A System to Present Gated Stimuli During fMRI: Observation of Amygdalar Activity

by

James F. Selph

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degrees of

Master of Engineering in Electrical Engineering and Computer Science
and
Bachelor of Science in Electrical Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

January 31, 1996

Author ....
Department of Electrical Engineering and Computer Science
January 31, 1996

Certified by ....
Frederic R. Morgenthaler
Professor of Electrical Engineering
Thesis Supervisor

Accepted by ....
Frederic R. Morgenthaler
Chairman, Departmental Committee on Graduate Students

Contents

A System to Present Gated Stimuli During fMRI: Observation of Amygdalar Activity

by

James F. Selph

Submitted to the Department of Electrical Engineering and Computer Science
on January 31, 1996 in partial fulfillment of the requirements for the degrees of
Master of Engineering in Electrical Engineering and Computer Science and
Bachelor of Science in Electrical Science and Engineering
at the MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Abstract

A system to present gated audio and visual stimuli during fMRI was developed and was used to detect amygdalar activity in response to emotional and emotionally neutral stimuli . Experimental paradigms were scripted and loaded into a computer as text files. A counter timer board in the computer is used to count scan initiation during an fMRI sequence. Based on the scan count and information in the script file, audio and visual stimuli stored in the computer are presented to the subject through a projector screen and headphones.

The system was implemented with studies designed to illicit amygdalar activity. Subjects were presented paradigms containing alternating periods of emotional stimuli (4 pictures over 24 seconds) and emotionally neutral stimuli (6.0 seconds per picture, over 24 sec. and over). Paradigms used emotional stimuli from one of three categories: pleasant, horrific, and faces expressing an emotion (happy, sad, fear, mad); all paradigms used pictures of brick walls for emotionally neutral stimuli. FMRI data was taken (at TE=40msec, TR=300msec, 75° flip angle) from two coronal slices 7mm apart crossing the amygdaloid complex, sampling once every 3.0 sec. per slice. Four pixels per slice were selected from the amygdala region for each experimental run. Data streams were divided into two distributions, scans after the viewing of emotional and neutral periods, and means calculated. Paired t-tests were performed on the means. Results showed a small increase in signal intensity above baseline, with mean increases of +0.22% (96.0% confidence) and +0.15% (98.4% confidence) for the posterior slices of the horror and face paradigms, respectively. The anterior slices of the nice and horror paradigms showed similar trends, while the posterior nice and anterior face showed no difference between the average difference of the means. Shifting the periods one and two samples to the right showed similar results, indicating delays in the processing of stimuli. Analysis averaging data streams indicated trends of an increase in signal intensity during the presentation of the first horror period; data streams also showed trends of a slight downward drift during the course of the experiment. Data acquisition with a head coil was considered superior to that of a surface coil for the subcortical amygdaloid structure.

Thesis Supervisor: Frederic R. Morgenthaler
Title: Professor of Electrical Engineering

Acknowledgments

I would like to take a moment to thank all the people who have supported me over the past eighteen months in the creation of this document.

First and foremost, I would like to thank my project advisors, Dr. Fein and Dr. Renshaw, who supervised the system design and the amygdala studies of my project. Always accessible, they provided the needed direction to my work. I have learned a great deal from working under them.

I would also like to thank my thesis advisor Prof. Morgenthaler, first for agreeing to supervise this project in the first place, as well as being a constant source of advice and encouragement during my many years at MIT. I greatly appreciate your patience and understanding while I worked on my thesis.

Thanks also goes to many people at the McLean Hospital MRI lab. Jim, who helped me to finally get the stimuli presentation system going (without damaging the scanner!). Dr. Todd, who was a steady supply of knowledge and advice on the acquisition of data. Abi, for all her help in running the experiments, and Camper for showing us how to analyze the data. And of course my laboratory coworkers, Anne and Eileen (who ran the scanner), Bernadette, and Jerry all of whom were interested in how I was doing and took the time to ask how things were going.

Last, but not least, I would like to thank some very important people in my life. My mother and father, who helped purchase this very expensive document. You have given me so much in my life, I do not know where to start. You are the best parents in the world. My brother John, for the many phone calls and friendly banter; I really enjoyed hearing how you were going scuba diving and snow skiing while I was writing my thesis. And of course, a special thanks to Claire, for bearing with me and being willing to walk the extra mile when I needed it, and without whom I would not have gotten my thesis done. The four of you are very special to me; I love you all very much.

Chapter 1

Introduction

1.1 Objectives

The past four years have seen a rapid expansion in study of the human body using functional magnetic resonance imaging. Magnetic Resonance Imaging (MRI) is a medical diagnostic procedure that uses a series of radiowave pulses to excite nuclei in a magnetic field, causing them to rebroadcast as they relax into equilibrium; these rebroadcast signals can be measured and filtered to produce a structural image of the brain. Recent advancements in imaging techniques have allowed researchers to observe regional changes in blood oxygenation levels as a result of local neural activity [1]. Since the first functional magnetic resonance images of the brain were produced without a contrast agent by Kwong et al in 1992 [2], researchers have had a non-invasive diagnostic tool that produces functional maps of the brain at both high temporal and spatial resolution.

The earliest functional MRI (fMRI) studies focused on the optical cortex, using alternating periods of optical stimulation to identify regions of the brain involved with visual processes [1, 2]; other studies have focused on the motor cortex and areas of the brain involved in speech and hearing. Functional MRI studies are now turning to areas of the brain whose functions are less understood, such as those involved with memory and emotion. One such area relevant to this study is the limbic system, along with one of its key structures-- the amygdala. The limbic system centers around processes related to behavior, emotion, and motivation. The amygdala is believed to be responsible for the determination of the emotional significance of stimuli, and is thought to be involved in the manifestations of emotional states, the most prominent being fear.

With the desire to use fMRI to probe more complex areas of the brain comes the need for the researcher to both present more complex stimuli and to better pinpoint the presentation of such stimuli. While the earliest stimuli presentation techniques relied optical stimulation (flashing lights), more recent techniques have presented of stimuli verbally (over headphones), or visual stimuli through the use of computer programs. Future projects have been proposed which plan to use a nonferrous air-pressure joystick that would allow scanning to report ethanol-induced alterations in subjects' mood states [3], while another proposes to obtain simultaneous EEG data without interfering with fMRI acquisition [4].

The brain imaging center at McLean Hospital recently expanded and has begun many new studies using fMRI. At the onset of this project, the brain imaging center at McLean Hospital had no system for the presentation of stimuli, other than the reading of verbal stimuli over a set of headphones. Projected studies at that time, including those investigating memory and emotion, required a system that could quickly and efficiently present a wider range of stimuli to subjects during fMRI, and to coordinate the timing of stimuli presentation with the scanning sequence.

In this project, a system was designed and implemented to present a variety of audio and visual stimuli during fMRI, and to gate the presentation of stimuli off the scanner. The presentation of a set of stimuli is controlled by a computer, which acquires information from the scanning sequence (via a counter-timer register) in order to determine when to present stimuli. Instructions for stimuli presentation are contained in a script file, which can be saved for use in subsequent trials and referred to during data analysis. Stimuli can be loaded into the computer in the form of picture files or sound files. The system offers a standardized method for the rapid presentation of stimuli over multiple experiments; it is relatively simple to use, and allows for the efficient use of scanner time and the analysis of data.

System development will be followed by assessing performance through experimental implementation: specifically, a series of experiments designed to produce changes in neural activity in the amygdala and to observe its activation during fMRI. Three sets of emotional stimuli (two sets containg pictures characterized as pleasant and horrific, a third containg faces displaying emotions of happiness, sadness, anger, and fear), and a set of emotionally neutral stimuli (pictures of bricks) were used. Alternating periods of emotional stimuli (from one of the three subsets) and emotionally neutral stimuli were presented during fMRI. The feasability of such an experiment has been suggested by PET experiments observing the processing of socially significant stimuli [5], and other functional imaging experiments that showed increases in signal intensity of the left amygdala in response to visual stimuli depicting sad faces [6]. Other PET studies suggest the feasibility of functionally observing the processing of socially significant stimuli [7, 8, 9, 10].

1.2 Scope

In order to understand the premise upon which the stimuli presentation program was created and the experimental results obtained with it, an overview of functional magnetic resonance imaging, and the amygdala is presented. The amygdala, the target of the cognitive paradigm for the functional imaging, is discussed in section 1.3. Section 1.4 gives a brief introduction of the fundamentals of MRI. Section 1.5 discusses fMRI and the technology underlying it.

Chapter 2 serves as bulk of the paper; it contains the design, implementation, and development of the system and program used to present gated stimuli during fMRI. It also describes experimental use of the system in a functional imaging study targeting the amygdala. Chapter 3 discusses system performance and results of the experiments. Finally, chapter 4 reflects on the results contained in chapter 3, as well as directions for future work.

1.3 The Amygdala

James Perez first proposed in 1937 that the limbic system is a neural system that serves as a substratum for emotions [11]. The limbic system, also known as the visceral or emotional brain [12], is a series of connections and nuclei within the brain that are concerned with processes related to behavioral expression, emotions, and motivation. Its functions center around those fundamental to the preservation of the organism and species, such as mating, fight or flight, feeding, and caring for the young. Though the precise borders of the system are debated, among the primitive subcortical structures that make up the limbic system are the hypothalamus, the hippocampus, the cingulate gyrus, and the amygdala, an almond-shaped [13] collection of nuclei. [14, 15, 16]

The amygdala, a key component of the limbic system [13], is a subcortical structure located in the dorsomedial tip of the temporal lobes, at the anterior end of the hippocampal formation, and at the anterior end of the lateral ventricle, continuous with the uncus of the parahippocampal gyrus. This location puts it in an ideal position to influence drive-related behavior patterns, operating as a higher order influence on the hypothalamus and emotions, as well as having a role in memory, learning, and stimulus reinforcement. [14, 15].

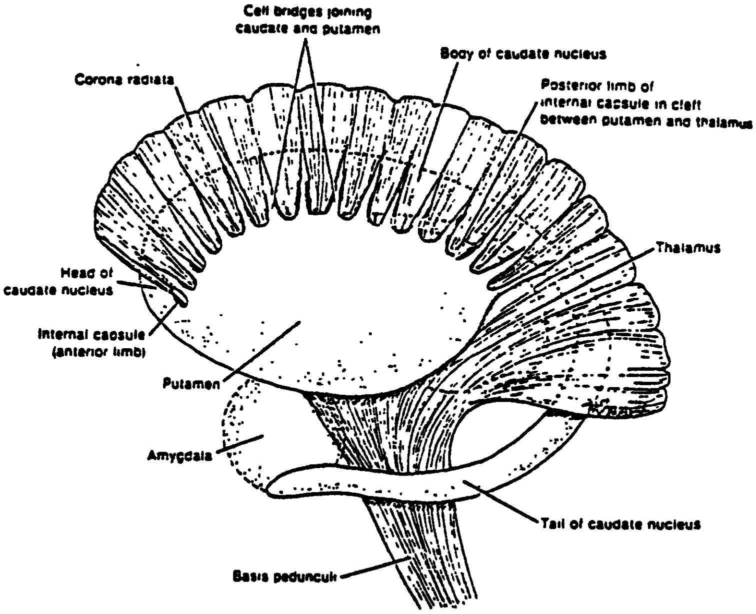
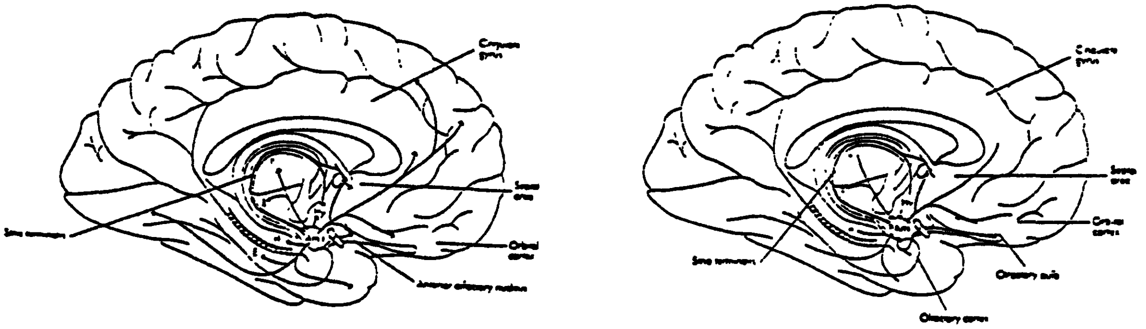


Fig. 1-1. Drawing of the amydgdala, thalamus, corpus striatum, and internal capsule separated from the rest of the body. From [14].

