myTraj;

	if (mine->verifyEqualDomain()) {
		_errorSystem->ReportError(field->windowManager, -1,
			"%d is not valid. The value must be greater than or equal to 1, "
			"but less than  3", value);
		return -1;
		}
	else
		return 0;
	}

int SmoothSumSineEditForm::validatez13(void *item,int ccode) {
	UIW_NUMBER *number = (UIW_NUMBER *)item;
	return (((SmoothSumSineEditForm *)number->parent)->doValidatez13(item,
ccode));
	}
```

```
int SmoothSumSineEditForm::doValidatez13(void *item, int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();

        SmoothSumSineEditForm *me = (SmoothSumSineEditForm *)(((UIW_NUMBER
*)item)->parent);
        SmoothSumSineTG *mine = (SmoothSumSineTG *)me->myTraj;

        if (mine->verifyz13()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%f is not valid. The value must be greater than or equal to 0.0, "
                        "but less than or equal to 1.0", value);
                return -1;
                }
        else
                return 0;
        }


int SmoothSumSineEditForm::validatez23(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SmoothSumSineEditForm *)number->parent)->doValidatez23(item,
ccode));
        }

int SmoothSumSineEditForm::doValidatez23(void *item, int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();

        SmoothSumSineEditForm *me = (SmoothSumSineEditForm *)(((UIW_NUMBER
*)item)->parent);
        SmoothSumSineTG *mine = (SmoothSumSineTG *)me->myTraj;

        if (mine->verifyz23()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%f is not valid. The value must be greater than or equal to 0.0, "
                        "but less than or equal to 1.0", value);
                return -1;
                }
        else
                return 0;
        }

int SmoothSumSineEditForm::validatew13(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SmoothSumSineEditForm *)number->parent)->doValidatew13(item,
ccode));
```

```
        }

int SmoothSumSineEditForm::doValidatew13(void *item, int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();

        SmoothSumSineEditForm *me = (SmoothSumSineEditForm *)(((UIW_NUMBER
*)item)->parent);
        SmoothSumSineTG *mine = (SmoothSumSineTG *)me->myTraj;

        if (mine->verifyw13()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%f is not valid. The value must be greater than 0.0, "
                        "but less than or equal to 2.0 Hertz", value);
                return -1;
                }
        else
                return 0;
        }

int SmoothSumSineEditForm::validatew23(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SmoothSumSineEditForm *)number->parent)->doValidatew23(item,
ccode));
        }

int SmoothSumSineEditForm::doValidatew23(void *item, int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();

        SmoothSumSineEditForm *me = (SmoothSumSineEditForm *)(((UIW_NUMBER
*)item)->parent);
        SmoothSumSineTG *mine = (SmoothSumSineTG *)me->myTraj;

        if (mine->verifyw23()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%f is not valid. The value must be greater than 0.0, "
                        "but less than or equal to 2.0 Hertz", value);
                return -1;
                }
        else
                return 0;
        }


SmoothSumSineTG::SmoothSumSineTG() : AbstractTG() {
        strcpy(myName,"Smooth Sum Sines");
        validAxis = Sled;
```

```
        duration = 100.00;
        frequency = 1.0/duration;
        numberSines = 10;
        phaseAngle3 = 0.0;
        usableTrack3 = 15.00;
        allowedDeriv3 = 100.00;
        equalDomain = 1;
        z13=0.707;
        z23=0.707;
        w13=0.350;
        w23=0.150;
}

SmoothSumSineTG::~SmoothSumSineTG() {
        }

int SmoothSumSineTG::readHeader(const char *filename) {
        FILE *f;

        // First, read the data of our ancestor(s).
        if (AbstractTG::readHeader(filename))
                return 1;

        // Open the file for reading. Note that an existing file is assumed.
        f = fopen(filename,"rb");

        // Seek past our ancestor(s) data. Note the true data size of the our
        // ancestor(s) is two less that the size of our immediate ancestor.
        fseek(f,sizeof(AbstractTG)-2,SEEK_SET);

        // Read our portion of the header. To do this, we must find our data,
        // which is located after our ancestor. The size to read is the
        // difference between our size and that of our ancestor.
        char *ptr = (char *)this;
        ptr += sizeof(AbstractTG);
        int size = sizeof(SmoothSumSineTG)-sizeof(AbstractTG);
        fread(ptr,size,1,f);

        // Close the file
        fclose(f);
        return 0;
        }

void SmoothSumSineTG::writeHeader(const char *filename) {
        FILE *f;

        // First, write the data of our ancestor(s).
        AbstractTG::writeHeader(filename);

        // Open the file for writing. Note that an existing file is assumed.
        f = fopen(filename,"rb+");

        // Seek past our ancestor(s) data. Note the true data size of the our
        // ancestor(s) is two less that the size of our immediate ancestor.
```

```
        fseek(f,sizeof(AbstractTG)-2,SEEK_SET);

        // Write our portion of the header. To do this, we must find our data,
        // which is located after our ancestor. The size to write is the
        // difference between our size and that of our ancestor.
        char *ptr = (char *)this;
        ptr += sizeof(AbstractTG);
        int size = sizeof(SmoothSumSineTG)-sizeof(AbstractTG);
        fwrite(ptr,size,1,f);

        // Close the file
        fclose(f);
        }

int SmoothSumSineTG::verifyDuration() {
        if (duration < 0.0)
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }


int SmoothSumSineTG::verifyFrequency() {
        if (frequency < 0.0 || frequency > getMaximumFrequency())
                return 1;       // Invalid.
        else {
                frequency = 1.0/duration;
                return 0;       // Okay
                }
        }

int SmoothSumSineTG::verifyNumberSines() {
        if ((numberSines < 1) || (numberSines > smssMaxSines))
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }

int SmoothSumSineTG::verifyPhaseAngle3() {
        if (phaseAngle3 < 0.0 || phaseAngle3 > 359.999)
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }

int SmoothSumSineTG::verifyUsableTrack3() {
        if (usableTrack3 < 0.0 || usableTrack3 > getTrackLength())
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }


int SmoothSumSineTG::verifyAllowedAccel() {
```

```
        if (allowedDeriv3 < 0.0 || allowedDeriv3 > getMaximumAccel())
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }

int SmoothSumSineTG::verifyAllowedVel() {
        if (allowedDeriv3 < 0.0 || allowedDeriv3 > getMaximumVelocity())
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }


int SmoothSumSineTG::verifyEqualDomain() {
        if (equalDomain < 1 || equalDomain > 3)
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }

int SmoothSumSineTG::verifyz13() {
        if (z13 < 0.0 || z13 > 1.0)
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }

int SmoothSumSineTG::verifyz23() {
        if (z23 < 0.0 || z23 > 1.0)
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }

int SmoothSumSineTG::verifyw13() {
        if (w13 <= 0.0 || w13 > 2.0)
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }

int SmoothSumSineTG::verifyw23() {
        if (w23 <= 0.0 || w23 > 2.0)
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }

int SmoothSumSineTG::verifyPrimes(){
        int j;

        for (j=0;j<numberSines;j++)
```

```
                if ((smprimes[j] <= 0.0) || (smprimes[j] > 200.0))
                        return 1;       // Invalid.

        return 0;       // Okay

}

void SmoothSumSineTG::setDef() {

        int j;
// initialize default primes

        smprimes[0]  = 3;
        smprimes[1]  = 5;
        smprimes[2]  = 7;
        smprimes[3]  = 9;
        smprimes[4]  = 11;
        smprimes[5]  = 13;
        smprimes[6]  = 17;
        smprimes[7]  = 19;
        smprimes[8]  = 23;
        smprimes[9]  = 29;
        smprimes[10] = 31;
        smprimes[11] = 37;
        smprimes[12] = 41;
        smprimes[13] = 43;
        smprimes[14] = 47;
        smprimes[15] = 53;
        smprimes[16] = 61;
        smprimes[17] = 73;
        smprimes[18] = 83;
        smprimes[19] = 101;

        return;
}

void SmoothSumSineTG::getFilt() {

        int j;
        UIW_WINDOW *window;


        if (numberSines<=10) {
        window=new UIW_WINDOW(26,3,21,numberSines+4,WOF_NO_FLAGS);
        *window
                + new UIW_BORDER
                + new UIW_TITLE(" Primes ")
                + new UIW_SYSTEM_BUTTON;

        for (j = 0; j < numberSines; j++) {
                 *window
                        + new
UIW_NUMBER(5,1+j,9,&smprimes[j],NULL,NMF_NO_FLAGS,
```

```
        WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER);
        }
        }
        else {
        window=new UIW_WINDOW(26,3,25,14,WOF_NO_FLAGS);
        *window
                + new UIW_BORDER
                + new UIW_TITLE(" Primes ")
                + new UIW_SYSTEM_BUTTON;

        for (j = 0; j < numberSines; j++) {
                if (j<10) {
                *window
                        + new
UIW_NUMBER(2,1+j,9,&smprimes[j],NULL,NMF_NO_FLAGS,

        WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER);

                }
                else {
                *window
                        + new UIW_NUMBER(12,1+j-
10,9,&smprimes[j],NULL,NMF_NO_FLAGS,

        WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER);
                }
        }

        }

        // Give it to the window manager.

        *_windowManager + window;

        return;
}

int SmoothSumSineTG::verifyParameters() {
        int error = 0;
        int j;
        long k;
        double wh;
        double num_real,num_imag,den;
        float d_scale;
        float p_scale;
        float derivMax;

        error = verifyDuration();
        error += verifyFrequency();
        error += verifyNumberSines();
        for (j=0;j<numberSines;j++)
                error += verifyPrimes();
        error += verifyPhaseAngle3();
```

```
        error += verifyUsableTrack3();
        error += verifyAllowedVel();
        error += verifyEqualDomain();
        error += verifyz13();
        error += verifyz23();
        error += verifyw13();
        error += verifyw23();

        // Calculate our derived parameters.
        if (!error) {

                        for (j = 0; j < numberSines; j++) {
                                wh = double (smprimes[j]/duration);
                                w[j] = float (2.0*PI*wh);

                                num_real=(w13*w13-wh*wh)*(w23*w23-wh*wh)-
4*z13*z23*w13*w23*wh*wh;
                                num_imag=2*z13*w13*wh*(w23*w23-
wh*wh)+2*z23*w23*wh*(w13*w13-wh*wh);

                                den=pow((w23*w23-
wh*wh),2)+pow((2*z23*w23*wh),2);

                                amplitude[j] = float
(w23*w23/(w13*w13)*sqrt(num_real*num_real+
                                                num_imag*num_imag)/den);

                                switch (equalDomain) {
                                case edPosition2:
//                                      amplitude[j] *= w[j];       //commanding velocity
                                        amplitude[j] *= 1.0;        //commanding position
                                        break;
                                case edVelocity2:
//                                      amplitude[j] *= 1.0;        //commanding velocity
                                        amplitude[j] /= w[j];       //commanding position
                                        break;
                                case edAcceleration2:
//          amplitude[j] /= w[j];       //commanding velocity
                                        amplitude[j] /= pow(w[j],2);  //commanding
position
                                        break;
                                default:
                                        amplitude[j] *= 1.0;
                                        break;
                                }
                        }

                phi[0] = 0.0;
                zphi[0] = 0.0;
                for (j = 1; j < numberSines; j++) {
                        phi[j] = phi[j-1] + (2.0*PI*phaseAngle3)/360;
                        zphi[j] = phi[j];
                        if (phi[j] >= 2.0*PI)
                                phi[j] = phi[j] - 2.0*PI;
```

```
                    zphi[j] = phi[j];
                }

            // Any other derived parameter calculations...

            // Calculate the number of commands per phase...
            numberCommands[0] = 0;
            numberCommands[1] = (long)(commandRate/frequency + 0.5);
            numberCommands[2] = 0;

            //Find Zero Crossing of velocity signal
            float zeroCross;
            float com = 0.0;
            float previousCom = 0.0;
            scaleFactor = 1.0;

            for (k = 0; k < numberCommands[1]; k++) {
                com = generateCommand(1,k);
                if ((com > 0.0 && previousCom < 0.0) || (com < 0.0 &&
previousCom > 0.0)){
                    zeroCross = k;
                    break;
                    }
                else {
                    zeroCross = 0;
                    previousCom = com;
                  }
                }
            for (j = 0; j < numberSines; j++)  {
                zphi[j] = phi[j] + w[j]*zeroCross/commandRate;
                while (zphi[j] >= 2.0*PI){
                    zphi[j] = zphi[j] - 2.0*PI;
                    }
                }

            // Calculate scale factor...
            derivMax = 0.0;
            float posFinal = 0.0;
            float command = 0.0;
            float previousCommand = 0.0;
            float deriv = 0.0;
            float maxLeft =0.0;
            float maxRight = 0.0;

            for (k = 0; k < numberCommands[1]; k++) {
                command = generateCommand(1,k);

                // Look for maximum and minimum accelerations or velocities
                deriv = (command - previousCommand)*commandRate;
                if (fabs(deriv) >= derivMax)
                    derivMax = fabs(deriv);

                // Update the range, save the current command for next time
                // around, and increment pointer.
```

```
//                      posFinal += command/commandRate;        //commanding velocity
                        posFinal = command;                     //commanding position

                        previousCommand = command;

                        if (posFinal > maxRight)
                                maxRight = posFinal;
                        if (posFinal < maxLeft)
                                maxLeft = posFinal;
                }

                d_scale = allowedDeriv3/derivMax;
                maxRight=usableTrack3/fabs(maxRight);
                maxLeft=usableTrack3/fabs(maxLeft);
                p_scale = (maxRight<maxLeft) ? maxRight : maxLeft;
                scaleFactor = (d_scale < p_scale) ? d_scale : p_scale;

                }
        else {
                numberCommands[0] = numberCommands[1] = numberCommands[2] = 0;
                }
        return error;
        }

float SmoothSumSineTG::generateCommand(int phase,long index) {


        float command = 0.0;
        for (int j = 0; j < numberSines; j++) {
                command += scaleFactor*amplitude[j]*sin(w[j]*index/commandRate +
zphi[j]);