Most of the information the amygdala receives is highly processed. The afferents relaying this information to the amygdala may be from one of many sources. The most prominent inputs are sensory modalities, primarily visceral or olfactory projections (though the amygdala is not essential for olfactory discrimination) from the temporal lobe cortex, as well as the temporal and anterior cingulate cortices (which deliver much of the audio, visual, and somatosensory stimuli that reach the amygdala). Fibers leaving the amygdala go through two major pathways (the stria terminalis and the ventral amygdalofugal pathway [17]), often the reverse of those leading into the amygdala. Primary efferents for the amygdala lead to the hypothalamus (which controls visceral functions and activities involved in drive and emotional states), the septal areas of the limbic system, and the dorsomedial nucleus of the thalamus (involved in appetite). Thus, in the most simple terms, the amygdala serves as a relay for sensory stimuli from the olfactory cortex to the hypothalamus and the tegmentum of the midbrain. However, the amygdala and its connections are more complex. Other sources of input include the hypothalamus, the septal area of the limbic system, the orbital and insular cortex, the gustatory system, the somatosensory system, and the parabroachial nucleus. The amygdala also projects back to many of these same structures, as well as olfactory regions, the prefrontal cortex, and the orbital and anterior cingulate cortices. [14, 15, 18, 19, 20].



(a) (b)

Fig 1-2. Schematic of major (a) inputs and (b) outputs of the amygdala. The major efferent pathways are also shown. ST indicates fibers passing through the stria terminalis;V indicates fibers passing through a ventral course discovered by Price and Amaral [17]. From [14]

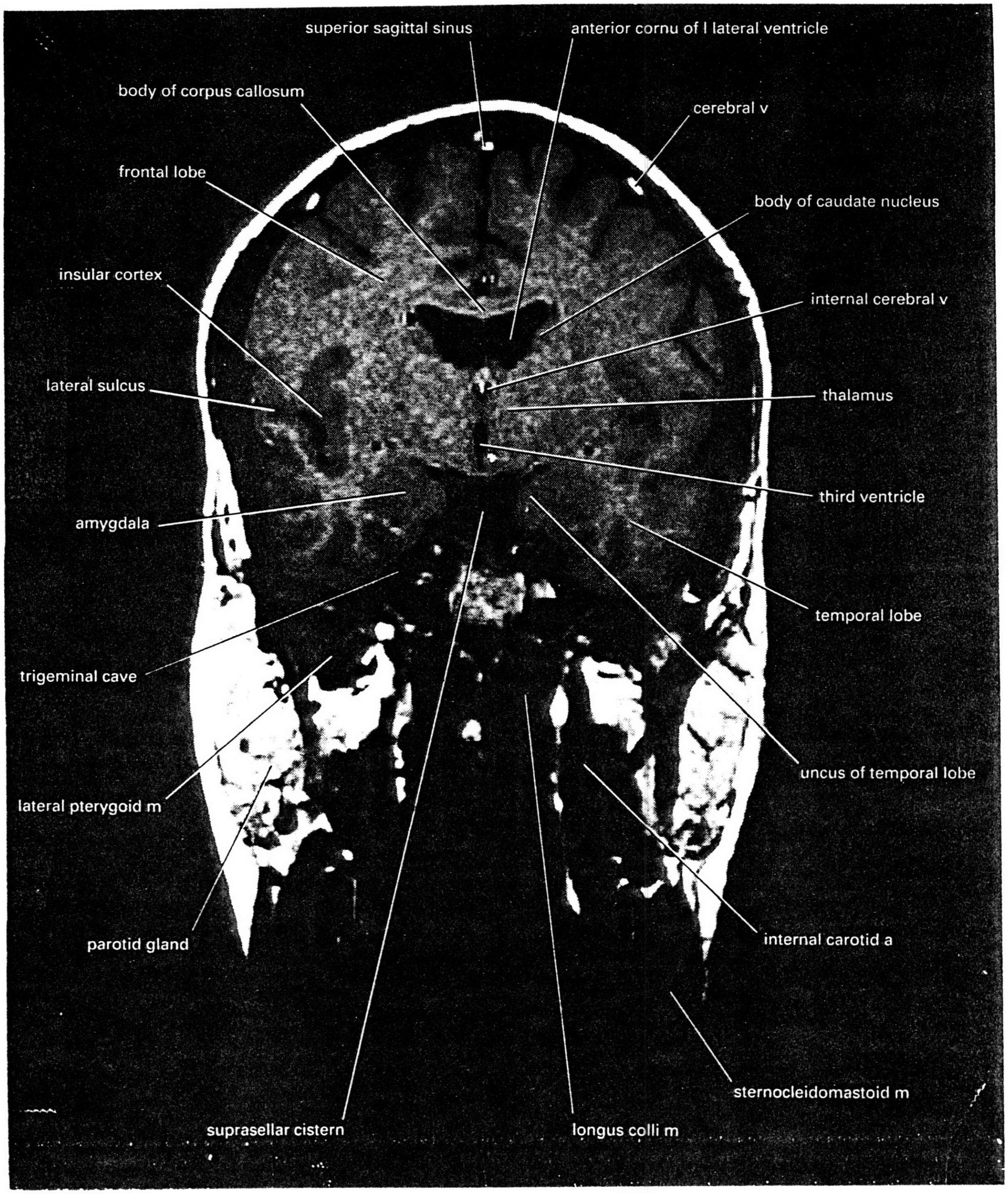
The role of the amygdala centers around processes in the brain related to feelings and emotion. The amygdala is believed to influence drive-related behavior patterns, serving as a higher order influence on the hypothalamus. Lesions and electrical stimulation of the amygdala produce a variety of effects on autonomic responses, emotional behavior, and feeding. These effects are similar to stimulation and lesioning of the lateral or medial hypothalamus. In general, almost all activity that results from stimulating or lesioning the hypothalamus can also be elicited by stimulating or lesioning the amygdala. Since the hypothalamus controls the visceral functions and activities involved in drive and emotional states, the amygdala is thus responsible for processing stimuli and directing these responses. [14, 15]

The notion that the amygdala contributes to the processing of the emotional significance of stimuli is also supported by findings from studies where neural activity has been recorded in the amygdala during exposure to emotional stimuli [21]. Studies have shown that units in the amygdala are responsive to faces, complex visual stimuli with clear emotional value [22]. Other studies have indicated that neurons of the amygdala are more sensitive to stimuli with biological significance, compared to other less relevant stimuli [23].

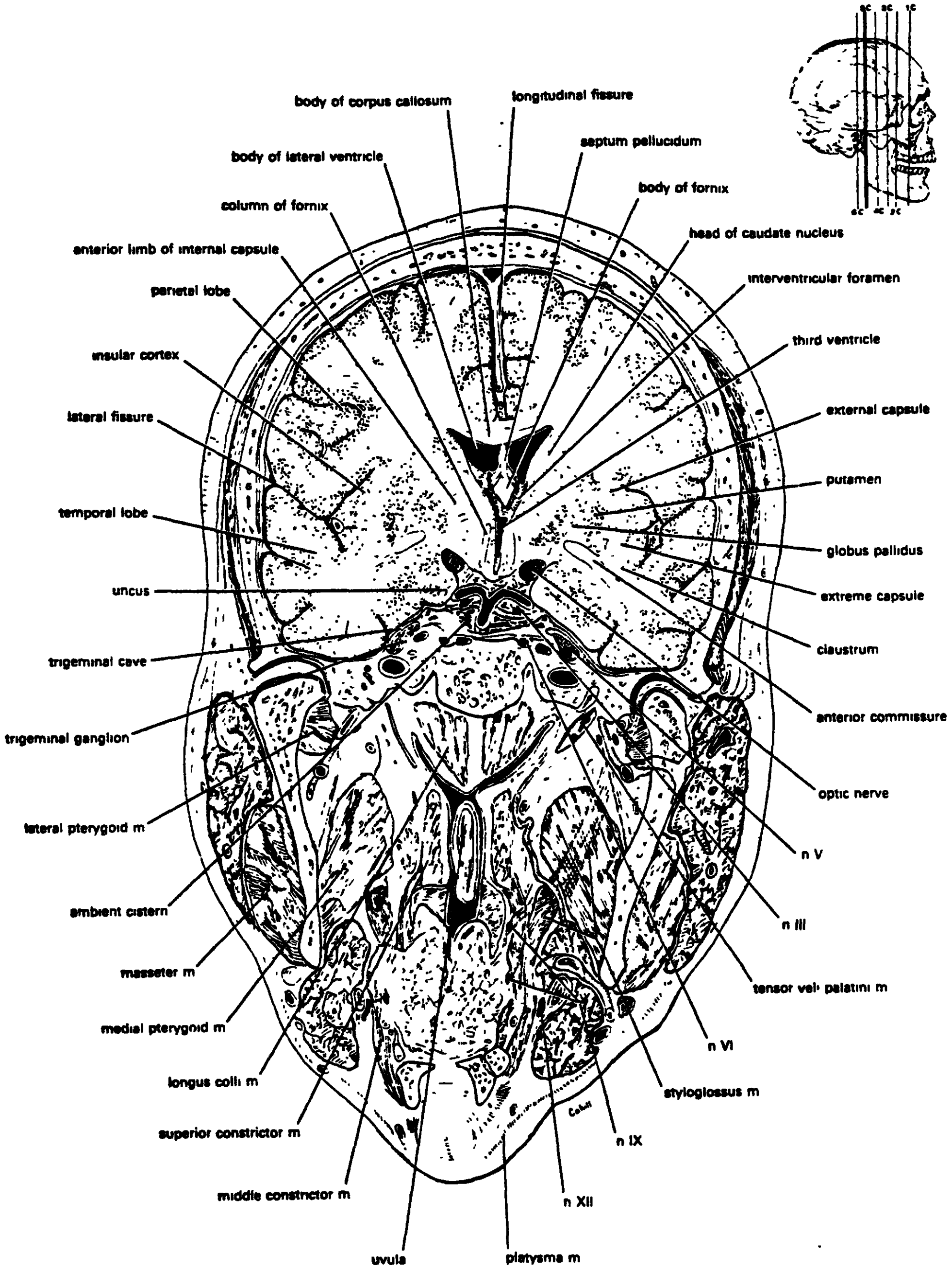
The role of the amygdala in subjective feelings is also well established. Research has shown that stimulation of the amygdala can result in a variety of emotions, but the most common is fear, accompanied by all of its autonomic manifestations. Studies have also shown that the amygdala plays a crucial role in the development and expression of conditioned fear; specifically, the association of the conditions and the state of fear [24]. Conversely, bilateral destruction of the amygdala causes a decrease in aggression. As a result, treated animals become tame and placid. This a unique form of memory deficit, one that impairs the ability to learn the appropriate emotional response to a stimulus [14].

Most recently, the amygdala has also been associated with the process of learning, particularly those tasks that require the coordination of different sensory modalities, or the association of a stimulus with an effective emotional response [14]. Studies of the Kluver-Bucy syndrome have strongly implicated the amygdala as a focal structure in the processing of emotion [21]. Damage to the amygdala has been shown to result in the oral tendencies, lack of fear, inability to discriminate food from other objects, hypersexuality, and tameness associated with the Kluver-Bucy syndrome [25]. It has since been suggested that the emotional changes of the Kluver-Bucy syndrome are a consequence of a deficit in learning stimulus reinforcement and associations [26, 27]. For example, the tameness, hypoemotionality, increased orality, and altered response to food are all a result of damage to the normal mechanisms by which stimuli become associated with reward and punishment [20]. Further, others have suggested that if visual stimuli processed in cortical areas do not reach the amygdala, their effective significance will not be registered [28]. This has been supported by several studies. One study showed that the association between stimuli and reinforcement or reward (i.e., food) is weakened in primates that have sustained damage to the amygdala [26]. Others animal studies have shown that when the amygdala is stimulated, the subject stops what it is doing and becomes attentive, followed by responses of defense, rage, or fleeing (subjective feelings) [15].

Below are cross-sectional slices of the amygdala.


superior sagittal sinus
anterior cornu of l lateral ventricle
body of corpus callosum
cerebral v
frontal lobe
body of caudate nucleus
insular cortex
internal cerebral v
lateral sulcus
thalamus
third ventricle
amygdala
temporal lobe
trigeminal cave
uncus of temporal lobe
lateral pterygoid m
parotid gland
internal carotid a
sternocleidomastoid m
suprasellar cistern
longus colli m

(a)


body of corpus callosum
longitudinal fissure
septum pellucidum
body of lateral ventricle
body of fornix
column of fornix
head of caudate nucleus
anterior limb of internal capsule
interventricular foramen
parietal lobe
third ventricle
insular cortex
external capsule
lateral fissure
putamen
temporal lobe
globus pallidus
uncus
extreme capsule
claustrum
trigeminal cave
anterior commissure
trigeminal ganglion
optic nerve
lateral pterygoid m
n V
ambient cistern
n III
masseter m
tensor veli palatini m
medial pterygoid m
n VI
longus colli m
styloglossus m
superior constrictor m
n IX
middle constrictor m
n XII
uvula
platysma m

(b)

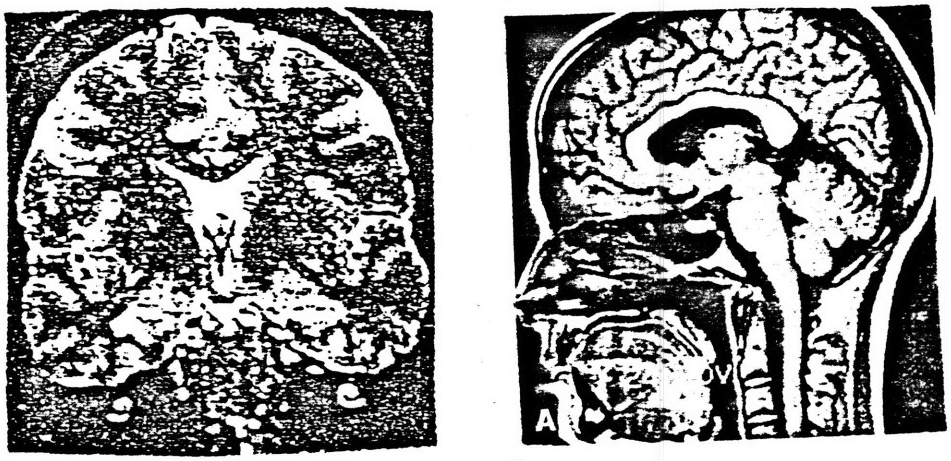


Fig 1-14. Two structural MRIs of the brain: a coronal slice (left) and a sagittal slice (right). From Oldendorf (1991) and Chakers, respectively.

A sample structural image of the brain is shown in Fig 1-14. Since it was first used to create structural images [32], MRI has been used in a variety of applications. In addition to localized spectroscopy [33] and chemical shift [34], MRI of water protons has been extended to NMR angiography [35], perfusion imaging [36], and perfusion imaging enhanced with contrast agents [37]. Most relevant to this project, it has been used to perform functional imaging.

1.4 Functional Imaging

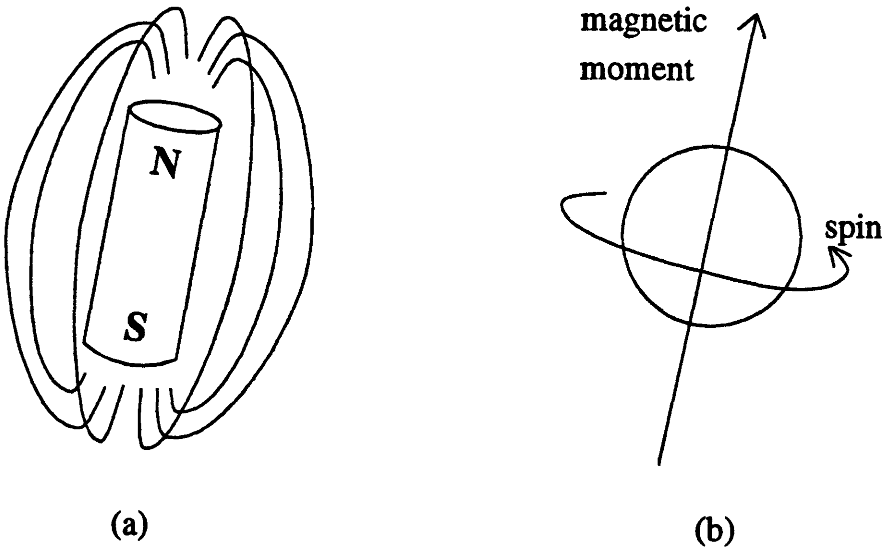
MRI techniques are powerful enough to detect local changes in cerebral blood flow, blood volume, and blood oxygenation levels. This can be accomplished with gradient echo techniques that are sensitive to changes in hemoglobin between oxygenated and deoxygenated states.

The brain possesses anatomically distinct processing regions. Determination of the brain's function in these regions comes partly from an understanding of the processes with which these regions are involved. Previously, this understanding came in the form of regional brain damage and stimulation or electroencephalography. Recently, more powerful methods have been developed for expanding our knowledge of the brain: functional imaging. Functional imaging refers to a class of clinical diagnostic procedures

Further reading is recommended for those interested in understanding the subject in greater detail.

The principle behind Nuclear Magnetic Resonance involves the use of applied magnetic fields and discrete radio waves to identify a particular voxel of tissue by its resonance response, a characteristic determined primarily by a material's magnetic properties.

MRI is made possible because nuclei exhibit angular momentum or spin. Since nuclei bear charges, their spinning produces what is called a magnetic moment-- a vector expressing the strength and direction of the magnetic fields surrounding the nucleus (Fig 1-4). Under normal conditions, such nuclei in a section of tissue will have there spins oriented randomly due to thermal interactions (Fig 1-5). However, when these nuclei (protons for example) are placed in a magnetic field, they will align with the field in one of two energy states: parallel or the higher energy antiparallel (Fig 1-6). In this state, the proton will precess about the field, and its precessional frequency will be proportional to the static field strength and the properties of the nuclei being excited (specifically the gyromagnetic constant), as shown in Fig 1-7.



(a) (b)

Fig 1-4. (a) A simple bar magnet. (b) A nucleus (bearing charges) and its angular momentum (spin) produce a magnetic moment (vector).

When a person enters an MRI scanner, the magnetic fields of the hydrogen nuclei will orient themselves and precess along the strong magnetic fields of the scanner. This orientation may be disturbed by exciting those nuclei with a burst of electromagnetic

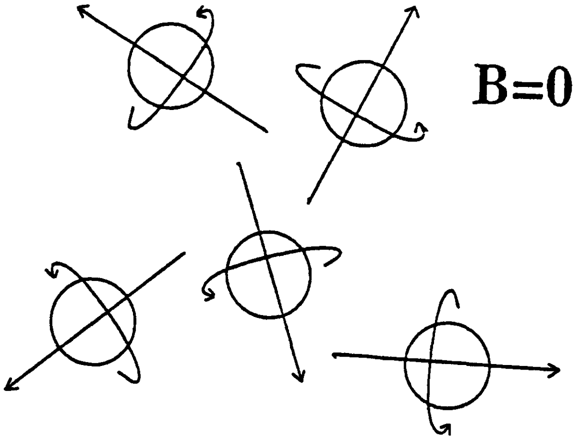


Fig 1-5. The magnetic moments of nuclei are randomly oriented outside the presence of an applied magnetic field.

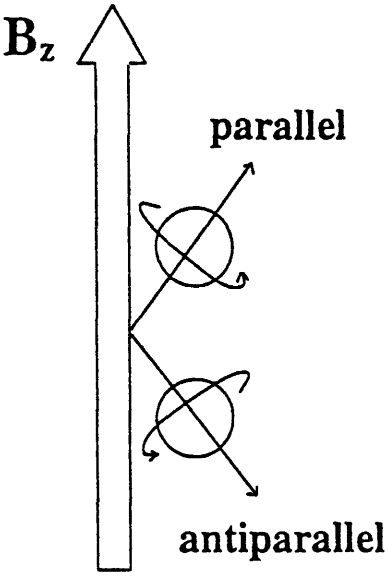
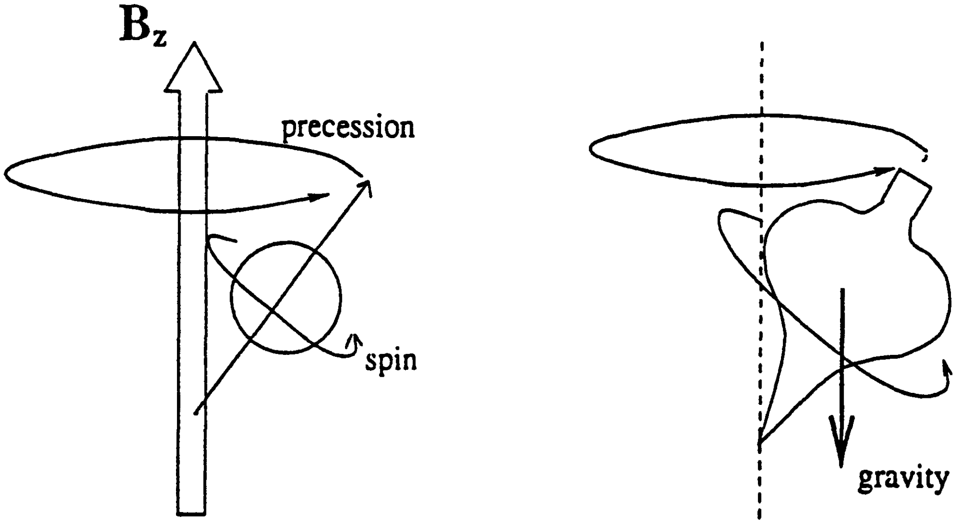


Fig 1-6. When exposed to a magnetic field, B_z, the nuclei in Fig 1-5 will align in one of two energy states, parallel or antiparallel.