                }
        return command;
        }

void SmoothSumSineTG::getParametersDisplaySize(UI_REGION& size) {
        if (size.right < 45)
                size.right = 45;
        size.bottom += 5;
        }


void SmoothSumSineTG::getParameters(int rate,int modal) {

        setDef();

        commandRate = rate;

        // Create an edit form.
        SmoothSumSineEditForm *form = new
SmoothSumSineEditForm(this,rate,modal);
```

```
        *form
                + new UIW_BORDER
                + new UIW_TITLE(myName)
                + new UIW_SYSTEM_BUTTON
                + new UIW_PROMPT(2,1,"Duration (s)",WOF_NO_FLAGS)
                + new UIW_NUMBER(39,1,8,&duration,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SmoothSumSineEditForm::validateFrequency)
                + new UIW_PROMPT(2,2,"Frequency (Hz)",WOF_NO_FLAGS)
                + new UIW_NUMBER(39,2,8,&frequency,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SmoothSumSineEditForm::validateFrequency)
                + new UIW_PROMPT(2,3,"Number of Sines",WOF_NO_FLAGS)
                + new
UIW_NUMBER(39,3,8,&numberSines,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SmoothSumSineEditForm::validateNumberSines)
                + new UIW_PROMPT(2,4,"Equal Amp Domain
(1=P,2=V,3=A)",WOF_NO_FLAGS)
                + new
UIW_NUMBER(39,4,8,&equalDomain,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SmoothSumSineEditForm::validateEqualDomain)
                + new UIW_PROMPT(2,5,"Numerator Damping
Ratio",WOF_NO_FLAGS)
                + new UIW_NUMBER(39,5,8,&z13,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SmoothSumSineEditForm::validatez13)
                + new UIW_PROMPT(2,6,"Denominator Damping
Ratio",WOF_NO_FLAGS)
                + new UIW_NUMBER(39,6,8,&z23,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SmoothSumSineEditForm::validatez23)
                + new UIW_PROMPT(2,7,"Numerator Natural Frequency
(Hz)",WOF_NO_FLAGS)
                + new UIW_NUMBER(39,7,8,&w13,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SmoothSumSineEditForm::validatew13)
```

```
                + new UIW_PROMPT(2,8,"Denominator Natural Frequency
(Hz)",WOF_NO_FLAGS)
                + new UIW_NUMBER(39,8,8,&w23,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SmoothSumSineEditForm::validatew23)
                + new UIW_PROMPT(2,9,"Phase Angle (deg)",WOF_NO_FLAGS)
                + new
UIW_NUMBER(39,9,8,&phaseAngle3,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SmoothSumSineEditForm::validatephaseAngle3)
                + new UIW_PROMPT(2,10,"Max Usable Track
(deg)",WOF_NO_FLAGS)
                + new
UIW_NUMBER(39,10,8,&usableTrack3,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SmoothSumSineEditForm::validateusableTrack3)
//              + new UIW_PROMPT(2,7,"Max Ang Acc (deg/s/s)",WOF_NO_FLAGS)
//              + new
UIW_NUMBER(35,7,8,&allowedAccel,NULL,NMF_NO_FLAGS,
//
WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,
//
SumSineEditForm::validateAllowedAccel)
                + new UIW_PROMPT(2,11,"Max Ang Vel (deg/s)",WOF_NO_FLAGS)
                + new
UIW_NUMBER(39,11,8,&allowedDeriv3,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SmoothSumSineEditForm::validateAllowedVel)
                + new
UIW_BUTTON(8,13,13,"Parameters",BTF_NO_FLAGS,WOF_BORDER,

SmoothSumSineEditForm::getPrimes)
                + new
UIW_BUTTON(35,13,4,"Ok",BTF_NO_FLAGS,WOF_BORDER,

SmoothSumSineEditForm::generateFunction);

        // Give it to the window manager.
        *_windowManager + form;
        }

void SmoothSumSineTG::getHeaderDisplaySize(UI_REGION& size) {
        size.left = 3;
        size.top = 1;
        size.right = 75;
        if (numberSines<10)
```

```
            size.bottom = 12+numberSines;
        else
            size.bottom = 22;
        }

void SmoothSumSineTG::displayHeader(UIW_WINDOW *window,int& left, int& top) {

        AbstractTG::displayHeader(window,left,top);

        top ++;
        *window
            + new UIW_PROMPT(left,top,"Frequency (Hz)",WOF_NO_FLAGS)
            + new UIW_PROMPT(left+20,top,"Pos Amp (deg)",WOF_NO_FLAGS)
            + new UIW_PROMPT(left+39,top,"Vel Amp (deg/s)",WOF_NO_FLAGS)
            + new UIW_PROMPT(left+58,top,"Phase (deg)",WOF_NO_FLAGS);

        float x, x2;
        int j;

        for (j = 0; j < numberSines; j++) {
            x = w[j]/(2.0*PI);
            if (j < 10)
        *window
            + new UIW_NUMBER(left,top+1+j,7,&x,NULL,NMF_NO_FLAGS,

WOF_NON_SELECTABLE|WOF_BORDER);
            else
        *window
            + new UIW_NUMBER(left+7,top+1+j-
10,7,&x,NULL,NMF_NO_FLAGS,

WOF_NON_SELECTABLE|WOF_BORDER);
            }

        for (j = 0; j < numberSines; j++) {
            x = amplitude[j]*scaleFactor;

            if (j < 10)
        *window
            + new UIW_NUMBER(left+18,top+1+j,8,&x,NULL,NMF_NO_FLAGS,

WOF_NON_SELECTABLE|WOF_BORDER);
            else
        *window
            + new UIW_NUMBER(left+26,top+1+j-
10,8,&x,NULL,NMF_NO_FLAGS,

WOF_NON_SELECTABLE|WOF_BORDER);
            }

        for (j = 0; j < numberSines; j++) {
            x = amplitude[j]*w[j]*scaleFactor;

            if (j < 10)
```

```
        *window
                + new UIW_NUMBER(left+38,top+1+j,8,&x,NULL,NMF_NO_FLAGS,
WOF_NON_SELECTABLE|WOF_BORDER);
                else
        *window
                + new UIW_NUMBER(left+46,top+1+j-
10,8,&x,NULL,NMF_NO_FLAGS,
WOF_NON_SELECTABLE|WOF_BORDER);
                }

        for (j = 0; j < numberSines; j++) {
                x = trunc(360.0*phi[j]/(2.0*PI),3);
                if (j < 10)
        *window
                + new UIW_NUMBER(left+58,top+1+j,5,&x,NULL,NMF_NO_FLAGS,
WOF_NON_SELECTABLE|WOF_BORDER);
                else
        *window
                + new UIW_NUMBER(left+64,top+1+j-
10,5,&x,NULL,NMF_NO_FLAGS,
WOF_NON_SELECTABLE|WOF_BORDER);
                }

        top += 8;
        }

void SmoothSumSineTG::dumpHeader(char *name,FILE *f) {
        AbstractTG::dumpHeader(name,f);
        }
```

```
////////////////////////////////////////////////////////////////
// Title:          SMSSINTG.HPP
// Author:         Scott B. Stephenson
// Date:           September, 1993
// $Revision:      1.9.1.6
////////////////////////////////////////////////////////////////

#ifndef  SMSSINTG_HPP
#define  SMSSINTG_HPP

// Interface Dependencies -------------------------------------------------

#ifndef          UI_WIN_HPP
#include <ui_win.hpp>
#endif

#ifndef __STDIO_H
#include <stdio.h>
#endif

#ifndef   ABSTRAJG_HPP
#include "abstrajg.hpp"
#endif

// End Interface Dependencies ---------------------------------------------

// Implementation Dependencies --------------------------------------------

// End Implementation Dependencies ----------------------------------------

const USHORT edPosition2      = 0x0001;
const USHORT edVelocity2      = 0x0002;
const USHORT edAcceleration2        = 0x0003;

#define          smssMaxSines                  20


class SmoothSumSineTG : public AbstractTG {
public:
        SmoothSumSineTG();
        ~SmoothSumSineTG();

        AbstractTG *dup() { return new SmoothSumSineTG(); }

        int readHeader(const char *filename);
        void writeHeader(const char *filename);

        void dumpHeader(char *name,FILE *f);

        float generateCommand(int phase, long index);
        void generateTrajectory(const char *filename);
```

```
        void getParametersDisplaySize(UI_REGION& size);
        void getParameters(int rate,int modal);

        void getHeaderDisplaySize(UI_REGION& size);
        void displayHeader(UIW_WINDOW *window,int& left, int& top);

        int verifyDuration();
        int verifyFrequency();
        int verifyNumberSines();
        int verifyPhaseAngle3();
        int verifyUsableTrack3();
        int verifyAllowedAccel();
        int verifyAllowedVel();
        int verifyParameters();
        int verifyEqualDomain();
        int verifyz13();
        int verifyz23();
        int verifyw13();
        int verifyw23();
        int verifyPrimes();
        void setDef();
        void getFilt();

        int getWidth();// { return 36; }
        int getHeight();// { return 11; }

        float duration;
        float frequency;
        int numberSines;
        int equalDomain;

        float amplitude[smssMaxSines];
        float w[smssMaxSines];
        float scaleFactor;
        float phi[smssMaxSines];
        float zphi[smssMaxSines];
        };
#endif
```

```
////////////////////////////////////////////////////////////////
// Title:          SSTG.CPP
// Author:         Scott B. Stephenson
// Date:           September, 1993
// $Revision:      1.9.1.6
////////////////////////////////////////////////////////////////

// Interface Dependencies -------------------------------------------------

#ifndef   SSTG_HPP
#include "sstg.hpp"
#endif

// End Interface Dependencies ---------------------------------------------

// Implementation Dependencies --------------------------------------------

#ifndef __MATH_H
#include <math.h>
#endif

#ifndef __STDIO_H
#include <stdio.h>
#endif

#ifndef __STRING_H
#include <string.h>
#endif

#ifndef   DISPVARS_HPP
#include "dispvars.hpp"
#endif

#ifndef   LINKCONV_HPP
#include "linkconv.hpp"
#endif

#ifndef   TRAJEDIT_HPP
#include "trajedit.hpp"
#endif

// End Implementation Dependencies ----------------------------------------

// Begin global variable declaration
int primesEn[ssMaxSinesEn2];
float phaseAngle4;
float usableTrack4;
float allowedDeriv4;
// End global variable declaration

class SumSineEnEditForm : public TrajEditForm {
public:
```

```
        SumSineEnEditForm(SumSineEnTG *traj,int rate,int flag);

        static int validateDuration(void *item, int ccode);
        static int validateFrequency(void *item, int ccode);
        static int validateNumberSines(void *item, int ccode);
        static int validatephaseAngle4(void *item, int ccode);
        static int validateusableTrack4(void *item, int ccode);
        static int validateAllowedAccel(void *item, int ccode);
        static int validateAllowedVel(void *item, int ccode);
        static int validateEqualDomain(void *item, int ccode);
        static void getParms(void *item, UI_EVENT &event);

private:
        int doValidateDuration(void *item, int ccode);
        int doValidateFrequency(void *item, int ccode);
        int doValidateNumberSines(void *item, int ccode);
        int doValidatephaseAngle4(void *item, int ccode);
        int doValidateusableTrack4(void *item, int ccode);
        int doValidateAllowedAccel(void *item, int ccode);
        int doValidateAllowedVel(void *item, int ccode);
        int doValidateEqualDomain(void *item, int ccode);
        void dogetParms(void *item, UI_EVENT &event);

        };

SumSineEnEditForm::SumSineEnEditForm(SumSineEnTG *traj,int rate,int flag) :
                TrajEditForm(traj,rate,3,3,51,12,flag,0) {
        }
int SumSineEnEditForm::validateDuration(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SumSineEnEditForm *)number->parent)->doValidateDuration(item,
ccode));
        }

int SumSineEnEditForm::doValidateDuration(void *item,int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();
        SumSineEnEditForm *me = (SumSineEnEditForm *)(((UIW_NUMBER *)item)-
>parent);
        SumSineEnTG *mine = (SumSineEnTG *)me->myTraj;

        if (mine->verifyDuration()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%f is not valid. The value must be greater than %f", value,0.0);
                return -1;
                }
        else
                return 0;
        }

void SumSineEnEditForm::dogetParms(void *object,UI_EVENT &event) {
```

```
	SumSineEnEditForm *me = (SumSineEnEditForm *)(((UIW_NUMBER
*)object)->parent);
	SumSineEnTG *mine = (SumSineEnTG *)me->myTraj;

	mine->getFilt();

	}

void SumSineEnEditForm::getParms(void *object,UI_EVENT &event) {

	SumSineEnEditForm *form = (SumSineEnEditForm *)(((UIW_BUTTON
*)object)->parent);
	form->dogetParms(object,event);
	}

int SumSineEnEditForm::validateFrequency(void *item,int ccode) {
	UIW_NUMBER *number = (UIW_NUMBER *)item;
	return (((SumSineEnEditForm *)number->parent)->doValidateFrequency(item,
ccode));
	}

int SumSineEnEditForm::doValidateFrequency(void *item,int ccode) {
	if (ccode == S_CURRENT)
		return (0);

	UIW_NUMBER *field = (UIW_NUMBER *)item;
	float value = *(float *)field->DataGet();
	SumSineEnEditForm *me = (SumSineEnEditForm *)(((UIW_NUMBER *)item)-
>parent);
	SumSineEnTG *mine = (SumSineEnTG *)me->myTraj;

	if (mine->verifyFrequency()) {
		_errorSystem->ReportError(field->windowManager, -1,
			"%f is not valid.  The value must be greater than %f, but less than"
			" %f", value,0.0,getMaximumFrequency());
		return -1;
		}
	else
		return 0;
	}

int SumSineEnEditForm::validateNumberSines(void *item,int ccode) {
	UIW_NUMBER *number = (UIW_NUMBER *)item;
	return (((SumSineEnEditForm *)number->parent)->doValidateNumberSines(item,
ccode));
	}

int SumSineEnEditForm::doValidateNumberSines(void *item,int ccode) {
	if (ccode == S_CURRENT)
		return (0);

	UIW_NUMBER *field = (UIW_NUMBER *)item;
	int value = *(int *)field->DataGet();
```