A precessing nuclei

A spinning top

Fig 1-7. A precessing nucleus and its magnetic moment are often compared to a top in a gravitational field.

energy. Specifically, with the application of radiofrequency (RF) waves at the Larmor frequency (a value proportional to both the static field strength and the nuclei's precessional frequency), the nuclei can be induced to flip into the higher, antiparallel, energy state. If these RF waves are transverse to the static field, the magnetic moments will be perturbed from their equilibrium orientation. This is a result of the combined effects of both the static magnetic field and the perpendicular, alternating RF fields. Their combined effect can produce a significant change in the orientation of the spin's magnetization, perturbing the spin's axis of net magnetization from its equilibrium position, which is parallel to the static field. Depending on the duration and strength of the RF excitation, the angle of precession with respect to the static field can be rotated

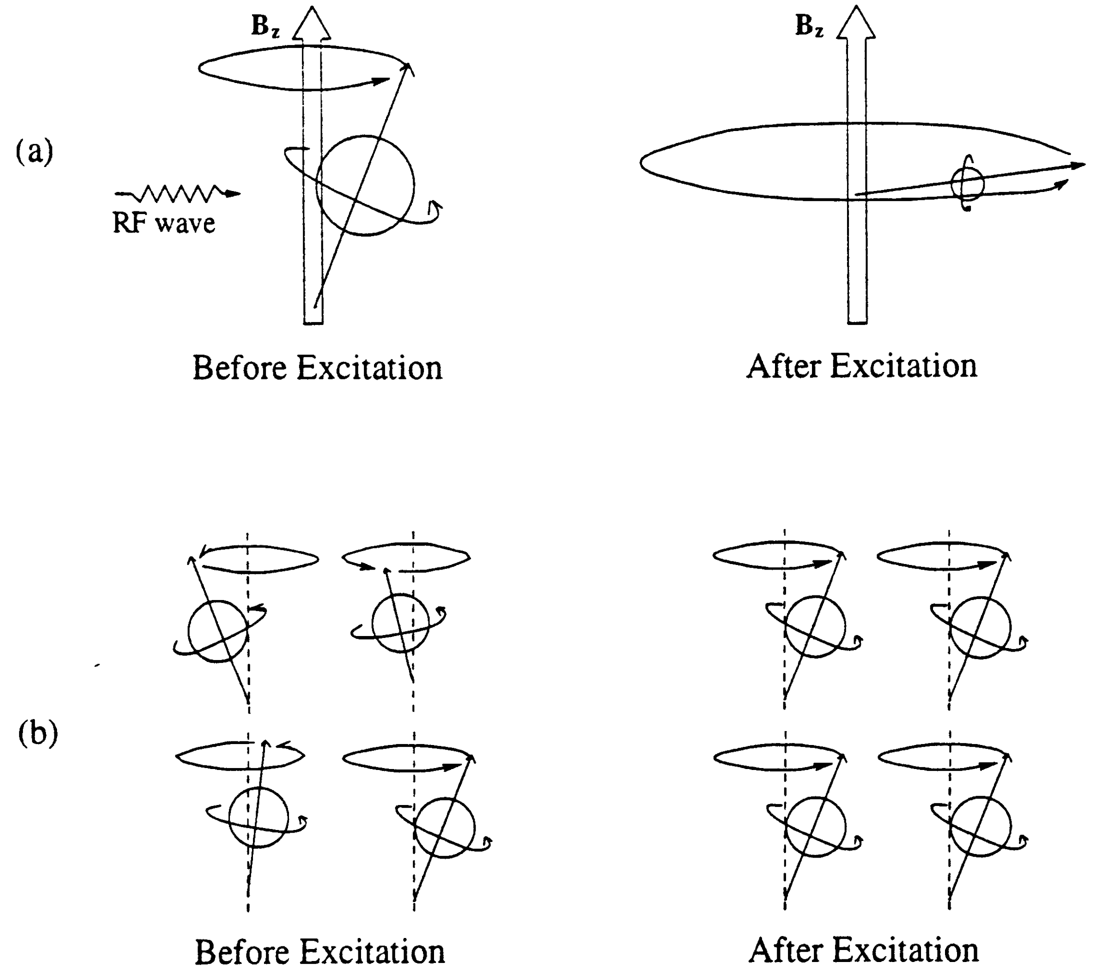


Fig 1-8. The combination of the RF alternating fields perpendicular to the static magnetic fields results in a perturbation of the angle of precession (with respect to the static field). The RF waves also result in the synchronization of the phases of the precessing nuclei.

(flipped) by a specific angle width (such as 90°; refer to Fig 1-8a). A second consequence of the application of the transverse RF fields is a new order in the precession of the nuclear magnetic spins (Fig 1-8b): all spins begin precessing in unison to the new driving force (i.e., they begin precessing in phase).

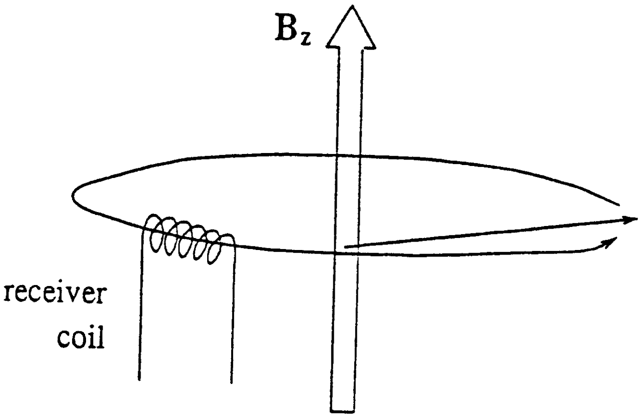


Fig 1-9. Following the RF pulse, the transverse magnetization precesses around the axial external field, inducing currents (signals) in a receiver coil. Similarly, it is alternating currents in such a coil produce the RF pulse.

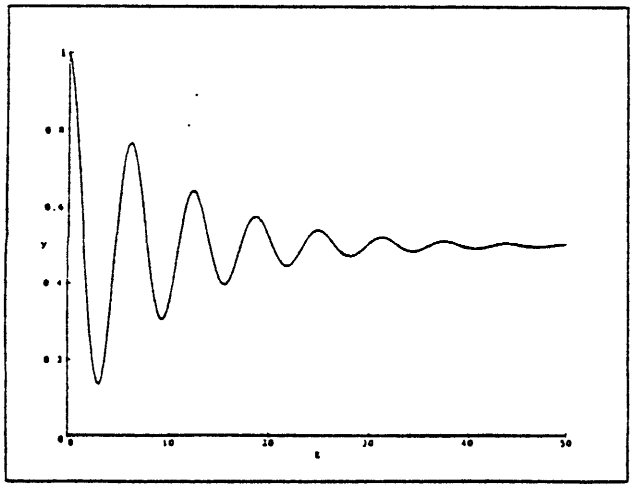


Fig 1-10. Free induction decay following a 90° pulse. The sinusoidal aspect is a result of the magnetic moments (when they are in phase) rotating in the *x-y* plane, passing in and out of the receiver coil.

When the RF waves are turned off, the magnetic moments of the nuclei are subjected only to the static magnetic field, causing the nuclei to return to their previous thermal equilibrium. Magnetic moments can be detected by a receiver coil (Fig 1-9) situated in the transverse plane, which is sensitive to transverse magnetization. With the RF influence removed, transverse magnetization begins to decay, and the change is detected by the receiver coil. This signal is referred to as the free induction decay, which has two characteristic decay constants, T_1 and T_2 (Fig 1-10). T_1 refers to the exponential rate of restoration of the longitudinal magnetization as the energy is lost, and the realignment of the net magnetization with the static magnetic field (including transitions from antiparallel to parallel) as shown in Fig 1-11. T_2 refers to the dephasing of nuclei precession, causing the magnitude of the transverse magnetization to oscillate(Fig 1-12).

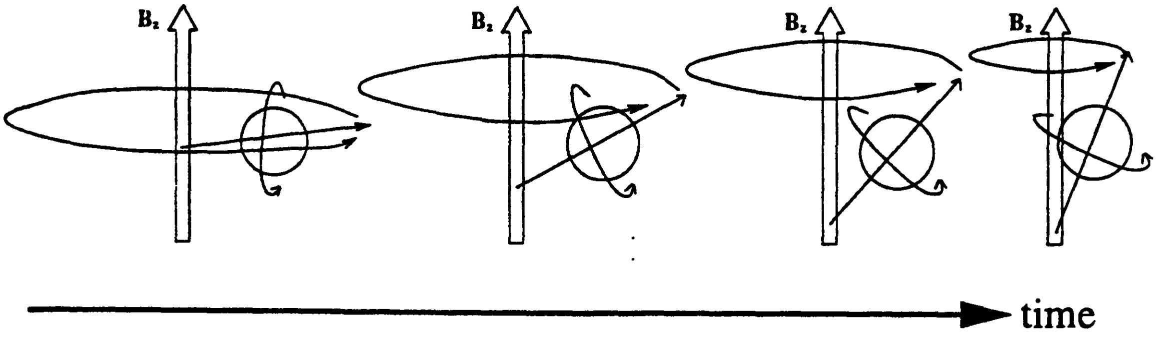


Fig 1-11. The net magnetization of a nucleus gradually relaxes into thermal equilibrium after the removal of the RF field.

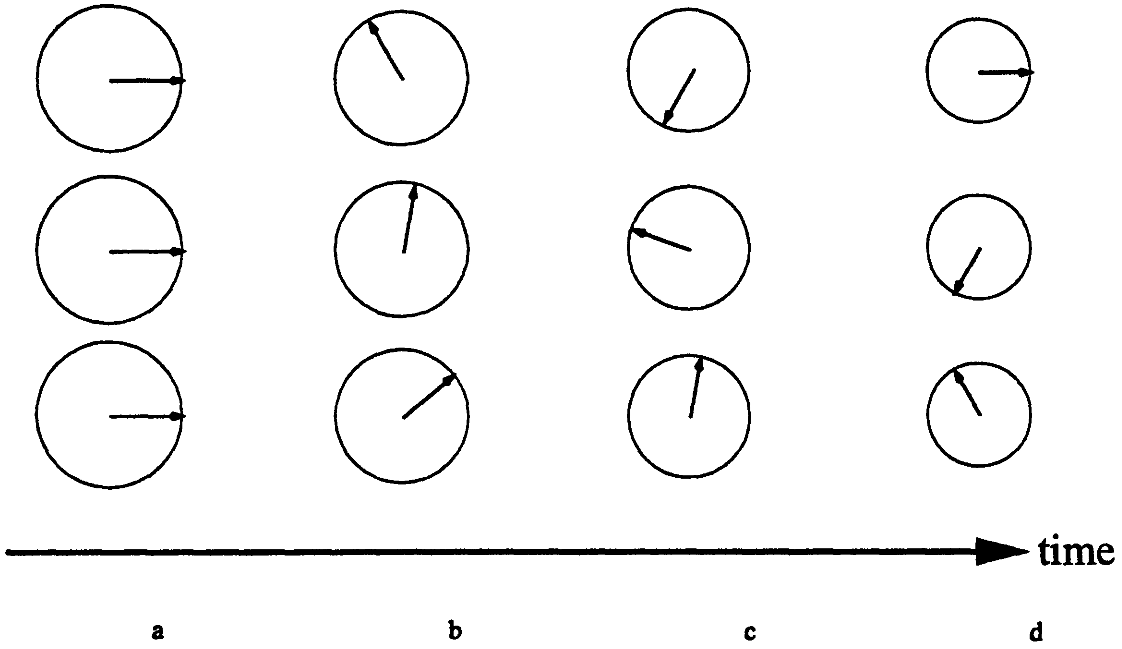


Fig 1-12. Three nuclei dephasing viewed along the *z*-axis, as a function of time, *t*. At *t=a*, there is a net magnetization in the *x* direction. By point *d*, the net magnetization is 0.