```
        SumSineEnEditForm *me = (SumSineEnEditForm *)(((UIW_NUMBER *)item)-
>parent);
        SumSineEnTG *mine = (SumSineEnTG *)me->myTraj;

        if (mine->verifyNumberSines()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%d is not valid. The value must be greater than 0, but less than"
                        " %d", value,ssMaxSinesEn2);
                return -1;
                }
        else
                return 0;
        }


int SumSineEnEditForm::validatephaseAngle4(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SumSineEnEditForm *)number->parent)->doValidatephaseAngle4(item,
ccode));
        }

int SumSineEnEditForm::doValidatephaseAngle4(void *item,int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();
        SumSineEnEditForm *me = (SumSineEnEditForm *)(((UIW_NUMBER *)item)-
>parent);
        SumSineEnTG *mine = (SumSineEnTG *)me->myTraj;

        if (mine->verifyPhaseAngle4()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%f is not valid. The value must be greater than or equal to 0, "
                        "but less than 360", value);
                return -1;
                }
        else
                return 0;
        }

int SumSineEnEditForm::validateusableTrack4(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SumSineEnEditForm *)number->parent)->doValidateusableTrack4(item,
ccode));
        }

int SumSineEnEditForm::doValidateusableTrack4(void *item,int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();
```

```
        SumSineEnEditForm *me = (SumSineEnEditForm *)(((UIW_NUMBER *)item)-
>parent);
        SumSineEnTG *mine = (SumSineEnTG *)me->myTraj;

        if (mine->verifyUsableTrack4()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%f is not valid. The value must be greater than %f, but less than"
                        " %f", value,0.0,getTrackLength());
                return -1;
                }
        else
                return 0;
        }

int SumSineEnEditForm::validateAllowedAccel(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SumSineEnEditForm *)number->parent)->doValidateAllowedAccel(item,
ccode));
        }

int SumSineEnEditForm::doValidateAllowedAccel(void *item,int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();
        SumSineEnEditForm *me = (SumSineEnEditForm *)(((UIW_NUMBER *)item)-
>parent);
        SumSineEnTG *mine = (SumSineEnTG *)me->myTraj;

        if (mine->verifyAllowedAccel()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%f is not valid. The value must be greater than %f, but less than"
                        " %f", value,0.0,getMaximumAccel());
                return -1;
                }
        else
                return 0;
        }

int SumSineEnEditForm::validateAllowedVel(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SumSineEnEditForm *)number->parent)->doValidateAllowedVel(item,
ccode));
        }

int SumSineEnEditForm::doValidateAllowedVel(void *item,int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        float value = *(float *)field->DataGet();
        SumSineEnEditForm *me = (SumSineEnEditForm *)(((UIW_NUMBER *)item)-
>parent);
```

```
        SumSineEnTG *mine = (SumSineEnTG *)me->myTraj;

        if (mine->verifyAllowedVel()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%f is not valid. The value must be greater than %f, but less than"
                        " %f", value,0.0,getMaximumVelocity());
                return -1;
                }
        else
                return 0;
        }


int SumSineEnEditForm::validateEqualDomain(void *item,int ccode) {
        UIW_NUMBER *number = (UIW_NUMBER *)item;
        return (((SumSineEnEditForm *)number->parent)->doValidateEqualDomain(item,
ccode));
        }

int SumSineEnEditForm::doValidateEqualDomain(void *item,int ccode) {
        if (ccode == S_CURRENT)
                return (0);

        UIW_NUMBER *field = (UIW_NUMBER *)item;
        int value = *(int *)field->DataGet();
        SumSineEnEditForm *me = (SumSineEnEditForm *)(((UIW_NUMBER *)item)-
>parent);
        SumSineEnTG *mine = (SumSineEnTG *)me->myTraj;

        if (mine->verifyEqualDomain()) {
                _errorSystem->ReportError(field->windowManager, -1,
                        "%d is not valid. The value must be greater than or equal to 1, "
                        "but less than  3", value);
                return -1;
                }
        else
                return 0;
        }

SumSineEnTG::SumSineEnTG() : AbstractTG() {
        strcpy(myName,"Sum Sines");
        validAxis = Sled;
        duration = 100.00;
        frequency = 1.0/duration;
        numberSines = 10;
        phaseAngle4 = 0.0;
        usableTrack4 = 15.00;
        allowedDeriv4 = 100.00;
        equalDomain = 1;

        }

SumSineEnTG::~SumSineEnTG() {
        }
```

```
int SumSineEnTG::readHeader(const char *filename) {
        FILE *f;

        // First, read the data of our ancestor(s).
        if (AbstractTG::readHeader(filename))
                return 1;

        // Open the file for reading. Note that an existing file is assumed.
        f = fopen(filename,"rb");

        // Seek past our ancestor(s) data. Note the true data size of the our
        // ancestor(s) is two less that the size of our immediate ancestor.
        fseek(f,sizeof(AbstractTG)-2,SEEK_SET);

        // Read our portion of the header. To do this, we must find our data,
        // which is located after our ancestor. The size to read is the
        // difference between our size and that of our ancestor.
        char *ptr = (char *)this;
        ptr += sizeof(AbstractTG);
        int size = sizeof(SumSineEnTG)-sizeof(AbstractTG);
        fread(ptr,size,1,f);

        // Close the file
        fclose(f);
        return 0;
        }

void SumSineEnTG::writeHeader(const char *filename) {
        FILE *f;

        // First, write the data of our ancestor(s).
        AbstractTG::writeHeader(filename);

        // Open the file for writing. Note that an existing file is assumed.
        f = fopen(filename,"rb+");

        // Seek past our ancestor(s) data. Note the true data size of the our
        // ancestor(s) is two less that the size of our immediate ancestor.
        fseek(f,sizeof(AbstractTG)-2,SEEK_SET);

        // Write our portion of the header. To do this, we must find our data,
        // which is located after our ancestor. The size to write is the
        // difference between our size and that of our ancestor.
        char *ptr = (char *)this;
        ptr += sizeof(AbstractTG);
        int size = sizeof(SumSineEnTG)-sizeof(AbstractTG);
        fwrite(ptr,size,1,f);

        // Close the file
        fclose(f);
        }

int SumSineEnTG::verifyDuration() {
```

```
        if (duration < 0.0)
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }


int SumSineEnTG::verifyFrequency() {
        if (frequency < 0.0 || frequency > getMaximumFrequency())
                return 1;       // Invalid.
        else {
                frequency = 1.0/duration;
                return 0;       // Okay
                }
        }

int SumSineEnTG::verifyNumberSines() {
        if ((numberSines < 1) || (numberSines > ssMaxSinesEn2))
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }

int SumSineEnTG::verifyPhaseAngle4() {
        if (phaseAngle4 < 0.0 || phaseAngle4 > 359.999)
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }

int SumSineEnTG::verifyUsableTrack4(){
        if (usableTrack4 < 0.0 || usableTrack4 > getTrackLength())
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }


int SumSineEnTG::verifyAllowedAccel() {
        if (allowedDeriv4 < 0.0 || allowedDeriv4 > getMaximumAccel())
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }

int SumSineEnTG::verifyAllowedVel() {
        if (allowedDeriv4 < 0.0 || allowedDeriv4 > getMaximumVelocity())
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }


int SumSineEnTG::verifyEqualDomain() {
```

```
        if (equalDomain < 1 || equalDomain > 3)
                return 1;       // Invalid.
        else
                return 0;       // Okay
        }

int SumSineEnTG::verifyPrimes(){
        int j;

        for (j=0;j<numberSines;j++)

                if ((primesEn[j] <= 0.0) || (primesEn[j] > 200.0))
                        return 1;       // Invalid.

        return 0;       // Okay

}

int SumSineEnTG::verifyAmps(){
        int j;

        for (j=0;j<numberSines;j++)

                if ((amplitude[j] <= 0.0) || (amplitude[j] > 10.0))
                        return 1;       // Invalid.

        return 0;       // Okay

}

void SumSineEnTG::setDef() {

        int j;
// initialize default primes

        primesEn[0]  = 3;
        primesEn[1]  = 5;
        primesEn[2]  = 7;
        primesEn[3]  = 9;
        primesEn[4]  = 11;
        primesEn[5]  = 13;
        primesEn[6]  = 17;
        primesEn[7]  = 19;
        primesEn[8]  = 23;
        primesEn[9]  = 29;
        primesEn[10] = 31;
        primesEn[11] = 37;
        primesEn[12] = 41;
        primesEn[13] = 43;
        primesEn[14] = 47;
        primesEn[15] = 53;
        primesEn[16] = 61;
        primesEn[17] = 73;
        primesEn[18] = 83;
```

```
        primesEn[19] = 101;

// initialize default amplitudes

        for (j = 0; j < ssMaxSinesEn2; j++)
                amplitude[j] = 1.0;

        return;
}

void SumSineEnTG::getFilt() {

        int j;
        UIW_WINDOW *window;


        if (numberSines<=10) {
        window=new UIW_WINDOW(26,3,39,numberSines+6,WOF_NO_FLAGS);
        *window
                + new UIW_BORDER
                + new UIW_TITLE(" Parameters ")
                + new UIW_SYSTEM_BUTTON
                + new UIW_PROMPT(7,1,"Primes",WOF_NO_FLAGS)
                + new UIW_PROMPT(23,1,"Amplitudes",WOF_NO_FLAGS);

        for (j = 0; j < numberSines; j++) {
                 *window
                        + new
UIW_NUMBER(6,3+j,9,&primesEn[j],NULL,NMF_NO_FLAGS,

        WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER);

                 *window
                        + new
UIW_NUMBER(23,3+j,9,&amplitude[j],NULL,NMF_NO_FLAGS,

        WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER);

        }
        }
        else {
        window=new UIW_WINDOW(26,3,46,16,WOF_NO_FLAGS);
        *window
                + new UIW_BORDER
                + new UIW_TITLE(" Parameters ")
                + new UIW_SYSTEM_BUTTON
                + new UIW_PROMPT(8,1,"Primes",WOF_NO_FLAGS)
                + new UIW_PROMPT(28,1,"Amplitudes",WOF_NO_FLAGS);

        for (j = 0; j < numberSines; j++) {
                 if (j<10) {
                 *window
                        + new
UIW_NUMBER(2,3+j,9,&primesEn[j],NULL,NMF_NO_FLAGS,
```

```
        WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER);

                *window
                        + new
UIW_NUMBER(23,3+j,9,&amplitude[j],NULL,NMF_NO_FLAGS,

        WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER);
                }
                else {
                *window
                        + new UIW_NUMBER(12,3+j-
10,9,&primesEn[j],NULL,NMF_NO_FLAGS,

        WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER);

                *window
                        + new UIW_NUMBER(33,3+j-
10,9,&amplitude[j],NULL,NMF_NO_FLAGS,

        WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER);
                }
        }

        }

        // Give it to the window manager.

        *_windowManager + window;

        return;
}

int SumSineEnTG::verifyParameters() {
        int error = 0;
        int j;
        long k;
        double wh;
        double num_real,num_imag,den;
        float d_scale;
        float p_scale;
        float derivMax;

        error = verifyDuration();
        error += verifyFrequency();
        error += verifyNumberSines();
        for (j=0;j<numberSines;j++)
                error += verifyPrimes();
        for (j=0;j<numberSines;j++)
                error += verifyAmps();
        error += verifyPhaseAngle4();
        error += verifyUsableTrack4();
        error += verifyAllowedVel();
        error += verifyEqualDomain();
```

```
        // Calculate our derived parameters.
        if (!error) {

                        for (j = 0; j < numberSines; j++) {
                                w[j] = 2.0*PI*primesEn[j]/duration;
                                switch (equalDomain) {
                                case edPositionEn:
//                                      amplitude[j] *= w[j];       //commanding velocity
                                        amplitude[j] *= 1.0;        //commanding position
                                        break;
                                case edVelocityEn:
//                                      amplitude[j] *= 1.0;        //commanding velocity
                                        amplitude[j] /= w[j];       //commanding position
                                        break;
                                case edAccelerationEn:
//          amplitude[j] /= w[j];       //commanding velocity
                                        amplitude[j] /= pow(w[j],2);  //commanding
position
                                        break;
                                default:
                                        amplitude[j] *= 1.0;
                                        break;
                                }

                        }

                phi[0] = 0.0;
                zphi[0] = 0.0;
                for (j = 1; j < numberSines; j++) {
                        phi[j] = phi[j-1] + (2.0*PI*phaseAngle4)/360;
                        zphi[j] = phi[j];
                        if (phi[j] >= 2.0*PI)
                                phi[j] = phi[j] - 2.0*PI;
                                zphi[j] = phi[j];
                        }

                // Any other derived parameter calculations...

                // Calculate the number of commands per phase...
                numberCommands[0] = 0;
                numberCommands[1] = (long)(commandRate/frequency + 0.5);
                numberCommands[2] = 0;

                //Find Zero Crossing of velocity signal
                float zeroCross;
                float com = 0.0;
                float previousCom = 0.0;
                scaleFactor = 1.0;

                for (k = 0; k < numberCommands[1]; k++) {
                        com = generateCommand(1,k);
                        if ((com > 0.0 && previousCom < 0.0) || (com < 0.0 &&
previousCom > 0.0)){
```

```
                        zeroCross = k;
                        break;
                        }
                else {
                        zeroCross = 0;
                        previousCom = com;
                 }
                }
        for (j = 0; j < numberSines; j++)  {
                zphi[j] = phi[j] + w[j]*zeroCross/commandRate;
                while (zphi[j] >= 2.0*PI){
                        zphi[j] = zphi[j] - 2.0*PI;
                        }
                }

        // Calculate scale factor...
        derivMax = 0.0;
        float posFinal = 0.0;
        float command = 0.0;
        float previousCommand = 0.0;
        float deriv = 0.0;
        float maxLeft =0.0;
        float maxRight = 0.0;

        for (k = 0; k < numberCommands[1]; k++) {
                command = generateCommand(1,k);

                // Look for maximum and minimum accelerations or velocities
                deriv = (command - previousCommand)*commandRate;
                if (fabs(deriv) >= derivMax)
                        derivMax = fabs(deriv);

                // Update the range, save the current command for next time
                // around, and increment pointer.