Since the RF excitation Larmor frequency is a function of the field strength, if a large gradient is imposed in the magnetic field, a finite slice of tissue can be excited. Since nuclei are sensitive to deviations from the Larmor frequency, gradients of 0.01% per Tesla can excite a thin slice Δz (Fig 1-13). By imposing gradient magnetic fields in the x and y direction of the excited slice during this relaxation, the state of the magnetic field around the nuclei (and subsequently the signal they rebroadcast) will be modified.

This allows for the eventual identification of the properties of a particular voxel within the excited slice. Parameters used to characterize nuclei include the density of excited nuclei (based on signal strength), T_1 (rate of energy loss), T_2 (rate of dephasing), magnetic susceptibility (the ability of a substance to be magnetized by the external field, due to the presence of diamagnetic, paramagnetic, and ferromagnetic concentrations that produce microscopic alterations in the static field), and chemical shift (the change in the Larmor frequency of a nucleus when bound to different sites of a molecule). The receiver antenna picks up the rebroadcast signals, which are then filtered, digitized, and processed by a computer to produce a grayscale image of the slice: a structural image of a cross-section of the brain at a high resolution.

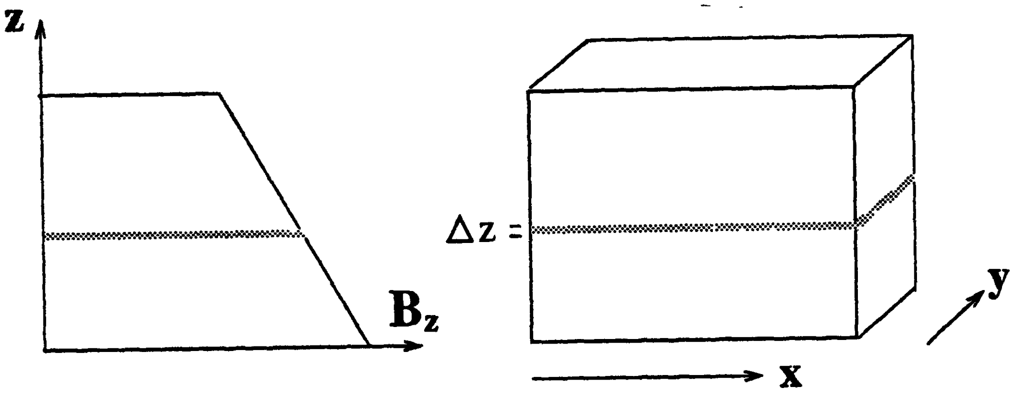


Fig. 1-13. Example of the excitation of a finite slice of material Δz in a section of tissue (right) through the use of a large gradient in the applied magnetic field (left).

MRI can be used to supply a great deal of information about a particular section of tissue. There are a variety of imaging techniques and pulse sequences (the number, strength, and time intervals between pulses) that use the above principles to excite particular nuclei in a particular way, giving the experimenter the specific information about the tissue. Among these are inversion recovery, free induction decay, partial saturation, volume imaging, spin echo, and gradient-echo (see Chakers, chapter 7, for a full review of common pulse sequences). One of the most used MRI techniques is echo planar imaging, which can efficiently sample the spatial information of an object after a single excitation of the water molecules, a process which can last under 100 ms [31]. A variety of head coils can also be used to transmit and receive the RF signals. Of particular note are the head coil and surface coil. The head coil furnishes uniform signal-to-noise ratio for each pixel in the measured region. With a surface coil, the signal-to-noise ratio decreases for measurements made farther away from the surface; it is generally used to measure cortical structures in the hemisphere closest to the coil.

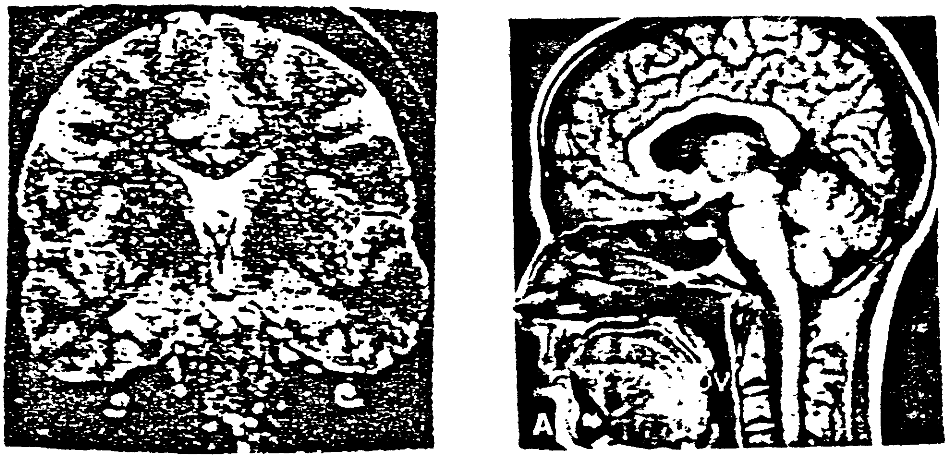


Fig 1-14. Two structural MRIs of the brain: a coronal slice (left) and a sagittal slice (right). From Oldendorf (1991) and Chakers, respectively.

A sample structural image of the brain is shown in Fig 1-14. Since it was first used to create structural images [32], MRI has been used in a variety of applications. In addition to localized spectroscopy [33] and chemical shift [34], MRI of water protons has been extended to NMR angiography [35], perfusion imaging [36], and perfusion imaging enhanced with contrast agents [37]. Most relevant to this project, it has been used to perform functional imaging.

1.4 Functional Imaging

MRI techniques are powerful enough to detect local changes in cerebral blood flow, blood volume, and blood oxygenation levels. This can be accomplished with gradient echo techniques that are sensitive to changes in hemoglobin between oxygenated and deoxygenated states.

The brain possesses anatomically distinct processing regions. Determination of the brain's function in these regions comes partly from an understanding of the processes with which these regions are involved. Previously, this understanding came in the form of regional brain damage and stimulation or electroencephalography. Recently, more powerful methods have been developed for expanding our knowledge of the brain: functional imaging. Functional imaging refers to a class of clinical diagnostic procedures used to map the functions of regions of the brain. Functional imaging techniques rely on the notion that conditions in the brain change during the performance of a cognitive task, a result of the tight relation between neuronal synaptic activity with energy metabolism and blood circulation [38]. Studies have shown that changes in neuronal activity are accompanied by local changes in cerebral blood flow [39], cerebral blood volume [40, 1], blood oxygenation[40, 41], and metabolism[43, 42]. These changes can be used to produce functional maps that show which parts of the brain are involved in performing a particular function. During cognitive task performance, local alterations in neuronal activity induce local changes in metabolism and cerebral perfusion (i.e., blood flow) [44]. Because cerebral hemodynamic states in nonactivated areas are stable over time, a resting state perfusion image can be subtracted from a stimulated state perfusion image to create a new functional map depicting local changes as a result of an activation task [37, 45, 46].

A variety of functional imaging techniques have been used to characterize the brain based on this knowledge. Early experiments used radioactive xenon to observe regional cerebral blood flow within the brain [47]. More recently, magnetoencephalography has been used to map the brain [48]. One of the more commonly used techniques in functional mapping is positron emission tomography (PET); this has also been used to observe regional cerebral blood flow [7, 8, 9, 10, 40].

The first functional imaging of human task activation using MRI were generated by Belliveau *et al.* in 1991 [1]. In this study, observation of localized cerebral hemodynamics was performed by measuring regional cerebral blood volume (CBV) during resting and activated cognitive states, using dynamic susceptibility-contrast NMR imaging of an intravenously administered contrast agent. CBV maps were correlated directly to high resolution structural images of the underlying anatomy, allowing for precise gray-white matter and activated-nonactivated boundaries. Photic stimulation was successful in producing regional changes in blood volume of 32 $^+/-$ 10%, consistent with other studies using other techniques.

Hemoglobin has magnetic properties [49] which suggests it could be used as an endogenous source of contrast material in magnetic resonance imaging. The presence of deoxyhemoglobin in blood changes the proton signal from water molecules surrounding a blood vessel using gradient echo MRI, producing blood oxygenated level contrast [50, 51]. This is because when the normally diamagnetic oxyhemoglobin gives up its oxygen, the resulting deoxyhemoglobin is paramagnetic. The presence of paramagnetic molecules in blood produces a difference in magnetic susceptibility, increasing the number of spins in the region observed by gradient-echo techniques.

By accentuating the effects of this natural contrast agent through gradient echo techniques, studies were able to produce images of blood vessels in the brain [52]. The experiments showed deoxyhemoglobin acts as an endogenous paramagnetic contrast agent and that changes in its local concentration lead to changes in the T_2-weighted MR signal [52]. This experiment suggested that real-time maps of blood oxygenation levels in the brain could be produced [52]. Such a study was performed by Kwong *et al.* in 1992; it represented the first use of rapid MRI without contrast agents to observe human brain activity [2]. The experiment observed human primary visual and motor cortex in response to cognitive tasks. Techniques to detect such changes were developed using high-speed echo planar imaging. Changes in blood oxygenation were detected by using a gradient echo sequence sensitive to the paramagnetic state of deoxygenated hemoglobin. Blood flow changes were calculated by a spin-echo inversion recovery, tissue relaxation parameter, T_1-sensitive pulse sequence. In each case, deoxyhemoglobin acted as an endogenous paramagnetic contrast agent, and changes in its local concentration led to changes in the T_2-weighted MR signal. Activity induced changes in the brain were observed in subtraction MR images (activated minus baseline) at a temporal resolution of seconds.

Since then, the model has been continuously improved. Presently, fMRI allows us to observe activity within a particular voxel of tissue at a high temporal resolution. It is superior to other imaging techniques, such as PET (see Fig 1-15), X-rays, electroencephalograph, and radioactive tracers (such as xenon) in terms of overall temporal (< 0.1 s) and spatial resolution (< 1 mm). More importantly, it serves as a non-invasive method for studying human subjects, allowing for repeated studies. The superiority of using MRI is evidenced by the fact that there are presently far more MRI facilities than its closest competitor, PET.

Typical fMRI data measures signal intensity within a slice of the brain over a finite period of time, sampling at a set interval in response to an activation task. Typically, a structural slice is selected for observation, and every voxel within the slice is sampled (Fig 1-16). The experimenter can use the structural image to pick a particular voxel to observe changes in signal activity (or brain metabolism) within a specified region of the brain. Most functional experiments sample during a paradigm (an experimental protocol for presenting stimuli), which alternates between period of neural stimulation and rest (Fig 1-17). Most cortical experiments observe changes of +/- 2% just over of baseline between activation and rest.