//              posFinal += command/commandRate;        //commanding velocity
                posFinal = command;                     //commanding position

                previousCommand = command;

                if (posFinal > maxRight)
                        maxRight = posFinal;
                if (posFinal < maxLeft)
                        maxLeft = posFinal;
        }

        d_scale = allowedDeriv4/derivMax;
        maxRight=usableTrack4/fabs(maxRight);
        maxLeft=usableTrack4/fabs(maxLeft);
        p_scale = (maxRight<maxLeft) ? maxRight : maxLeft;
        scaleFactor = (d_scale < p_scale) ? d_scale : p_scale;

        }
else {
```

```
                numberCommands[0] = numberCommands[1] = numberCommands[2] = 0;
                }
        return error;
        }

float SumSineEnTG::generateCommand(int phase,long index) {


        float command = 0.0;
        for (int j = 0; j < numberSines; j++) {
                command += scaleFactor*amplitude[j]*sin(w[j]*index/commandRate +
zphi[j]);

                }
        return command;
        }

void SumSineEnTG::getParametersDisplaySize(UI_REGION& size) {
        if (size.right < 45)
                size.right = 45;
        size.bottom += 5;
        }


void SumSineEnTG::getParameters(int rate,int modal) {

        setDef();

        commandRate = rate;

        // Create an edit form.
        SumSineEnEditForm *form = new SumSineEnEditForm(this,rate,modal);


        *form
                + new UIW_BORDER
                + new UIW_TITLE(myName)
                + new UIW_SYSTEM_BUTTON
                + new UIW_PROMPT(2,1,"Duration (s)",WOF_NO_FLAGS)
                + new UIW_NUMBER(39,1,8,&duration,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SumSineEnEditForm::validateFrequency)
                + new UIW_PROMPT(2,2,"Frequency (Hz)",WOF_NO_FLAGS)
                + new UIW_NUMBER(39,2,8,&frequency,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SumSineEnEditForm::validateFrequency)
                + new UIW_PROMPT(2,3,"Number of Sines",WOF_NO_FLAGS)
                + new
UIW_NUMBER(39,3,8,&numberSines,NULL,NMF_NO_FLAGS,
```

```
WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SumSineEnEditForm::validateNumberSines)
            + new UIW_PROMPT(2,4,"Equal Amp Domain
(1=P,2=V,3=A)",WOF_NO_FLAGS)
            + new
UIW_NUMBER(39,4,8,&equalDomain,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SumSineEnEditForm::validateEqualDomain)
            + new UIW_PROMPT(2,5,"Phase Angle (deg)",WOF_NO_FLAGS)
            + new
UIW_NUMBER(39,5,8,&phaseAngle4,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SumSineEnEditForm::validatephaseAngle4)
            + new UIW_PROMPT(2,6,"Max Usable Track (deg)",WOF_NO_FLAGS)
            + new
UIW_NUMBER(39,6,8,&usableTrack4,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SumSineEnEditForm::validateusableTrack4)
            + new UIW_PROMPT(2,7,"Max Ang Vel (deg/s)",WOF_NO_FLAGS)
            + new
UIW_NUMBER(39,7,8,&allowedDeriv4,NULL,NMF_NO_FLAGS,

WOF_AUTO_CLEAR|WOF_NO_ALLOCATE_DATA|WOF_BORDER,

SumSineEnEditForm::validateAllowedVel)
            + new
UIW_BUTTON(8,9,12,"Parameters",BTF_NO_FLAGS,WOF_BORDER,
                                                SumSineEnEditForm::getParms)
            + new UIW_BUTTON(35,9,4,"Ok",BTF_NO_FLAGS,WOF_BORDER,

SumSineEnEditForm::generateFunction);

        // Give it to the window manager.
        *_windowManager + form;
        }

void SumSineEnTG::getHeaderDisplaySize(UI_REGION& size) {
        size.left = 3;
        size.top = 1;
        size.right = 75;
        if (numberSines<10)
                size.bottom = 12+numberSines;
        else
                size.bottom = 22;
        }
```

```
void SumSineEnTG::displayHeader(UIW_WINDOW *window,int& left, int& top) {

        AbstractTG::displayHeader(window,left,top);

        top ++;
        *window
                + new UIW_PROMPT(left,top,"Frequency (Hz)",WOF_NO_FLAGS)
                + new UIW_PROMPT(left+20,top,"Pos Amp (deg)",WOF_NO_FLAGS)
                + new UIW_PROMPT(left+39,top,"Vel Amp (deg/s)",WOF_NO_FLAGS)
                + new UIW_PROMPT(left+58,top,"Phase (deg)",WOF_NO_FLAGS);

        float x, x2;
        int j;

        for (j = 0; j < numberSines; j++) {
        x = w[j]/(2.0*PI);
                if (j < 10)
        *window
                + new UIW_NUMBER(left,top+1+j,7,&x,NULL,NMF_NO_FLAGS,

WOF_NON_SELECTABLE|WOF_BORDER);
                else
        *window
                + new UIW_NUMBER(left+7,top+1+j-
10,7,&x,NULL,NMF_NO_FLAGS,

WOF_NON_SELECTABLE|WOF_BORDER);
                }

        for (j = 0; j < numberSines; j++) {
        x = amplitude[j]*scaleFactor;

                if (j < 10)
        *window
                + new UIW_NUMBER(left+18,top+1+j,8,&x,NULL,NMF_NO_FLAGS,

WOF_NON_SELECTABLE|WOF_BORDER);
                else
        *window
                + new UIW_NUMBER(left+26,top+1+j-
10,8,&x,NULL,NMF_NO_FLAGS,

WOF_NON_SELECTABLE|WOF_BORDER);
                }

        for (j = 0; j < numberSines; j++) {
        x = amplitude[j]*w[j]*scaleFactor;

                if (j < 10)
        *window
                + new UIW_NUMBER(left+38,top+1+j,8,&x,NULL,NMF_NO_FLAGS,

WOF_NON_SELECTABLE|WOF_BORDER);
                else
```

```
        *window
                + new UIW_NUMBER(left+46,top+1+j-
10,8,&x,NULL,NMF_NO_FLAGS,

WOF_NON_SELECTABLE|WOF_BORDER);
                }

        for (j = 0; j < numberSines; j++) {
        x = trunc(360.0*phi[j]/(2.0*PI),3);
                if (j < 10)
        *window
                + new UIW_NUMBER(left+58,top+1+j,5,&x,NULL,NMF_NO_FLAGS,

WOF_NON_SELECTABLE|WOF_BORDER);
                else
        *window
                + new UIW_NUMBER(left+64,top+1+j-
10,5,&x,NULL,NMF_NO_FLAGS,

WOF_NON_SELECTABLE|WOF_BORDER);
                }

        }

void SumSineEnTG::dumpHeader(char *name,FILE *f) {
        AbstractTG::dumpHeader(name,f);
        }
```

```
////////////////////////////////////////////////////////////////
// Title:          SSTG.HPP
// Author:         Scott B. Stephenson
// Date:           September, 1993
// $Revision:   1.9.1.6
////////////////////////////////////////////////////////////////

#ifndef  SSTG_HPP
#define  SSTG_HPP

// Interface Dependencies -------------------------------------------------

#ifndef          UI_WIN_HPP
#include <ui_win.hpp>
#endif

#ifndef __STDIO_H
#include <stdio.h>
#endif

#ifndef   ABSTRAJG_HPP
#include "abstrajg.hpp"
#endif

// End Interface Dependencies ---------------------------------------------

// Implementation Dependencies --------------------------------------------

// End Implementation Dependencies ----------------------------------------

const USHORT edPositionEn      = 0x0001;
const USHORT edVelocityEn      = 0x0002;
const USHORT edAccelerationEn  = 0x0003;

#define          ssMaxSinesEn2                20


class SumSineEnTG : public AbstractTG {
public:
        SumSineEnTG();
        ~SumSineEnTG();

        AbstractTG *dup() { return new SumSineEnTG(); }

        int readHeader(const char *filename);
        void writeHeader(const char *filename);

        void dumpHeader(char *name,FILE *f);

        float generateCommand(int phase, long index);
        void generateTrajectory(const char *filename);
```

```
        void getParametersDisplaySize(UI_REGION& size);
        void getParameters(int rate,int modal);

        void getHeaderDisplaySize(UI_REGION& size);
        void displayHeader(UIW_WINDOW *window,int& left, int& top);

        int verifyDuration();
        int verifyFrequency();
        int verifyNumberSines();
        int verifyPhaseAngle4();
        int verifyUsableTrack4();
        int verifyAllowedAccel();
        int verifyAllowedVel();
        int verifyParameters();
        int verifyEqualDomain();
        int verifyPrimes();
        int verifyAmps();
        void getFilt();
        void setDef();


        int getWidth();// { return 36; }
        int getHeight();// { return 11; }

        float duration;
        float frequency;
        int numberSines;
        int equalDomain;

        float amplitude[ssMaxSinesEn2];
        float w[ssMaxSinesEn2];
        float scaleFactor;
        float phi[ssMaxSinesEn2];
        float zphi[ssMaxSinesEn2];
        };
#endif
```

APPENDIX B

MATLAB M-FILE LISTINGS FOR DATA ANALYSIS FOR THE LINK MANUAL ROLL STABILIZATION EXPERIMENT

ave

cldf

cldf_anal

errorbar

freq_anal

main

model_err

model_fit

oldf

oldf_anal

overlay

phase_correct

psd

raw

rms

spm

B.1. Introduction

This appendix contains program listings of all the routines used to analyze and plot the data for the Link manual roll stabilization experiments. Similar routines were used to analyze the data in the sled experiments, with two minor changes:

1. Conversion factors were replaced with the correct values for the sled.

2. The code for processing trials with the CV visual fields was removed.

Since the files are identical except for these changes, listings are not included in this appendix (in the interest of saving trees).

```
function ave(sample_freq,run_time,sub_code,tc)
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       AVE computes the mean of position, velocity, or control wheel
%       for each visual field condition for one or all subjects and
%       displays it on a plot complete with errorbars.
%Routines called:
%       none
%Parameters:
%       sample_freq is the sampling frequency
%       run_time is the length of the run (in seconds)
%       sub_code is the subject code
%       tc is the number of subjects to be overlayed (1 or 6)

dark=[0,9];
con=[4,11];
cvl=[1];
cvr=[7];
fix=[3,6];
ss=[2,5,8,10];

ave_type=menu('Mean of?','Link Position','Link Velocity','Control Wheel');

if (ave_type==1),
   file_ext=['.linkpos'];
   plot_title=[plot_title,'Mean Position : '];
   y_label=['Degrees'];
elseif (ave_type==2),
   file_ext=['.linkvel'];
   plot_title=[plot_title,'Mean Velocity : '];
   y_label=['Degrees/s'];
elseif (ave_type==3),
   file_ext=['.joystick'];
   plot_title=[plot_title,'Mean Control Wheel : '];
   y_label=['Degrees'];
end;

row=[1 1 1 1 1 1]';

for sub_num=1:tc,

   col=1;

   for plot_1=1:6,

      if (plot_1==1),
         vec=fix;
      elseif (plot_1==2),
         vec=dark;
```

```
      elseif (plot_1==3),
         vec=ss;
      elseif (plot_1==4),
         vec=cvl;
      elseif (plot_1==5),
         vec=cvr;
      elseif (plot_1==6),
         vec=con;
      end;

      for rn_1=1:length(vec),

         if vec(rn_1)<10,
              file=['load ',sub_code(sub_num,:),'0',...
sprintf('%g',vec(rn_1)),file_ext];
         else,
              file=['load ',sub_code(sub_num,:),...
sprintf('%g',vec(rn_1)),file_ext];
         end;

         chdir SS4:scott:Thesis5ConvertedData:link;
         eval(file);
         chdir SS4:scott:Thesis5analysis:link;

         if (ave_type==1),
            y=14.0/2048*pos;
            clear pos;
         elseif (ave_type==2),
            y=-1.0*10/2048/0.715.*tach;
            clear tach;
         elseif (ave_type==3),
            y=-1.0*14/2048*joy;
            clear joy;
         elseif (ave_type==4),
            y=-1.0*10/2048/0.2887.*trig;
         end;

         ave(row(col),col)=mean(y);

         row(col)=row(col)+1;

      end;

      col=col+1;

   end;

end;


a=tc*length(fix);
b=tc*length(dark);
c=tc*length(ss);
d=tc*length(cvl);
```

```
e=tc*length(cvr);
f=tc*length(con);

aa=ave(1:a,1);
bb=ave(1:b,2);
cc=ave(1:c,3);
dd=ave(1:d,4);
ee=ave(1:e,5);
ff=ave(1:f,6);

keyboard;

e1=[std(aa) std(bb) std(cc) std(dd) std(ee) std(ff)];

clear ave;
ave=[mean(aa) mean(bb) mean(cc) mean(dd) mean(ee) mean(ff)];

hold off;
clg;
axis('square');

if tc==1,
   e1=[0 0 e1(3)/sqrt(4) 0 0 0];
    errorbar([0 1 2 3 4 5],ave,e1,'plot',"'wx'");
elseif tc>1,
   a=sqrt(a);
   b=sqrt(b);
   c=sqrt(c);
   d=sqrt(d);
   e=sqrt(e);
   f=sqrt(f);
   e1=[e1(1)/a e1(2)/b e1(3)/c e1(4)/d e1(5)/e e1(6)/f];
    errorbar([0 1 2 3 4 5],ave,e1,'plot',"'wx'");
end;

grid; hold on;

if tc>1,
    errorbar([0 1 2 3 4 5],ave,e1,'plot',"'wx'");
else,
    errorbar([0 1 2 3 4 5],ave,e1,'plot',"'wx'");
end;

hold off;

xlabel('Visual Field Code');
ylabel(y_label);

xstr=['0=FIX  1=DARK  2=SS  3=CVL  4=CVR  5=CON'];

if tc==1,
   plot_title=[sub_code,' : ',plot_title];
end;
```

```
title([plot_title,xstr]);
axis('normal');
return;
```

```
function cldf(sample_freq,run_time,sub_code,tc)
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       CLDF plots the mean operator describing function (and lead-lag fit with pure
%       delay if desired) for one or all subjects for a given visual field condition complete
%       with one standard deviation errorbars.
%Routines called:
%       cldf_anal
%Parameters:
%       sample_freq is the sampling frequency
%       run_time is the length of the run (in seconds)
%       sub_code is the subject code
%       tc is the number of subjects to be overlayed (1 or 6)

d=[3,5,11,17,23,31,41,53,71,89,109,137]/run_time; % vestibular frequencies

dark=[0,9];
con=[4,11];
cvl=[1];
cvr=[7];
fix=[3,6];
ss=[2,5,8,10];

pt=menu('Visual field condition?','DARK','FIX','CON','CVL','CVR','SS');

if tc==1,
   plot_title1=[sub_code,': Closed-Loop Describing Function: '];
else,
   plot_title1=['Closed-Loop Describing Function: '];
end;

if pt==1,
    vft=['DARK'];
   vftarr=dark;
elseif pt==2,
    vft=['FIX '];
   vftarr=fix;
elseif pt==3,
    vft=['CON '];
   vftarr=con;
elseif pt==4,
    vft=['CVL '];
   vftarr=cvl;
elseif pt==5,
    vft=['CVR '];
   vftarr=cvr;
elseif pt==6,
    vft=['SS  '];
   vftarr=ss;
```