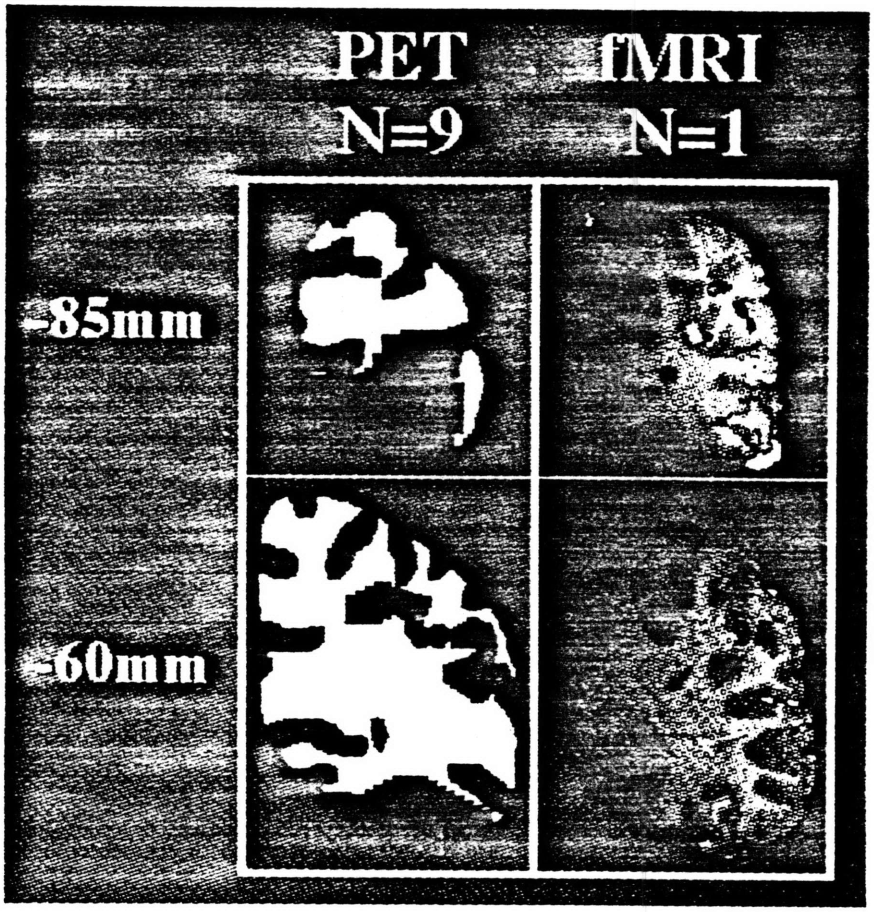


Fig 1-15. Comparison of functional data from a nine-subject PET study (left) and a single-subject MRI study (right). From [29].

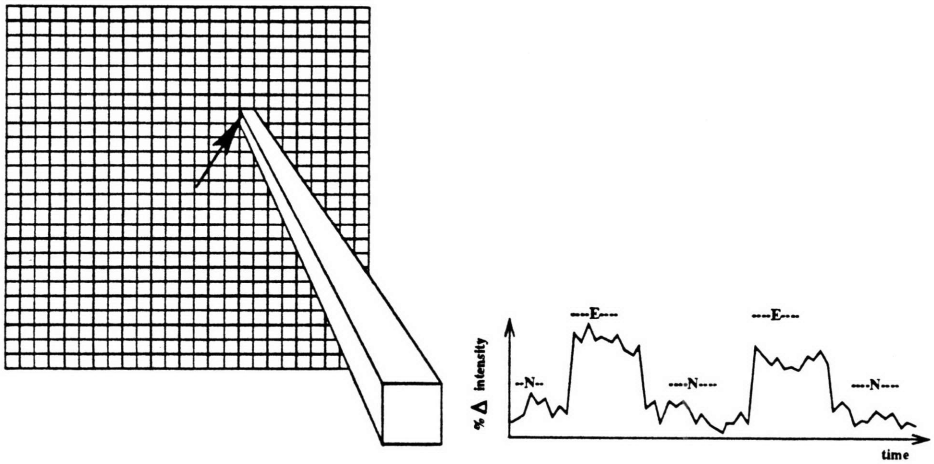


Fig 1-16. Illustration of fMRI of a single slice of tissue and the observation of the intensity as a function of time for that pixel.

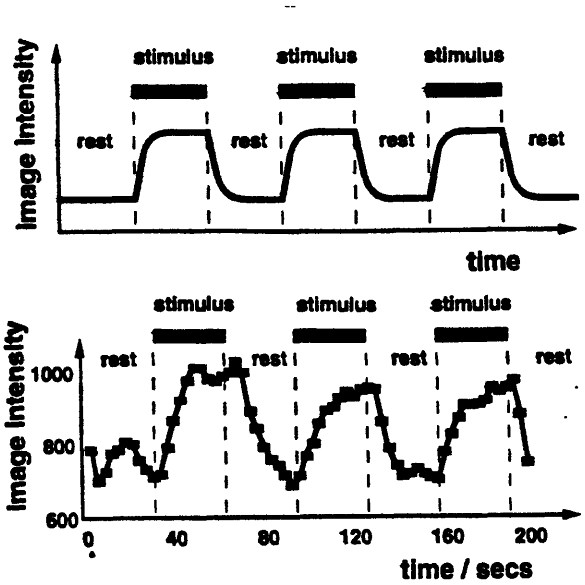


Fig 1-17. Idealized time course (top) and actual data (below) in response to periods of stimulation and rest. From [29].

Of considerable importance for the experimenter is picking out the correct region (i.e., selecting the correct group of pixels from a map of the brain) for observation. Head movement can create artifacts, the result of movement of tissue in and out of a voxel. This is most prominent at the borders of the region, where slight differences in tissue magnetic characteristics can alter the measured intensity. A significant amount of research has been conducted concerning this issue [53, 54] with some concluding that the dissimilarity of the characteristic response within each group of pixels is maximized when analyzed in clusters, increasing intra-group homogeneity.

Chapter 2

System Design and Experimental Methods

2.1 The design problem

The MRI laboratory of McLean Hospital is equipped with a General Electric Signa 1.5 Tesla Magnetic Resonance Imaging Scanner. The scanner has been retrofitted with an Advanced NMR Systems echo planar imaging coil. The MRI scanner allows the researchers at McLean to observe functional imaging on the order of fractions of a second. The laboratory currently has no system for conducting cognitive paradigms, other than the manual presentation of audio stimuli and/or instructions over a microphone. It requires the experimenter to record the time of presentation with a stopwatch and to later manually coordinate those times with the recorded functional data.

Such a system may be useful for a narrow range of cognitive experiments, such as certain speech, hearing, or memory experiments, where the presentation of a single stimulus (i.e., the reading of a sentence) occurs over several seconds. But it is less than adequate for many others experiments, such as those requiring the quick and repeated presentation of stimuli (such as tones and images). At the very least, the present system does not take full advantage of the temporal resolution and accuracy of the MRI scanner. Specifically, a system which presents stimuli electronically would improve both speed and precision, allowing the researcher to better pinpoint the moment that a particular stimulus occurs in analyzing the recorded activation. It would also be of benefit to experiments in which the presentation of a great number of stimuli occur over a short period of time, and where the response to these sets of stimuli are averaged. Further, a broader range of experiments could be performed if the laboratory were set up to present a broader range of stimuli. Finally, the computerized presentation of stimuli would avoid the needless and error prone method of recording the time of stimulus presentation, give better consistency over experiments, avoid needless record keeping, and increase overall user efficiency.

The approach to this problem was to design a system for the computerized delivery of audio and visual stimuli to subjects in the MRI scanner, in which the delivery of the stimuli is electronically coordinated with the acquisition of a set of scans. In meeting the above standards, it was proposed that such a system should fulfill the following goals:

(1) The amount of time for the computer to generate and present stimuli should be minimal, allowing the user to pinpoint the moment of stimulation in a set of scans.

(2) The system should be flexible, allowing for both a wide range of stimuli and paradigm, as well as room for expansion to accommodate other system upgrades.

(3) It should be relatively easy to learn and use.

(4) The system should be self-contained (i.e., control of the presentation should be kept within the computer).

(5) It should exert no control over the scanner, but rather should be dependent upon the scanner for timing.

(6) It should be cost efficient, utilizing equipment already present in the laboratory.

2.2 System Design

The following components were already present in the laboratory at the onset of the project: the General Electric Signa 1.5 Tesla Magnetic Resonance Imaging Scanner equipped for the acquisition of high-speed functional imaging; a cassette deck that is wired to headphones worn by patients during MRI; a two-way intercom between the patient (via the headphones) and the MR imaging control room; and a Resonance Technologies projector, which takes a National Television Cable Standard (NTCS) feed delivered along a co-axial cable and projects video images onto a two-way screen which can be viewed by the patient on a mirror in the head coil. The laboratory is also equipped with Sparc workstations and Macintosh computers. Every effort was made to utilize all available equipment. In fact, all of the equipment listed above was incorporated into the final design.

The system developed was constrained by the equipment that was available; of particular importance was the scanner. The system must allow the user to pinpoint the moment in a set of scans when a stimulus is presented, and it must do so independent of the scanner operation (i.e., the system should exert no control over the scanner). Image acquisition normally begins with a set of scout scans and is followed by functional imaging, while scanning at preset intervals for a preset duration, as shown below. Much of the success of the system depends on its ability to identify the moment at which these scans occur, and then deliver stimuli as quickly as possible.

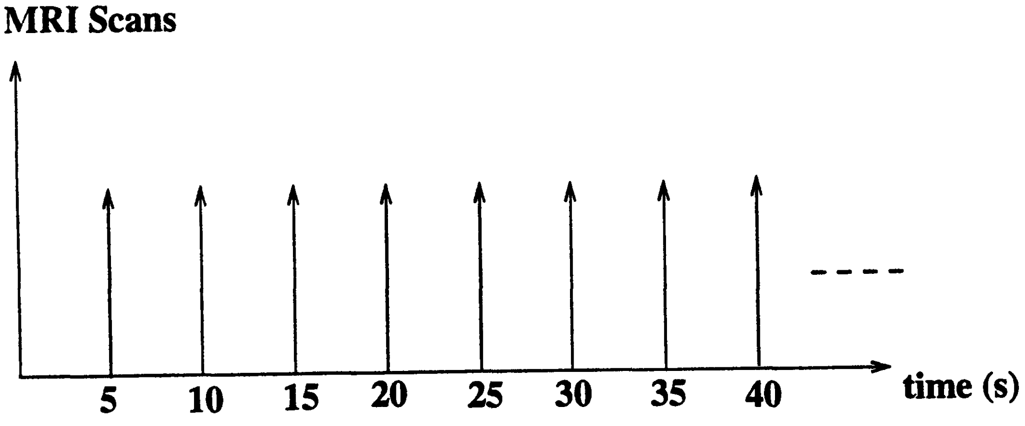


Fig 2-1. Single scans as a function of time, with each scan occurring every t_0 seconds. The scanning interval and duration is set before data acquisition.

As stated, one of the design goals require the presentation of stimuli to be as rapid as possible and for the system to be self-contained. In keeping with these goals, the designed system was centered around a computer. Using a computer gives the ability to present the stimuli electronically and thus quickly. It also gives the flexibility to store and present a variety of stimuli, as well as centralize control of its presentation.

Finally, modern computers are both universally familiar and cost-efficient, offering the advantage of developed software and equipment.

Because the lab already possessed a variety of Macintosh computers and programs, a Power Mac 7100/66 with a 14" monitor was selected. It is among the fastest Macintosh computers available at the present time, and offers adequate room for expansion. Depending on system performance or future project goals, this could include increasing RAM, the addition of a removable hard drive, or upgrading the video card. At the time the system was designed, it was determined that none of these options were necessary for upcoming projects. It was also decided that the purchase of a CD ROM for this computer would be unnecessary. At present, CD access time is slow and thus would not be utilized during experiments. Other CD ROMs are available in the lab for the importation of picture files from CD. Finally, a lead shield, much like the one used by other computers in the MRI control room needed to be built around the computer. The shield is important in giving adequate viewing of the Macintosh monitor, since Magnetic fields can skew the screen image (see A.4).

Since one of the project goals stipulated that the system to be designed exert no control over the MRI scanner, it was precluded that the system be *dependent* on the user and the scanner for information about the scan time and interval. In other words, the system cannot exert any control over the initiation or timing of a set of scans. Rather, the initiation and timing must be controlled through the scanner control room; this information must somehow be fed to the system. One way of doing this is by reading information from a trigger located on the scanner machinery, which produces a standard 5V TTL pulse at the onset of a single scan. A digital input/output board located inside a computer would allow the system to identify the moment that a scan is initiated and to keep track of the number of scans that have already occurred. Feeding off the trigger in this way should not affect the scanner performance.

There were several considerations to be made in selecting an I/O board. The first was room for expansion. Only one line is in fact needed -- the one receiving the signal from the trigger. Extra digital and counter ports can be used for other purposes, for example, operating a joystick or communicating with other hardware. A second consideration is whether to use digital I/O hardware or counter-timer hardware. Using a digital port would require writing code to identify the pulse and would offer little advantage over using a serial port. Such a method would require the interruption of data collection for the processor to identify the pulse, and a great deal of programming. Conversely, the counter-timer can perform a variety of functions, among them event counting. Using a counter-timer would require no programming for the identification and counting of a pulse; the ability to do so is built into the hardware. The I/O board selected for the system was the NB-TIO-10 from National Instruments. The ND-TIO-10 comes with ten timer-counter ports and two 8-bit digital I/O ports. It offers the speed and ease of counter timers each with a 65,536 count, as well as a pair of digital ports for system expansion. The board is compatible with several Macintosh programming packages, requiring only a few lines of code to access the counter.

The computer must have some means for delivering stimuli to subjects during imaging. Audio and visual stimuli can be imported by disc in the form of a file (e.g., PICT picture files, Quicktime motion video files, and sound files) and stored in the hard drive. Macintosh computers come with a wide range of sound capabilities, including a 16-bit sound card, an RCA sound port, and standard system sounds (beeps, bells, and whistles). Audio output from an RCA jack on the Macintosh can pass through the cassette/tuner and be heard through the headphones in the scanner. As for visual stimuli, pictures can be scanned in using a color scanner and the Adobe Photoshop software package already located on another computer in the lab. Originally, TelevEyes Plus, an external device which takes the signal fed into the monitor and creates a second, identical NTCS signal, would be used to create a signal that passes through a VCR, and then to a Resonance Technologies projector, where it would be viewed by the subject. The purchase of an Adlus Video Card made this unnecessary, providing the same function as the TelevEyes software, except that it delivers the images of a second internal screen.

The final aspect of the system design was the software used in its development, and the Macintosh offers as many software options as its PC competitors. Software is necessary to take data from the counter-timer, acquire information from the experimenter, decide which stimuli to present and when to present it, and to actually present that stimuli. Programming is the only viable option, in terms of reading the counter-timer and giving an audio/visual presentation. Powerpoint and other media presentation programs cannot access information off the I/O board. Also, other forms of media programming, such as Hypertalk, do not take advantage of the software that accompanies I/O boards; though easier to use, they offer another level of programming architecture and will consequently decrease performance. Think Pascal and Think C are compatible with the software package of the digital I/O board and offer access to the Macintosh Toolbox. Though programming in these languages is time consuming and tedious, it offers advantages in terms of power and speed. The more familiar Think C version 6.0 was used for system development. It does, however, have one drawback: at present, there is no native programming software for the Power Macs. Though reviews vary, nonnative software is believed to run almost as fast on the Power Macs as it does on a native upper end Quadra. When upgraded to native, it is expected to run two to three times as fast. Think C seems to hold the best chance of being distributed as a native version first (an overview of the Macintosh programming and Think C is contained in Appendix A.1 and A.2).

While the system developed should be powerful enough to quickly perform a variety of tasks, it should also be flexible enough to perform any individual paradigm. To achieve this flexibility each experiment has a script (a set of instructions unique to the experiment) for the program to follow. The script contains only the necessary information for the program to do its job. This includes the following for each set of stimuli: the type of stimuli, its (file)name of the particular stimuli, and the scan number after which it should it be presented, following its appropriate time delay. A script file can be easily created with an editor and later accessed by the stimuli presentation program during an experiment. Such a setup should be simple and easy to use. By separating the creation of a script file from the presentation program, it allows the user to create a paradigm on any computer and keep a hard copy of the file to be used for every repetition of the experiment. A sample script template is contained in Appendix C.1.

2.3 System Development

System development proceeded in two stages: hardware implementation and software development.

2.3.1 Hardware Implementation

Hardware implementation started with the purchase of selected components and integrating them into the system. The Macintosh computer was placed in the MRI control room with both the video board and I/O board installed, along with a protective shield surrounding both screen and monitor. An RCA cable was used to connect the audio output jack on the Mac to the tuner in the control room (which has an audio feed to the headphones in the scanner). Co-axial cabling was connected from the video board on the Macintosh to the Resonance Technology projector in the scanning room. The cabling was routed outside the scanning room in an attempt to avoid any effects from the magnetic fields. Finally, a connection from the trigger on the scanner machinery to the counter-timer port on the I/O board was made with a coaxial cable with a LIMO connection on the scanner end and a standard 50-pin connection (wired to the source input of counter-timer number one and the ground connection of the I/O board) on the computer end.

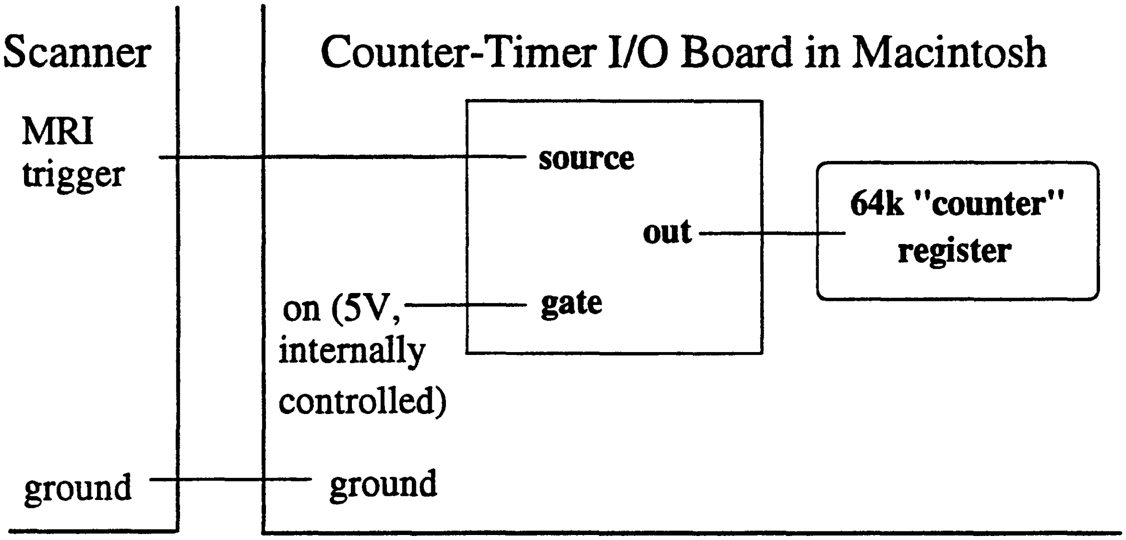


Fig 2-2. Schematic of the counter-timer connections. Refer to Appendix A.3 and Fig A-1 for more information.

Finally, the scanning room at McLean has a large skylight through which the magnet was installed by lowering it through the ceiling. The skylight makes the room difficult to darken, presenting logistical problems as the projector and screen are clearest when the room is dark. In order to solve this, a tarp was placed over the skylight

Testing of the hardware followed. A small program (contained in Appendix B.3) was written in Think C in order to test the board. The program, when initiated, resets the board, sets the counter memory to zero, and programs counter-timer number one for event counting. When the scanner begins scanning the counter timer counts each scan and holds the count number in the counter register. Concurrently, the program runs through a loop that continually accesses the value in the register. If that number changes, the new count number is displayed on the Macintosh monitor. The program was tested with a set of dummy scans; each was correctly identified and counted, thus verifying correct operation of the program.

Similarly, the performance of other aspects of the system were also verified. One program (contained in Appendix B.4) displays a picture that can be viewed on the projector screen and used to assess the performance of both the video card and the projector. Delivery of sound was verified simply by having someone wear headphones while one of the system chimes was played.

2.3.2 Software Development

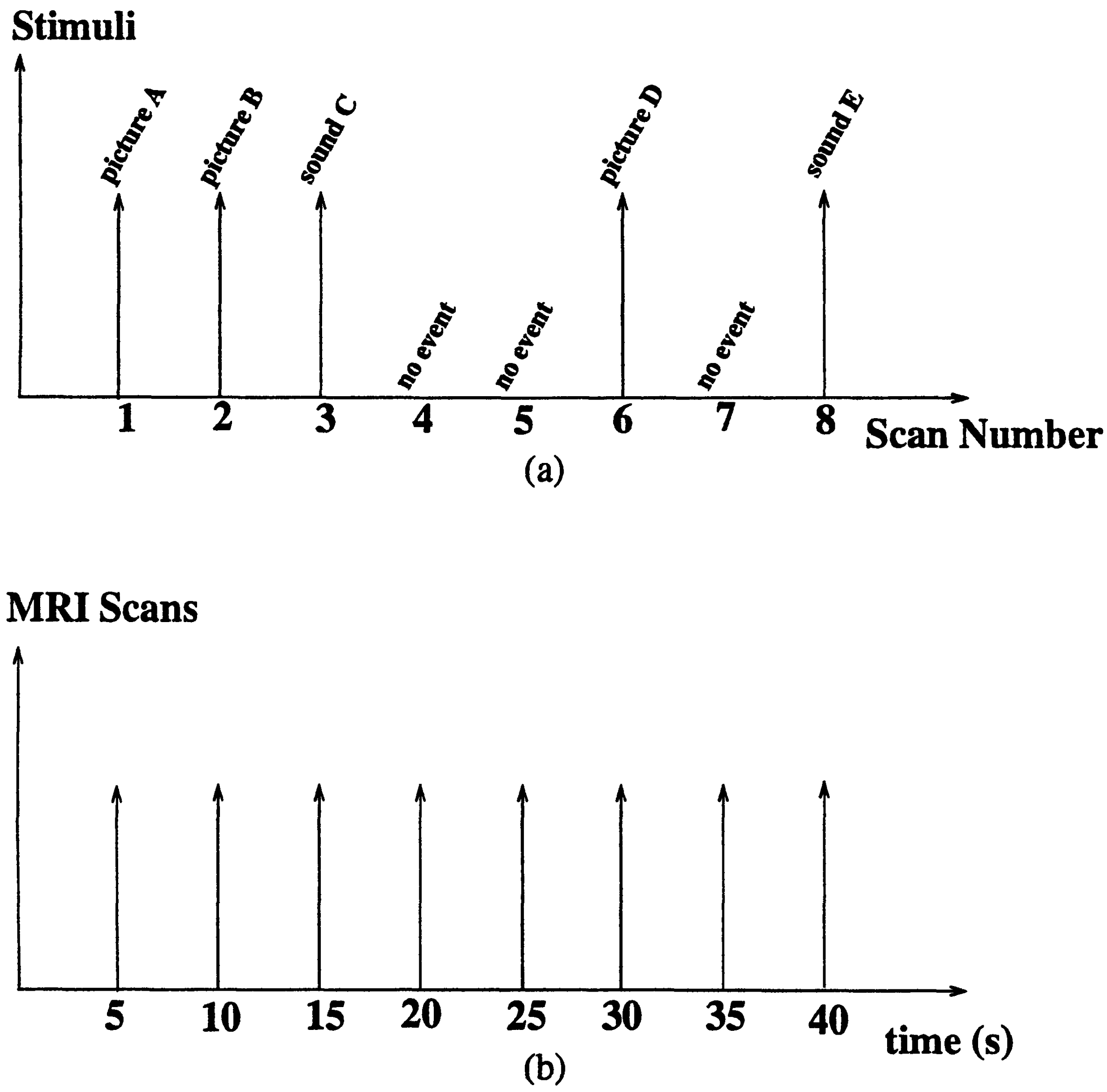
The second and more time consuming aspect of the system development was writing code for the program that would operate the system (an introduction to programming on the Macintosh and Think C is contained in Appendix A.1 and A.2). As stated, the project file, MRIgsp.π, was created with Think C v.6.0 for the Macintosh. A resource file, MRIgsp.π.rsrc was created using ResEdit. The resource file creates a 640x480 pixel plain box window to the left of the screen. This is the region presented by the Radius video board, which feeds motion video to the Resonance Technologies projector.