```
end;

plot_title1=[plot_title1,vft];

mo=menu('Model order?','Lead-Lag+Delay','Mean, no fit','All Data, no fit');

if mo==1,

   plot_title1=['Lead-Lag+Delay: ',plot_title1];

   [ves,kf,tau1f,tau2f,taudf]=cldf_anal(sub_code,vftarr,tc,d,sample_freq,mo);

   m1=log10(d(1)*2*pi);
   m2=log10(d(12)*2*pi);
   w=logspace(m1,m2,500)/2/pi;
   numf=kf*[tau1f 1];
   denf=[tau2f 1];

   [lmf,lpf]=bode(numf,denf,2*pi*w);
   lpf=lpf-360.0*taudf*w';                  % add in the delay

    stf1=['k    = ',sprintf('%g',kf)];
    stf2=['tau1  = ',sprintf('%g',tau1f),' s'];
    stf3=['tau2  = ',sprintf('%g',tau2f),' s'];
    stf4=['taud  = ',sprintf('%g',taudf),' s'];

elseif mo==2,

   [ves,dum1,dum2,dum3,dum4]=cldf_anal(sub_code,vftarr,tc,d,sample_freq,mo);
   clear dum1; clear dum2; clear dum3; clear dum4;

elseif mo==3,

   [ves,dum1,dum2,dum3,dum4]=cldf_anal(sub_code,vftarr,tc,d,sample_freq,mo);
   clear dum1; clear dum2; clear dum3; clear dum4;

end;

[n,m]=size(ves);
if n>1,
   p_act=mean(ves);
else,
   p_act=ves;
end;
gain=abs(ves);
phase=180/pi*phase_correct(angle(ves));

ga=abs(p_act);                          %compute mean gain
ph=180.0/pi*phase_correct(angle(p_act));      %compute mean phase

clg; hold off;

if mo==3,
    subplot(211);semilogx(d,20*log10(gain),'wx'); grid;
```

```
   subplot(212);semilogx(d,phase,'wx'); grid;
else,
   if n>2,
      egain=std(20*log10(gain));
      ephase=std(phase);

      if mo==1,
         subplot(211);
           errorbar(d,20*log10(ga),egain,'semilogx','"wx"'); grid;
          hold on; semilogx(w,20*log10(lmf),'w');
           errorbar(d,20*log10(ga),egain,'semilogx','"wx"'); hold off;
           subplot(212); errorbar(d,ph,ephase,'semilogx','"wx"'); grid;
          hold on; subplot(212); semilogx(w,lpf,'w');
           errorbar(d,ph,ephase,'semilogx','"wx"'); hold off;
      else,
         subplot(211);
           errorbar(d,20*log10(ga),egain,'semilogx','"wx"'); grid;
           subplot(212); errorbar(d,ph,ephase,'semilogx','"wx"'); grid;
      end;
   else,
      if mo==1,
        subplot(211);
         axis([-2,0,-20,20]); semilogx(d,20*log10(ga),'wx'); grid;
         hold on; semilogx(w,20*log10(lmf),'w'); hold off;
          subplot(212); axis([-2,0,-200,0]); semilogx(d,ph,'wx'); grid;
          hold on; semilogx(w,lpf,'w'); hold off;
      else,
         subplot(211);
          axis([-2,0,-20,20]); semilogx(d,20*log10(ga),'wx'); grid;
         subplot(212);
          axis([-2,0,-200,0]); semilogx(d,ph,'wx'); grid;
      end;
   end;
end;

hold on;

subplot(211);
title(plot_title1);
xlabel('Frequency (Hz)');
ylabel('Decibels');

subplot(212);
xlabel('Frequency (Hz)');
ylabel('Degrees');

if mo==1,
   v=axis;
   axis;

   inc=(abs(v(3))+abs(v(4)))/10.0;
   base=(v(3)+v(4))/2+4*inc;

   text(.4,base,stf1);
```

```
   text(.4,base-inc,stf2);
   text(.4,base-2*inc,stf3);
   text(.4,base-3*inc,stf4);
end;

hold off;

return;
```

```
function [ves,c1,c2,c3,c4]=cldf_anal(sub_code,vftarr,tc,d,sample_freq,mo)
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       CLDF_ANAL loads the describing function data generated by freq_anal
%       for one or all subjects for a given visual field and passes back the
%       the data at the vestibular frequencies, as well as the parameters for
%       a lead-lag with pure delay transfer function fit to the mean data (if
%       requested).
%Routines called:
%       model_fit
%Parameters:
%       sub_code is the subject code
%       vftarr contains the run numbers for the particular visual field
%       tc is the number of subject to be overlayed (1 or 6)
%       d is the frequencies in the disturbance (vestibular)
%       sample_freq is the sampling frequency
%       mo is a flag:
%               mo == 1 causes a transfer function fit to be done
%               mo <> 1 causes the describing function data to be passed
%                       back with no fit done.

ves=[];

cc=1;
rownum=1;

for j=1:tc,
   for kk=1:length(vftarr),
      if vftarr(kk)<10,
           file=['load ',sub_code(j,:),'0',sprintf('%g',vftarr(kk)),...
'.ves'];
      else,
          file=['load ',sub_code(j,:),sprintf('%g',vftarr(kk)),'.ves'];
      end;
      chdir SS4:scott:Thesis5ConvertedData:link;
      eval(file);
      chdir SS4:scott:Thesis5analysis:link;

      ves(rownum,:)=p_act.'; %non-conjugate transpose!
      rownum=rownum+1;
   end;
end;

if mo==1,
   ga=abs(ves);
   ph=180.0/pi*phase_correct(angle(ves));
   [n,m]=size(ves);
   if n==1,
      e=[ones(ga)' ones(ph)'];
```

```
      [c1 c2 c3 c4]=model_fit(ves,e,d);
   else,
      e=[std(20*log10(ga))' std(ph)'];
      [c1 c2 c3 c4]=model_fit(mean(ves),e,d);
   end;
end;

return;
```

```
function errorbar(x, y, std,plot_type,plot_code)
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       ERRORBAR generates a plot of vector X versus vector Y complete
%       with errorbars. E is a vector the same length as X and Y that
%       specifies the lengths of the error bars. The error bars are drawn a
%       a distance of E(i) above and below the points in (X,Y) so that each
%       bar is 2*E(i) long.
%Routines called:
%       none
%Parameters:
%       plot_type is a string specifying the type of plot to be made and
%               can be one of the following: 'semilogx','semilogy','loglog','plot'.
%       plot_code is a string specifying the point type and color and can be
%               one of the following: '"r."', '"r+"','"r*"','"ro"','"rx"'
%               where r can be any allowed color such as g,b,w,i,etc.

%       For example,
%
%               x = 1:10;
%               y = sin(x);
%               e = std(y)*ones(x);
%               errorbar(x,y,e,'plot','"rx"');
%
%       Draws error bars of unit standard deviation on linear axes with points
%       shown as red x's

npt = max(size(x));
if nargin == 2
          std = y;
          y = x;
          x(:) = 1:npt;
end;
if length(plot_type)>4,
  tee=0.0; %no tees because we have log scale on x axis
else,
  tee = (max(x)-min(x))/100;          % make tee .02 x-distance for error bars
end;

xl = x - tee;
%xl(1)=xl(1)+tee;
xr = x + tee;
%xr(length(xr))=xr(length(xr))-tee;

ytop = y + std;
ybot = y - std;
x1=[xl'  xr'];
x2=[x'   x'];
y1=[ytop'  ytop'];
```

```
y2=[ybot' ybot']';
y3=[ytop' ybot']';

com=[plot_type,'(x1,  y1,"w-",x1,y2,"w-",x2,y3,"w-")'];
eval(com);
hold on;
com=[plot_type,'(x, y, ',plot_code,')'];
eval(com);
hold off;

return;
```

```
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       FREQ_ANAL computes the describing function -lamda/phi at each of the
%       12 frequencies in the input disturbance. The results are written to files
%       for use by the analysis routines cldf and oldf.
%Routines called:
%       none
%Parameters:
%       none

clear;

num_runs=12;        %number of runs per subject

ss_runs=[2,5,8,10];

sample_freq=40.0;   %sampling frequency
run_time=204.8;     %length of run
K=1.0;              %hand position to joystick command gain

d1=[3,5,11,17,23,31,41,53,71,89,109,137]/run_time; % vestibular frequencies
d2=[4,7,13,19,29,37,43,61,83,97,127,151]/run_time; % visual frequencies

num_fft_pts=8192;

sub_code = input('Subject Code: (2 characters) ','s');
co=1;

for j=0:num_runs-1,

if j<10,
    file=['load ',sub_code,'0',sprintf('%g',j),'.joystick'];
else,
    file=['load ',sub_code,sprintf('%g',j),'.joystick'];
end;

chdir SLS_HD:scott:thesis5converteddata:link;
eval(file);

if j<10,
    file=['load ',sub_code,'0',sprintf('%g',j),'.linkpos'];
else,
    file=['load ',sub_code,sprintf('%g',j),'.linkpos'];
end;

eval(file);
chdir SLS_HD:scott:thesis5analysis:link;

v1=-1.0*14.0/2048*joy; % joystick command in deg
```

```
clear joy;

v2=14.0/2048*pos;   % Link position in deg
clear pos;

y1=fft(v1,num_fft_pts);
y2=fft(v2,num_fft_pts);

tf1=-1.0*K*y1./y2;

if j==ss_runs(co),
   if j<10,
        file=['load ',sub_code,'0',sprintf('%g',j),'.projvel'];
      chdir SLS_HD:scott:thesis5converteddata:link;
      eval(file);
      chdir SLS_HD:scott:thesis5analysis:link;
   else,
        file=['load ',sub_code,sprintf('%g',j),'.projvel'];
      chdir SLS_HD:scott:thesis5converteddata:link;
      eval(file);
      chdir SLS_HD:scott:thesis5analysis:link;
   end;

   trig=-1.0*10/2048/0.2887*trig;

   pf1=0.0;
   v3 = 0.0*ones(trig);

   for i=1:length(trig),
      pf1=pf1+trig(i)/sample_freq;
      v3(i)=pf1;
   end;

   clear trig;

   y3=fft(v3,num_fft_pts);
    tf2=-1.0*K*y1./y3;

end;

f = sample_freq*(0:num_fft_pts/2-1)/num_fft_pts;

for zz=1:length(d1),
   ff=find(f>=d1(zz));
   ind=ff(1);
   if f(ind)==d1(zz),
      p_act(zz,1)=tf1(ind);
   else,
      p_act(zz,1)=0.0;
   end;
end;

if j<10,
     file=['save ',sub_code,'0',sprintf('%g',j),'.ves',' p_act'];
```

```
else,
    file=['save ',sub_code,sprintf('%g',j),'.ves',' p_act'];
end;

chdir SLS_HD:scott:thesis5converteddata:link;
eval(file);
chdir SLS_HD:scott:thesis5analysis:link;

clear p_act;

if j==ss_runs(co),
  for zz=1:length(d2),
     ff=find(f>=d2(zz));
     ind=ff(1);
     if f(ind)==d2(zz),
        p_act(zz,1)=tf2(ind);
     else,
        p_act(zz,1)=0.0;
     end;
  end;

  if co<length(ss_runs),
     co=co+1;
  end;

  if j<10,
       filen=['save ',sub_code,'0',sprintf('%g',j),'.vis',' p_act'];
  else,
      filen=['save ',sub_code,sprintf('%g',j),'.vis',' p_act'];
  end;

  chdir SLS_HD:scott:thesis5converteddata:link;
  eval(file);
  chdir SLS_HD:scott:thesis5analysis:link;
end;

clear p;

end;
```

```
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       Link MAIN is the driver which calls the appropriate data analysis routines
%       chosen by the user. This is the script which should be executed to start a
%       data analysis session.
%Routines called:
%       raw, ave, rms, overlay, psd, spm, cldf, oldf
%Parameters:
%       none

clear;

sample_freq=40.0;   %sampling frequency
run_time=204.8;     %length of run
num_subjects=6;     %number of subjects in study
num_fft_pts=8192;   %number of data points in Fast Fourier Transform

mt=menu('Display plot for','Single Subject','All Subjects');

if mt==1,
   sub_code = input('Subject Code: (2 characters) ','s');
   tc=1;
else,
     sub_code=['a1';'b1';'c1';'d1';'e1';'u1'];
   tc=num_subjects;
end;

another=2;

while another>1,

   chdir SS4:scott:Thesis5analysis:link;

   if mt==1,
        choice=menu('Plot type:','Raw Data','Mean','RMS','Overlay',...
'PSD','SPM','CLDF','OLDF');

      if choice==1,
         raw(sample_freq,run_time,sub_code,tc);
      elseif choice==2,
         ave(sample_freq,run_time,sub_code,tc);
      elseif choice==3,
         rms(sample_freq,run_time,sub_code,tc);
      elseif choice==4,
         overlay(sample_freq,run_time,num_fft_pts,sub_code,tc);
      elseif choice==5,
         psd(sample_freq,run_time,num_fft_pts,sub_code,tc);
      elseif choice==6,
         spm(sample_freq,run_time,num_fft_pts,sub_code,tc);
```

```
      elseif choice==7,
         cldf(sample_freq,run_time,sub_code,tc);
      elseif choice==8,
         oldf(sample_freq,run_time,sub_code,tc);
      end;

      chdir SS4:scott:Thesis5analysis:link;

   else,
         choice=menu('Plot type:','Mean','RMS','PSD','SPM','CLDF',...
'OLDF');

      if choice==1,
         ave(sample_freq,run_time,sub_code,tc);
      elseif choice==2,
         rms(sample_freq,run_time,sub_code,tc);
      elseif choice==3,
         psd(sample_freq,run_time,num_fft_pts,sub_code,tc);
      elseif choice==4,
         spm(sample_freq,run_time,num_fft_pts,sub_code,tc);
      elseif choice==5,
         cldf(sample_freq,run_time,sub_code,tc);
      elseif choice==6,
         oldf(sample_freq,run_time,sub_code,tc);
      end;

      chdir SS4:scott:Thesis5analysis:link;

   end;

    another=menu('Analyze another?','No','Yes');

   if another==2,
        sub=menu('Subject','New','Current');
      if sub==1,
         main;
      end;
   end;

end;
```

```
function [f,g] = model_err(model_parms,p_act,e,freqs,norm_parms)
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       MODEL_ERR computes the squared-error between the actual describing
%       function data in the frequency domain and the current lead-lag fit. The routine
%       is called by the MatLab routine optim, which attempts to minimize this squared-
%       error.
%Routines called:
%       none
%Parameters:
%       model_parms contains the unnormalized current parameters
%       p_act contains the actual frequency domain describing function
%               data (complex)
%       e is a vector containing the weights for the fit
%       freqs are the frequencies where the fit is to be performed
%       norm_parms are the normalized transfer function parameters

model_parms = model_parms .* norm_parms;

k   = model_parms(1);
tau1 = model_parms(2);
tau2 = model_parms(3);
taud = model_parms(4);
num  = k*[tau1 1];
den  = [tau2 1];

lag  = 360.0*taud*freqs';          %compute the delay in degrees

[lm,lp]=bode(num,den,2*pi*freqs);
lp=lp-lag;                         %add in the delay

ga=abs(p_act)';
ph=180.0/pi*phase_correct(angle(p_act))';

%compute the squared-error

err1 = (20*log10(lm)-20*log10(ga))./e(:,1);
err2 = (lp-ph)./e(:,2);

se = err1 .* err1 + err2 .* err2;

%plot(se,'wx')

f = sum(se);

fprintf('SE = %f\n',f);

%hold off; clg;
%subplot(211),semilogx(freqs,20*log10(ga),'wx',freqs,20*log10(lm),'w'),grid;
```

```
%subplot(212),semilogx(freqs,ph,'wx',freqs,lp,'w'),grid;

%
% dummy value which 'constr' requires but is unused for our
% purposes;  this must be some constant value for our purposes
%
g = -1;

return;
```

```
function [K,tau1,tau2,taud]=model_fit(p_act,e,d)
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       MODEL_FIT specifies the initial guesses for the lead-lag fit
%       parameters K, tau1, tau2, and taud and initiates the fitting process.
%Routines called:
%       none
%Parameters:
%       p_act contains the actual frequency domain describing function
%                data (complex)
%       e is a vector containing the weights for the fit
%       d are the frequencies where the fit is performed

K   = 5.00;      %initial dc gain
tau1 =  1.00;     %numerator time constant in seconds
tau2 =  1.00;     %denominator time constant in seconds
taud =  0.01;     %initial time delay in seconds

norm_parms  = [K; tau1 ; tau2 ; taud];
model_parms = [1; 1 ; 1 ; 1];

vlb =    [  0.10 ;     0.01  ;    0.01  ;   0.10];     %lower bounds
vub =    [  4.00 ;   100.00  ;  100.00  ;  100.00];     %upper bounds

options = [ 0.00 ;   0.001   ; 0.001];              %error tolerances

[model_parms, options] = constr('model_err', model_parms, options, vlb, vub, [],  p_act
, e, d , norm_parms);

model_parms = model_parms .* norm_parms;

fprintf('Number of iterations = %5.0f\n',options(10));
fprintf('Squared-error = %7.4f\n',options(8));

K=model_parms(1);
tau1=model_parms(2);
tau2=model_parms(3);
taud=model_parms(4);

fprintf('K = %f\n',K);          %dc gain
fprintf('tau1 = %f s\n',tau1);   %numerator time constant in seconds
fprintf('tau2 = %f s\n',tau2);   %denominator time constant in seconds
fprintf('taud = %f s\n',taud);   %delay time in seconds

return;
```

```
function oldf(sample_freq,run_time,sub_code,tc)
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       OLDF plots the open-loop mean describing function (and lead-lag fit
%       with pure delay if desired) for one or all subjects for a given visual field
%       condition complete with one standard deviation errorbars.
%Routines called:
%       oldf_anal
%Parameters:
%       sample_freq is the sampling frequency
%       run_time is the length of the run (in seconds)
%       sub_code is the subject code
%       tc is the number of subjects to be overlayed (1 or 6)

d=[3,5,11,17,23,31,41,53,71,89,109,137]/run_time; % vestibular frequencies

dark=[0,9];
con=[4,11];
cvl=[1];
cvr=[7];
fix=[3,6];
ss=[2,5,8,10];

pt=menu('Visual field condition?','DARK','FIX','CON','CVL','CVR','SS');

if tc==1,
   plot_title1=[sub_code,': Open-Loop Describing Function: '];
else,
   plot_title1=['Open-Loop Describing Function: '];
end;

if pt==1,
    vft=['DARK'];
   vftarr=dark;
elseif pt==2,
    vft=['FIX '];
   vftarr=fix;
elseif pt==3,
    vft=['CON '];
   vftarr=con;
elseif pt==4,
    vft=['CVL '];
   vftarr=cvl;
elseif pt==5,
    vft=['CVR '];
   vftarr=cvr;
elseif pt==6,
    vft=['SS  '];
   vftarr=ss;
```

```
end;

plot_title1=[plot_title1,vft];

mo=menu('Model order?','Lead-Lag+Delay','Mean, no fit','All Data, no fit');

if mo==1,

   plot_title1=['Lead-Lag+Delay: ',plot_title1];

   [ves,kf,tau1f,tau2f,taudf]=oldf_anal(sub_code,vftarr,tc,d,sample_freq,mo);

   m1=log10(d(1)*2*pi);
   m2=log10(d(12)*2*pi);
   w=logspace(m1,m2,500)/2/pi;
   numf=kf*[tau1f 1];
   denf=[tau2f 1];

   [lmf,lpf]=bode(numf,denf,2*pi*w);
   lpf=lpf-360.0*taudf*w';                    % add in the delay

    stf1=['k    = ',sprintf('%g',kf)];
    stf2=['tau1  = ',sprintf('%g',tau1f),' s'];
    stf3=['tau2  = ',sprintf('%g',tau2f),' s'];
    stf4=['taud  = ',sprintf('%g',taudf),' s'];

elseif mo==2,

   [ves,dum1,dum2,dum3,dum4]=oldf_anal(sub_code,vftarr,tc,d,sample_freq,mo);
   clear dum1; clear dum2; clear dum3; clear dum4;

elseif mo==3,

   [ves,dum1,dum2,dum3,dum4]=oldf_anal(sub_code,vftarr,tc,d,sample_freq,mo);
   clear dum1; clear dum2; clear dum3; clear dum4;

end;

[n,m]=size(ves);
if n>1,
   p_act=mean(ves);
else,
   p_act=ves;
end;
gain=abs(ves);
phase=180/pi*phase_correct(angle(ves));

ga=abs(p_act);                          %compute mean gain
ph=180.0/pi*phase_correct(angle(p_act));      %compute mean phase

clg; hold off;

if mo==3,
   subplot(211);semilogx(d,20*log10(gain),'wx'); grid;
```

```
   subplot(212);semilogx(d,phase,'wx'); grid;
else,
   if n>2,
      egain=std(20*log10(gain));
      ephase=std(phase);

      if mo==1,
         subplot(211);
           errorbar(d,20*log10(ga),egain,'semilogx','"wx"'); grid;
          hold on; semilogx(w,20*log10(lmf),'w');
           errorbar(d,20*log10(ga),egain,'semilogx','"wx"'); hold off;
           subplot(212); errorbar(d,ph,ephase,'semilogx','"wx"'); grid;
          hold on; subplot(212); semilogx(w,lpf,'w');
           errorbar(d,ph,ephase,'semilogx','"wx"'); hold off;
      else,
         subplot(211);
           errorbar(d,20*log10(ga),egain,'semilogx','"wx"'); grid;
           subplot(212); errorbar(d,ph,ephase,'semilogx','"wx"'); grid;
      end;
   else,
      if mo==1,
        subplot(211);
          axis([-2,0,-20,20]); semilogx(d,20*log10(ga),'wx'); grid;
          hold on; semilogx(w,20*log10(lmf),'w'); hold off;
          subplot(212); axis([-2,0,-200,0]); semilogx(d,ph,'wx'); grid;
          hold on; semilogx(w,lpf,'w'); hold off;
      else,
         subplot(211);
          axis([-2,0,-20,20]); semilogx(d,20*log10(ga),'wx'); grid;
         subplot(212);
          axis([-2,0,-200,0]); semilogx(d,ph,'wx'); grid;
      end;
   end;
end;

hold on;

subplot(211);
title(plot_title1);
xlabel('Frequency (Hz)');
ylabel('Decibels');

subplot(212);
xlabel('Frequency (Hz)');
ylabel('Degrees');

if mo==1,
   v=axis;
   axis;

   inc=(abs(v(3))+abs(v(4)))/10.0;
   base=(v(3)+v(4))/2+4*inc;

    text(.4,base,stf1);
```

```
    text(.4,base-inc,stf2);
    text(.4,base-2*inc,stf3);
    text(.4,base-3*inc,stf4);
end;

hold off;

return;
```

```
function [ves,c1,c2,c3,c4]=oldf_anal(sub_code,vftarr,tc,d,sample_freq,mo)
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       OLDF_ANAL loads the describing function data generated by freq_anal
%       for one or all subjects for a given visual field and passes back the
%       the data at the vestibular frequencies, as well as the parameters for
%       a lead-lag with pure delay transfer function fit to the mean data (if
%       requested.
%Routines called:
%       model_fit
%Parameters:
%       sub_code is the subject code
%       vftarr contains the run numbers for the particular visual field
%       tc is the number of subject to be overlayed (1 or 6)
%       d is the frequencies in the disturbance (vestibular)
%       sample_freq is the sampling frequency
%       mo is a flag:
%               mo == 1 causes a transfer function fit to be done
%               mo <> 1 causes the describing function data to be passed
%                       back with no fit done.

%store the Link closed-loop dynamics at the disturbance frequencies
gain = [0.5852;0.6888;0.9718;0.9473;0.9744;0.9538;0.5423;0.2135;...
0.3810;-0.5400;-0.7082;-2.8348]'; %gain
phase = [1.2684;-8.0116;-21.8144;-28.7031;-39.1025;-37.8285;...
-32.7279;-49.1164;-68.0153;-86.7074;-93.4051;-119.3100]'; %phase

ves=[];

cc=1;
rownum=1;

for j=1:tc,
   for kk=1:length(vftarr),
      if vftarr(kk)<10,
           file=['load ',sub_code(j,:),'0',sprintf('%g',vftarr(kk)),...
'.ves'];
      else,
           file=['load ',sub_code(j,:),sprintf('%g',vftarr(kk)),'.ves'];
      end;
      chdir SS4:scott:Thesis5ConvertedData:link;
      eval(file);
      chdir SS4:scott:Thesis5analysis:link;

      ves(rownum,:)=p_act.'; %non-conjugate transpose!
      rownum=rownum+1;
   end;
end;
```

```
ga=20*log10(abs(ves));
ph=180.0/pi*phase_correct(angle(ves));

[n,m]=size(ves);
for jj=1:n,
   olg(jj,:)=20*log10(gain)+ga(jj,:);
   olp(jj,:)=phase+ph(jj,:);
end;

olp=180/pi*phase_correct(pi/180.0*olp);

clear ves;

for ii=1:n,
   for jj=1:m,
      ves(ii,jj)=10^(olg(ii,jj)/20.0)*exp(sqrt(-1)*olp(ii,jj)*pi/180.0);
   end;
end;

if mo==1,
   [n,m]=size(ves);
   if n==1,
      e=[ones(olg)' ones(olp)'];
      [c1 c2 c3 c4]=model_fit(ves,e,d);
   else,
      e=[std(olg)' std(olp)'];
      [c1 c2 c3 c4]=model_fit(mean(ves),e,d);
   end;
end;

return;
```

```
function overlay(sample_freq,run_time,num_fft_pts,sub_code,tc)
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       OVERLAY allows the user to overlay plots of any two time-series
%       from any two runs.
%Routines called:
%       none
%Parameters:
%       sample_freq is the sampling frequency
%       run_time is the length of the run (in seconds)
%       num_fft_pts is the number of points in the fft
%       sub_code is the subject code
%       tc is the number of subjects

dark=[0,9];
con=[4,11];
cvl=[1];
cvr=[7];
fix=[3,6];
ss=[2,5,8,10];

d1=[3,5,11,17,23,31,41,53,71,89,109,137]/run_time; % vestibular frequencies
d2=[4,7,13,19,29,37,43,61,83,97,127,151]/run_time; % visual frequencies

max_freq=max(d2);

plot_title=['Subject ',sub_code,': '];

overlay_type=menu('Choose overlay plot type','Link Position','Link Velocity',...
  'Joystick','Visual Velocity');

if (overlay_type==1),
   file_ext=['.linkpos'];
  plot_title=[plot_title,'Link Position, '];
elseif (overlay_type==2),
   file_ext=['.linkvel'];
  plot_title=[plot_title,'Link Velocity, '];
elseif (overlay_type==3),
   file_ext=['.joystick'];
  plot_title=[plot_title,'Control Wheel, '];
else,
   file_ext=['.projvel'];
  plot_title=[plot_title,'Visual Velocity, '];
end;

plot_1=menu('Visual  Field','FIX','DARK','SS','CVL','CVR','CON');

if (plot_1==1),
    rn_1=input('Trial number: ');
```

```
    vec=fix;
     plot_title=[plot_title,'FIX',sprintf('%g',fix(rn_1)),' (solid) and '];
elseif (plot_1==2),
    rn_1=input('Trial number: ');
    vec=dark;
     plot_title=[plot_title,'DARK',sprintf('%g',dark(rn_1)),' (solid) and '];
elseif (plot_1==3),
    rn_1=input('Trial number: ');
    vec=ss;
     plot_title=[plot_title,'SS',sprintf('%g',ss(rn_1)),' (solid) and '];
elseif (plot_1==4),
    rn_1=1;
    vec=cvl;
     plot_title=[plot_title,'CVL',sprintf('%g',cvl(rn_1)),' (solid) and '];
elseif (plot_1==5),
    rn_1=1;
    vec=cvr;
     plot_title=[plot_title,'CVR',sprintf('%g',cvr(rn_1)),' (solid) and '];
elseif (plot_1==6),
    rn_1=input('Trial number: ');
    vec=con;
     plot_title=[plot_title,'CON',sprintf('%g',con(rn_1)),' (solid) and '];
end;

plot_2=menu('Overlay  with','FIX','DARK','SS','CVL','CVR','CON');;

if (plot_2==1),
    rn_2=input('Trial number: ');
    vec2=fix;
     plot_title=[plot_title,'FIX',sprintf('%g',fix(rn_2)),' (dashed)'];
elseif (plot_2==2),
    rn_2=input('Trial number: ');
    vec2=dark;
     plot_title=[plot_title,'DARK',sprintf('%g',dark(rn_2)),' (dashed)'];
elseif (plot_2==3),
    rn_2=input('Trial number: ');
    vec2=ss;
     plot_title=[plot_title,'SS',sprintf('%g',ss(rn_2)),' (dashed)'];
elseif (plot_2==4),
    rn_2=1;
    vec2=cvl;
     plot_title=[plot_title,'CVL',sprintf('%g',cvl(rn_2)),' (dashed)'];
elseif (plot_2==5),
    rn_2=1;
    vec2=cvr;
     plot_title=[plot_title,'CVR',sprintf('%g',cvr(rn_2)),' (dashed)'];
elseif (plot_2==6),
    rn_2=input('Trial number: ');
    vec2=con;
     plot_title=[plot_title,'CON',sprintf('%g',con(rn_2)),' (dashed)'];
end;

if vec(rn_1)<10,
    file=['load ',sub_code,'0',sprintf('%g',vec(rn_1)),file_ext];
```

```
else,
   file=['load ',sub_code,sprintf('%g',vec(rn_1)),file_ext];
end;

chdir SLS_HD:scott:Thesis5ConvertedData:link;
eval(file);
chdir SS2:Thesis5analysis:link;

if (overlay_type==1),
   y(:,1)=14.0/2048*pos;
   clear pos;
elseif (overlay_type==2),
   y(:,1)=-1.0*10/2048/0.715*tach;
   clear tach;
elseif (overlay_type==3),
   y(:,1)=-1.0*14/2048*joy;
   clear joy;
elseif (overlay_type==4),
   y(:,1)=-1.0*10/2048/0.2887*trig;
end;

if vec(rn_2)<10,
    file=['load ',sub_code,'0',sprintf('%g',vec2(rn_2)),file_ext];
else,
   file=['load ',sub_code,sprintf('%g',vec2(rn_2)),file_ext];
end;

chdir SLS_HD:scott:Thesis5ConvertedData:link;
eval(file);
chdir SLS_HD:scott:Thesis5analysis:link;

if (overlay_type==1),
   y(:,2)=14.0/2048*pos;
   clear pos;
elseif (overlay_type==2),
   y(:,2)=-1.0*10/2048/0.715*tach;
   clear tach;
elseif (overlay_type==3),
   y(:,2)=-1.0*14/2048*joy;
   clear joy;
elseif (overlay_type==4),
   y(:,2)=-1.0*10/2048/0.2887*trig;
end;

psd=menu('PSD?','Yes','No');

if (psd==2),
   if (overlay_type==1),
      y_label=['Degrees'];
   else
      y_label=['Degrees/s'];
   end;

   x_label=['Time (s)'];
```

```
   t=1/sample_freq:1/sample_freq:run_time;

   hold off; clg;

   size(y)

     plot(t,y(:,1),'w',t,y(:,2),'w--');

elseif (psd==1),
   x_label=['Frequency(Hz)'];
   y_label=['Power'];
   plot_title=['PSD, ',plot_title];
   hold off; clg;

   y1=fft(y(:,1),num_fft_pts);
   y2=fft(y(:,2),num_fft_pts);

   Py1=y1.*conj(y1)/8192;
   Py2=y2.*conj(y2)/8192;

   f=sample_freq*(0:(num_fft_pts/2-1))/num_fft_pts;

   freq_find=find(f>=max_freq);
   ind=freq_find(1)-1;

    plot(f(2:ind),Py1(2:ind),'w',f(2:ind),Py2(2:ind),'w'); grid;
   hold;

   if (plot_1==3|plot_2==3),
        plot(d1,0*ones(d1),'wx',d2,0*ones(d2),'wo');
   else,
       plot(d1,0*ones(d1),'wx');
   end;
end;

grid;
xlabel(x_label);
ylabel(y_label);
title(plot_title);
hold off;

return;
```

```
function [phase]=phase_correct(phase)
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       Correct the phase boundaries by adding multiples of 2*pi
%       Phase is corrected across rows, and MUST be in radians.
%Routines called:
%       none
%Parameters:
%       phase is the row vector containing the phase to be corrected

tol_pos=170*pi/180; %here is a tolerance
tol_neg=-1*pi;      %here is another tolerance

[n,m]=size(phase);

for kk=1:n,
   for ll=1:m,
      if phase(ll)>tol_pos,
         phase(kk,ll)=phase(kk,ll)-2*pi;
      elseif phase(kk,ll)<tol_neg,
         phase(kk,ll)=phase(kk,ll)+2*pi;
      end;
   end;
end;

return;
```

```
function psd(sample_freq,run_time,num_fft_pts,sub_code,tc)
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       PSD plots a power spectra density of position, velocity, control
%       wheel, or visual field velocity at the frequencies in the vestibular
%       and visual disturbances. The subject remnant is also displayed on the
%       plot, as are errorbars.
%Routines called:
%       none
%Parameters:
%       sample_freq is the sampling frequency
%       run_time is the length of the run (in seconds)
%       num_fft_pts is the number of points in the fft
%       sub_code is the subject code
%       tc is the number of subjects to be overlayed (1 or 6)

pos_d1=[2.4009,2.4410,2.4545,2.3733,2.1623,1.7746,1.2495,0.8847,...
0.6246,0.5475,0.5822,0.5959]; %vestibular dist. amplitudes (deg)

d1=[3,5,11,17,23,31,41,53,71,89,109,137]/run_time; % vestibular frequencies
d2=[4,7,13,19,29,37,43,61,83,97,127,151]/run_time; % visual frequencies

dark=[0,9];
con=[4,11];
cvl=[1];
cvr=[7];
fix=[3,6];
ss=[2,5,8,10];

max_freq=max(d2);

if tc<2,
   plot_title=['Subject ',sub_code,': '];
end;

psd_type=menu('PSD of?','Link Position','Link Velocity','Control Wheel',...
'Visual Velocity');

if (psd_type==1),
   file_ext=['.linkpos'];
   plot_title=[plot_title,'Link Position: '];
   ampst=['(deg)'];
elseif (psd_type==2),
   file_ext=['.linkvel'];
   plot_title=[plot_title,'Link Velocity: '];
   ampst=['(deg/s)'];
elseif (psd_type==3),
   file_ext=['.joystick'];
   plot_title=[plot_title,'Control Wheel: '];
```

```
   ampst=['(deg)'];
else,
   file_ext=['.projvel'];
   plot_title=[plot_title,'Visual Velocity: '];
   ampst=['(deg/s)'];
end;

plot_1=menu('Visual  Field','FIX','DARK','SS','CVL','CVR','CON');

if (plot_1==1),
   vec=fix;
   plot_title=[plot_title,'FIX'];
elseif (plot_1==2),
   vec=dark;
   plot_title=[plot_title,'DARK'];
elseif (plot_1==3),
   vec=ss;
   plot_title=[plot_title,'SS'];
elseif (plot_1==4),
   vec=cvl;
   plot_title=[plot_title,'CVL'];
elseif (plot_1==5),
   vec=cvr;
   plot_title=[plot_title,'CVR'];
elseif (plot_1==6),
   vec=con;
   plot_title=[plot_title,'CON'];
end;

f = sample_freq*(0:num_fft_pts/2-1)/num_fft_pts;
temp = f*run_time;

freq_find=find(f>=max_freq);
ind=freq_find(1);

for j=1:length(d1),
   ff=find(f>=d1(j));
   ves_ind(j)=ff(1);
end;

for j=1:length(d2),
   ff=find(f>=d2(j));
   vis_ind(j)=ff(1);
end;

cves=1;
cvis=1;
co=1;

for j=2:ind,
   if j==ves_ind(cves),
      if cves<length(d1),
         cves=cves+1;
      end;
```

```
   elseif j==vis_ind(cvis),
      if cvis<length(d2),
         cvis=cvis+1;
      end;
   elseif temp(j)==round(temp(j)),
      rem_ind(co)=j;
      frem(co)=f(j);
      co=co+1;
   end;
end;

row1=ones(d1)';
row2=ones(d2)';
row3=ones(f(1:length(f)-24))';

for sub_num=1:tc,

   for ww=1:length(vec),

      if vec(ww)<10,
           file=['load ',sub_code(sub_num,:),'0',sprintf('%g',vec(ww)),...
file_ext];
      else,
          file=['load ',sub_code(sub_num,:),sprintf('%g',vec(ww)),file_ext];
      end;

      chdir SLS_HD:scott:Thesis5ConvertedData:link;
      eval(file);
      chdir SLS_HD:scott:Thesis5analysis:link;

      if (psd_type==1),
         y=14.0/2048*pos;
         clear pos;
      elseif (psd_type==2),
         y=-1.0*10/2048/0.715*tach;
         clear tach;
      elseif (psd_type==3),
         y=-1.0*14/2048*joy;
         clear joy;
      elseif (psd_type==4),
         y=-1.0*10/2048/0.2887*trig;
      end;

      yy=fft(y,num_fft_pts);
      Py=abs(yy)*2/num_fft_pts;

      for j=1:length(d1),
         ampd1(row1(j),j)=Py(ves_ind(j));
         row1(j)=row1(j)+1;
      end;

      for j=1:length(d2),
         ampd2(row2(j),j)=Py(vis_ind(j));
         row2(j)=row2(j)+1;
```

```
      end;

      for j=1:length(rem_ind),
         Pyrem(row3(j),j)=Py(rem_ind(j));
         row3(j)=row3(j)+1;
      end;
   end;
end;
chdir SLS_HD:scott:thesis5analysis:link;

if (plot_1~=4&plot_1~=5)|tc>1,
   Pyrem=mean(Pyrem);  %compute mean remnant response
end;

hold off; clg;

axis('square');

axis([0 0.8 0 2.5]);      %set axis limits

fl=0;

%if tc==1&plot_1~=3,
if tc==1,
   fl=1;
elseif tc==1&plot_1==3,
   e1=std(ampd1)/sqrt(4);
   e2=std(ampd2)/sqrt(4);
elseif tc>1&(plot_1==4|plot_1==5),
   e1=std(ampd1)/sqrt(6);
   e2=std(ampd2)/sqrt(6);
elseif tc>1&plot_1==3,
   e1=std(ampd1)/sqrt(24);
   e2=std(ampd2)/sqrt(24);
else,
   e1=std(ampd1)/sqrt(12);
   e2=std(ampd2)/sqrt(12);
end;

if fl==1,
   if plot_1==4|plot_1==5,
      if psd_type==1|psd_type==3,
            plot(d1,pos_d1,'+',d1,ampd1,'wx',d2,ampd2,'wo',frem,Pyrem','w*');
         hold on; plot(d1,pos_d1,'w-',d1,ampd1,'w--'); hold off;
      else,
           plot(d1,ampd1,'wx',d2,ampd2,'wo',frem,Pyrem','w*');
         hold on; plot(d1,pos_d1,'w-',d1,ampd1,'w--'); hold off;
      end;
      grid;
   else,
      if psd_type==1|psd_type==3,
           plot(d1,pos_d1,'+',d1,mean(ampd1),'wx',d2,mean(ampd2),'wo',frem,...
Pyrem','w*');
         hold on; plot(d1,pos_d1,'w-',d1,mean(ampd1),'w--'); hold off;
```

```
      else,
           plot(d1,mean(ampd1),'wx',d2,mean(ampd2),'wo',frem,...
Pyrem','w*');
          hold on; plot(d1,pos_d1,'w-',d1,mean(ampd1),'w--'); hold off;
      end;
      grid;
   end;
else,
   if psd_type==1|psd_type==3,
        plot(d1,pos_d1,'+',frem,Pyrem,'w*');
       hold on; plot(d1,pos_d1,'w-'); hold off;
   else,
       plot(frem,Pyrem,'w*');
       hold on; plot(d1,pos_d1,'w-'); hold off;
   end;
   hold on;
     errorbar(d1,mean(ampd1),e1,'plot',"'wx'");
    hold on; plot(d1,mean(ampd1),'w--');
     errorbar(d2,mean(ampd2),e2,'plot',"'wo'");
   hold off;
   grid;
end;

mrem=mean(Pyrem);
hold on;
plot([0.0 max_freq],[mrem mrem]','w-');

xlabel('Frequency(Hz)');
ylabel(['Amplitude',ampst]);
plot_title=['PSD : ',plot_title];
title(plot_title);
hold off;

axis('normal');

%for jj=1:length(vec),
%    ampd2(jj,:)'
%end;

return;
```

```
function raw(sample_freq,run_time,sub_code,tc)
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       RAW plots the raw subject position, velocity, control wheel,
%       and visual field velocity for a single trial with a single
%       visual field condition.
%Routines called:
%       none
%Parameters:
%       sample_freq is the sampling frequency
%       run_time is the length of the run (in seconds)
%       sub_code is the subject code
%       tc is the number of subjects to be overlayed (1 or 6)

dark=[0,9];
con=[4,11];
cvl=1;
cvr=7;
fix=[3,6];
ss=[2,5,8,10];

plot_1=menu('Visual  Field','FIX','DARK','SS','CVL','CVR','CON');

if (plot_1==1),
   vec=fix;
elseif (plot_1==2),
   vec=dark;
elseif (plot_1==3),
   vec=ss;
elseif (plot_1==4),
   vec=cvl;
elseif (plot_1==5),
   vec=cvr;
elseif (plot_1==6),
   vec=con;
end;

rn_1=input('Trial number: ');

if (plot_1==1),
    plot_title=[plot_title,sprintf('%g',vec(rn_1)),' : FIX'];
elseif (plot_1==2),
    plot_title=[plot_title,sprintf('%g',vec(rn_1)),' : DARK'];
elseif (plot_1==3),
    plot_title=[plot_title,sprintf('%g',vec(rn_1)),' : SS'];
elseif (plot_1==4),
    plot_title=[plot_title,sprintf('%g',vec(rn_1)),' : CVL'];
elseif (plot_1==5),
    plot_title=[plot_title,sprintf('%g',vec(rn_1)),' : CVR'];
```

```
elseif (plot_1==6),
   plot_title=[plot_title,sprintf('%g',vec(rn_1)),' : CON'];
end;

if vec(rn_1)<10,
    file=['load ',sub_code,'0',sprintf('%g',vec(rn_1)),'.linkpos'];
else,
    file=['load ',sub_code,sprintf('%g',vec(rn_1)),'.linkpos'];
end;

chdir SLS_HD:scott:Thesis5ConvertedData:link;
eval(file);
chdir SLS_HD:scott:Thesis5analysis:link;

pos=14.0/2048*pos;

if vec(rn_1)<10,
    file=['load ',sub_code,'0',sprintf('%g',vec(rn_1)),'.linkvel'];
else,
    file=['load ',sub_code,sprintf('%g',vec(rn_1)),'.linkvel'];
end;

chdir SLS_HD:scott:Thesis5ConvertedData:link;
eval(file);
chdir SLS_HD:scott:Thesis5analysis:link;

tach=-1.0*10/2048/0.715*tach;

if vec(rn_1)<10,
    file=['load ',sub_code,'0',sprintf('%g',vec(rn_1)),'.joystick'];
else,
    file=['load ',sub_code,sprintf('%g',vec(rn_1)),'.joystick'];
end;

chdir SLS_HD:scott:Thesis5ConvertedData:link;
eval(file);
chdir SLS_HD:scott:Thesis5analysis:link;

joy=-1.0*14/2048*joy;

if vec(rn_1)<10,
    file=['load ',sub_code,'0',sprintf('%g',vec(rn_1)),'.projvel'];
else,
    file=['load ',sub_code,sprintf('%g',vec(rn_1)),'.projvel'];
end;

chdir SLS_HD:scott:Thesis5ConvertedData:link;
eval(file);
chdir SLS_HD:scott:Thesis5analysis:link;

ptach=-1*10/2048/0.2887*trig;
clear trig;

t=1/sample_freq:1/sample_freq:run_time;
```

```
hold off;
clg;
axis([0,250,-15,15]);
plot(t,pos,'w'); grid; hold; plot(t,pos,'w'); hold off;
hold off;
xlabel('Time(s)');
ylabel('Degrees');
title([plot_title,' : Link Position']);

%pause;
%clg;
%axis([0,250,-15,15]);
%plot(t,tach,'w'); grid; hold; plot(t,tach,'w'); hold off;
%xlabel('Time(s)');
%ylabel('Degrees/s');
%title([plot_title,' : Link Velocity']);

pause;
clg;
axis([0,250,-15,15]);
plot(t,joy,'w'); grid; hold; plot(t,joy,'w'); hold off;
xlabel('Time(s)');
ylabel('Degrees');
title([plot_title,' : Control Wheel']);

pause;
clg;
axis([0,250,-30,30]);
plot(t,ptach,'w'); grid; hold; plot(t,ptach,'w'); hold off;
xlabel('Time(s)');
ylabel('Degrees/s');
title([plot_title,' : Projector Velocity']);
```

```
function rms(sample_freq,run_time,sub_code,tc)

%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       RMS computes the RMS of position, velocity, or control wheel
%f      or each visual field condition for one or all subjects and
%       displays it on a plot complete with errorbars.
%Routines called:
%       none
%Parameters:
%       sample_freq is the sampling frequency
%       run_time is the length of the run (in seconds)
%       num_fft_pts is the number of points in the fft
%       sub_code is the subject code
%       tc is the number of subjects to be overlayed (1 or 6)

dark=[0,9];
con=[4,11];
cvl=[1];
cvr=[7];
fix=[3,6];
ss=[2,5,8,10];

rms_type=menu('RMS of?','Link Position','Link Velocity','Control Wheel');

if (rms_type==1),
   file_ext=['.linkpos'];
   plot_title=[plot_title,'RMS Position : '];
   y_label=['Degrees'];
elseif (rms_type==2),
   file_ext=['.linkvel'];
   plot_title=[plot_title,'RMS Velocity : '];
   y_label=['Degrees/s'];
elseif (rms_type==3),
   file_ext=['.joystick'];
   plot_title=[plot_title,'RMS Joystick : '];
   y_label=['Degrees'];
end;

row=[1 1 1 1 1 1]';

for sub_num=1:tc,

   col=1;

   for plot_1=1:6,

      if (plot_1==1),
         vec=fix;
```

```
      elseif (plot_1==2),
         vec=dark;
      elseif (plot_1==3),
         vec=ss;
      elseif (plot_1==4),
         vec=cvl;
      elseif (plot_1==5),
         vec=cvr;
      elseif (plot_1==6),
         vec=con;
      end;

      for rn_1=1:length(vec),
         if vec(rn_1)<10,
              file=['load ',sub_code(sub_num,:),'0',...
sprintf('%g',vec(rn_1)),file_ext];
         else,
             file=['load ',sub_code(sub_num,:),...
sprintf('%g',vec(rn_1)),file_ext];
         end;

         chdir SLS_HD:scott:Thesis5ConvertedData:link;
         eval(file);
         chdir SLS_HD:scott:Thesis5analysis:link;

         if (rms_type==1),
            y=14.0/2048*pos;
            clear pos;
         elseif (rms_type==2),
            y=-1.0*10/2048/0.715*tach;
            clear tach;
         elseif (rms_type==3),
            y=-1.0*14/2048*joy;
            clear joy;
         elseif (rms_type==4),
            y=-1.0*10/2048/0.2887*trig;
         end;

         rms(row(col),col)=sqrt(mean(y.*y));

         row(col)=row(col)+1;

      end;

      col=col+1;

   end;

end;

a=tc*length(fix);
b=tc*length(dark);
c=tc*length(ss);
d=tc*length(cvl);
```

```
e=tc*length(cvr);
f=tc*length(con);

aa=rms(1:a,1);
bb=rms(1:b,2);
cc=rms(1:c,3);
dd=rms(1:d,4);
ee=rms(1:e,5);
ff=rms(1:f,6);

e1=[std(aa) std(bb) std(cc) std(dd) std(ee) std(ff)];

clear rms;
rms=[mean(aa) mean(bb) mean(cc) mean(dd) mean(ee) mean(ff)];

hold off;
clg;
axis('square');

if tc==1,
   e1=[0 0 e1(3)/sqrt(4) 0 0 0];
    errorbar([0 1 2 3 4 5],rms,e1,'plot','"wx"');
elseif tc>1,
   a=sqrt(a);
   b=sqrt(b);
   c=sqrt(c);
   d=sqrt(d);
   e=sqrt(e);
   f=sqrt(f);
   e1=[e1(1)/a e1(2)/b e1(3)/c e1(4)/d e1(5)/e e1(6)/f];
    errorbar([0 1 2 3 4 5],rms,e1,'plot','"wx"');
end;

grid; hold on;

if tc>1,
    errorbar([0 1 2 3 4 5],rms,e1,'plot','"wx"');
else,
    errorbar([0 1 2 3 4 5],rms,e1,'plot','"wx"');
end;

hold off;

xlabel('Visual Field Code');
ylabel(y_label);

xstr=['0=FIX  1=DARK  2=SS  3=CVL  4=CVR  5=CON'];

if tc==1,
   plot_title=[sub_code,' : ',plot_title];
end;

title([plot_title,xstr]);
```

```
axis('normal');
return;
```

```
function spm(sample_freq,run_time,num_fft_pts,sub_code,tc)
%Author:
%       Scott Stephenson
%Date:
%       September, 1993
%Description:
%       SPM plots a power spectra density of trainer position with and
%       without subject compensation at the frequencies in the vestibular
%       disturbance complete with errorbars. The area between the curves
%       represents the improvement due to subject nulling.
%Routines called:
%       none
%Parameters:
%       sample_freq is the sampling frequency
%       run_time is the length of the run (in seconds)
%       num_fft_pts is the number of points in the fft
%       sub_code is the subject code
%       tc is the number of subjects to be overlayed (1 or 6)

pos_d1=[2.4009,2.4410,2.4545,2.3733,2.1623,1.7746,1.2495,0.8847,...
0.6246,0.5475,0.5822,0.5959]; % vestibular dist. amplitudes (deg)

d1=[3,5,11,17,23,31,41,53,71,89,109,137]/run_time; % vestibular frequencies

dark=[0,9];
con=[4,11];
cvl=[1];
cvr=[7];
fix=[3,6];
ss=[2,5,8,10];

max_freq=max(d1);

if tc<2,
   plot_title=['Subject ',sub_code,': '];
end;

plot_1=menu('Visual  Field','FIX','DARK','SS','CVL','CVR','CON');

if (plot_1==1),
   vec=fix;
   plot_title=[plot_title,'FIX'];
elseif (plot_1==2),
   vec=dark;
   plot_title=[plot_title,'DARK'];
elseif (plot_1==3),
   vec=ss;
   plot_title=[plot_title,'SS'];
elseif (plot_1==4),
   vec=cvl;
   plot_title=[plot_title,'CVL'];
elseif (plot_1==5),
```

```
    vec=cvr;
    plot_title=[plot_title,'CVR'];
elseif (plot_1==6),
    vec=con;
    plot_title=[plot_title,'CON'];
end;

f = sample_freq*(0:num_fft_pts/2-1)/num_fft_pts;
temp = f*run_time;

freq_find=find(f>=max_freq);
ind=freq_find(1);

for j=1:length(d1),
   ff=find(f>=d1(j));
   ves_ind(j)=ff(1);
end;

cves=1;

for j=2:ind,
   if j==ves_ind(cves),
      if cves<length(d1),
         cves=cves+1;
      end;
   end;
end;

row1=ones(d1)';
row=1;
spm_len=tc*length(vec);
spm=0.0*ones(spm_len,1);

for sub_num=1:tc,
   for ww=1:length(vec),
      if vec(ww)<10,
           file=['load ',sub_code(sub_num,:),'0',sprintf('%g',vec(ww)),...
'.linkpos'];
      else,
           file=['load ',sub_code(sub_num,:),sprintf('%g',vec(ww)),'.linkpos'];
      end;

      chdir SLS_HD:scott:Thesis5ConvertedData:link;
      eval(file);
      chdir SLS_HD:scott:Thesis5analysis:link;

      y=14.0/2048*pos;
      clear pos;

      yy=fft(y,num_fft_pts);
      Py=abs(yy)*2/num_fft_pts;

      for j=1:length(d1),
         ampd1(row1(j),j)=Py(ves_ind(j));
```

```
        row1(j)=row1(j)+1;
        spm(row)=spm(row)+(pos_d1(j)-Py(ves_ind(j)));
      end;
      spm(row)=spm(row)./sum(pos_d1);
      row=row+1;
   end;
end;

chdir SLS_HD:scott:thesis5analysis:link;

fl=0;

if tc==1&plot_1~=3,
   fl=1;
elseif tc==1&plot_1==3,
   e1=std(ampd1)/sqrt(4);
   e2=std(spm)/sqrt(4);
elseif tc>1&(plot_1==4|plot_1==5),
   e1=std(ampd1)/sqrt(6);
   e2=std(spm)/sqrt(6);
elseif tc>1&plot_1==3,
   e1=std(ampd1)/sqrt(24);
   e2=std(spm)/sqrt(24);
else,
   e1=std(ampd1)/sqrt(12);
   e2=std(spm)/sqrt(12);
end;

hold off; clg;

axis('square');

axis([0 0.8 0 2.5]);      %set axis limits

if fl==1,
   if plot_1==4|plot_1==5,
        plot(d1,ampd1,'wo',d1,pos_d1,'wx',...
d1,ampd1,'w--',d1,pos_d1,'w-');
   else,
        plot(d1,mean(ampd1),'wo',d1,pos_d1,'wx',...
d1,mean(ampd1),'w--',d1,pos_d1,'w-');
   end;
else,
     errorbar(d1,mean(ampd1),e1,'plot','"wo"'); grid;
   hold on;
     plot(d1,pos_d1,'w-',d1,pos_d1,'wx',d1,mean(ampd1),'w--');
   hold off;
end;

grid;
xlabel('Frequency (Hz)');
ylabel('Amplitude (deg)');
title(plot_title);
```

```
axis('normal');

if fl==1,
   fprintf('\nThe mean SPM was %6.4f.\n',mean(spm));
else,
   fprintf('\nThe mean SPM was %6.4f +/- %6.4f.\n',mean(spm),e2);
end;

for ww=1:length(vec),
   ampd1(ww,:)'
end;

return;
```