Four resource and library files were used in creating the program: MacTraps, ANSI, NI_DAQ_MCAL1, and ChkErr.c (the first two accompany Think C, the latter accompanies the National Instruments I/O board). MacTraps gives access to the functions in the Macintosh Toolbox, while ANSI was used for displaying text and working with strings and pointers. ChkErr.c and NI_DAQ_MCAL1 were used by the program to access the board and correctly gate off the scanner. The source file, MRIgsp.c, is the heart of the stimuli presentation program and was written by the author. The project file is divided into two segments: ANSI in the first, the other four files in the second. MRIgsp.π, when compiled, creates the stand alone program that is used for the gated presentation of stimuli doing fMRI.

The source code for the project was contained in the file MRIgsp.c and is the heart of the stimuli presentation program. A copy of the source code is contained in

Appendix B.1. One of the goals of the presentation system is to reduce the time required to present the stimuli to a minimum. It is therefore advantageous to do as much housekeeping as possible before the experiment begins (i.e., all computations that can be done before the scanning starts are done in advance). The program begins by including library files, defining global variables and arrays, and initiating the Macintosh Toolbox, all of which are used to perform other tasks by the program. The program follows with a brief text introduction and pointers to documentation on how the program works. It allows the user to exit if he or she is not yet ready to perform an experiment

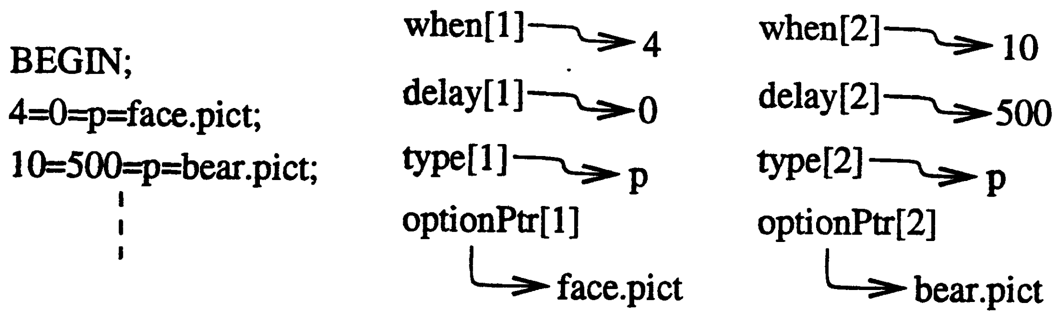
The program then asks for the name of the file containing the script to be used in the experiment. As described earlier, the script contains the scan number after which a stimulus occurs, the type of stimulus (identified by a single letter), the (file)name of the stimuli, and the delay it should follow after the initiation of the scan. The script file is created as a simple text file, a sample of which is contained in Appendix C.1 (see also Fig 2-3). The program opens and reads the script file, reading the file twice. The first time it reads the file, it counts the number of stimuli events (note from the sample file in Appendix C.1 that a stimulus event is designated as one line ending with a semicolon after the begin line). The program assigns a numerical order to each set of stimuli, such that the first line is identified as event number one, the second line is event number two, and so on. The second time the file is read, the program goes through each event line one at a time, incorporating the information in each line into one of four arrays. The scan number and delay value are placed in an array of integers, when[*n*] and delay[*n*], respectively. The single letter identifying the type of stimulus is placed in the character array, type[*n*]. A string is created from the stimulus (file)name, and is pointed to by one of the pointers in the pointer array optionPtr[*n*]. For each array, the value *n* corresponds to the stimulus number. For example, the character array when[*n*] refers to the scan number after which the *n*th stimuli should occur (i.e., when[1] refers to the scan number after which the first stimulus should occur, and when[2] refers to the scan after which the second stimulus should occur). Similarly, type[2] identifies the type of stimulus, optionPtr[2] points to the name of the stimulus, and delay[2] holds the value of the time delay, all for the second stimulus event. The program refers back to these strings later, when it is time to present the stimuli.



(a)

(b)

Fig 2-3. Example of the relation between scanner operation and the corresponding events designated in the script file. The scanner is set so that the brain is scanned at a predetermined interval, such as once every five seconds as shown in (a). The corresponding stimuli are shown in (b). For example, after scan number one (which occurs at $t=5$ seconds, picture A is presented. After scan number two (at $t=10$ seconds), picture B is presented. Some scans are not followed with stimuli, such as scan number four and five. Also, the presentation of some stimuli may be delayed after the recognition of a scan. For more information on the script files, see Appendix C.1



```
BEGIN;
4=0=p=face.pict;
10=500=p=bear.pict;
```

Fig 2-4. Sample lines from a script file and schematics of arrays. The first two events in a script file are shown; also shown are the values, characters, and strings that are held by the first and second values of the four arrays.

Other computations can be performed before the experiment begins. The window created in the resource file is opened off the screen and in the region to be viewed by the subject. The menu bar is removed (it is not be accessible during the experiment). The program asks the user if he or she has completed all preparation for the experiment. This includes the placing of the subject in the scanner, adjusting the view screen, and completing all preliminary or scout scans. When the user is ready, the board is reset, its register set to zero, and its configuration is set for event counting. The program then enters the stimuli presentation loop, and the scanning sequence is initiated .

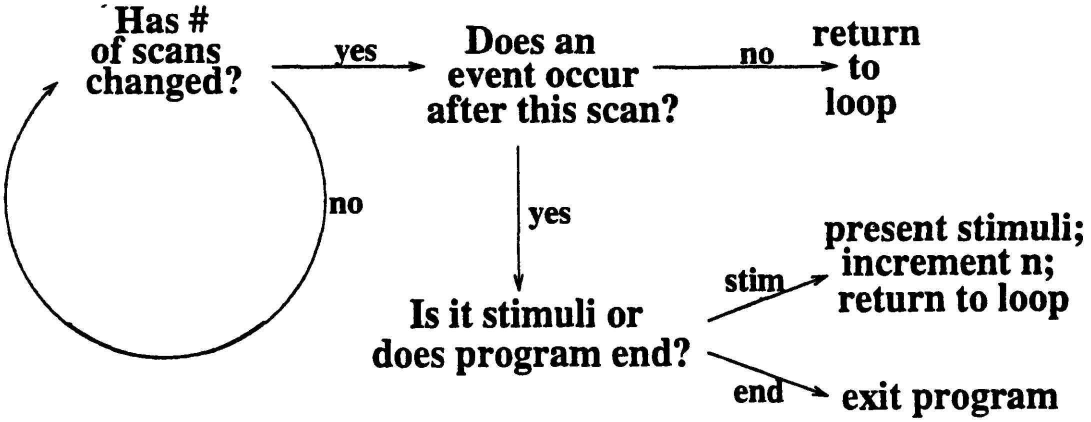


Fig 2-5. Fundamental code schematic.

A series of programming loops are used to present the stimuli. Note that when the program first enters the loop, the scanner has not yet begun scanning. When the program enters the stimulus presentation loop, *n*, the event number, is set to $n=1$; this means the program is set to present the first stimulus. The loop (which is a "while" loop) continually accesses the register of the counter timer, which holds the value of the number of scans that have occurred. The program remains in this loop until the value of the register changes; this happens when a scan occurs. Thus, while in the loop, the program continually performs two functions: accessing the register and checking to see if the value has changed. This is ideal: the less time spent in this process, the faster the computer can respond to a change in the count. When a change in the count does occur, the program checks to see if a stimulus is presented after this scan. It does this by comparing the register count (the number of scans that have occurred) with the value in when[*n*] (which holds the scan number after which the next stimulus should be presented). If a stimulus should not be presented in this scan, the program returns to the beginning of the while loop and continues to access the register, and check if another scan has occurred. If a stimulus is to be presented, the program then waits the amount of time specified in delay[*n*]. The program then proceeds to present the type of stimulus identified by type[*n*]. If the type of stimulus is a picture, then the PICT file named in optionPtr[*n*] is displayed in the region viewed by the subject. If the stimulus is a system sound, the sound identified in optionPtr[*n*] is presented. If the type of stimulus is a blank screen or the end of the program, then those functions are performed. If the type is an "end", the program exits the loops and terminates. If the event is not an "end", the program increments *n* (so that the next stimulus in the script is be presented), returns to the beginning of the loops, and repeats the process until all stimuli have been presented.

The advantage of this system is that the program spends minimal amounts of time accessing the counter-timer register, deciding if a scan has occurred, and deciding if a stimulus should be presented after a scan. No processing time is devoted to recognizing a pulse, since that function is performed by the board hardware. Nor is any time devoted to event tracking (checking for the occurrence of a keyboard stroke, a mouse movement or click, the insertion of a disk, or changes in another program or device).

The program is also designed to either avoid or alert the user of errors. When running, the program allows the user to exit the program before the experiment begins. It alerts the user if an incorrect file name has been entered. The program scrolls through the list of stimuli contained in the script to be presented, allowing the user to recognize the file that will be used. The program also alerts the user of errors in a script file and avoids some errors such as extra spaces. Flags go up whenever there is an error in board operation, or when opening a picture file.

One way to avoid errors during an experiment is to perform a trial run of the stimuli and script in advance. To accomplish this, another program was written to simulate the experiment without depending on the scanner or scanning chamber (the source code is contained in Appendix B.2). This second program is essentially the same as the presentation program, with two key differences. The first difference is that the visual stimuli are presented in a window that fills the computer screen. Second, instead of gating off the scanner, the program asks the user how often would scans occur if in fact the scanner were actually running. It then simulates how the scanning and stimuli would occur using an internal clock.

2.4 System Summary

2.4.1 Operation

In summary, the system works as follows. Visual and audio stimuli are loaded into the computer via diskette in the form of PICT files or audio files. These files can be created using other computers in the lab, one of which uses a microphone to create sound files, or another which uses a scanner to create PICT files. An experimental paradigm is designed and written in the form of a text file. This text file lists all events that occur during an experiment, containing what stimuli should be presented and when they should be presented during the scanning process. The following parameters are also designated: the section to be imaged, the duration and interval of each scan, etc.

Experiments were undertaken after all preliminary work was completed. The subject was placed in the scanner, with either a head coil or surface coil, and the mirror was adjusted so that the screen could be viewed. After scout localization scans, which allow the user to identify the region of observation, the stimuli presentation program was run. The program asks the user which script file is to be used in the experiment, then begins counting the number of scans occurring. At this point, the technician begins the experimental scans, which occur at a set interval over a finite period of time. The counter-timer begins counting these scans, while the program runs a continuous loop that reads the value of the counter (i.e., the number of scans that have occurred). That number changes each time a scan occurs; the program then determines if the stimuli conditions should change during that scan. If a stimulus is to be presented, the program sends an audio output to the headphones, or a video output through the video board to the Resonance Technologies Projector, where it can be viewed on the two-way screen by the patient.

2.4.2 Design Schematic

A schematic of the final system design is shown below.

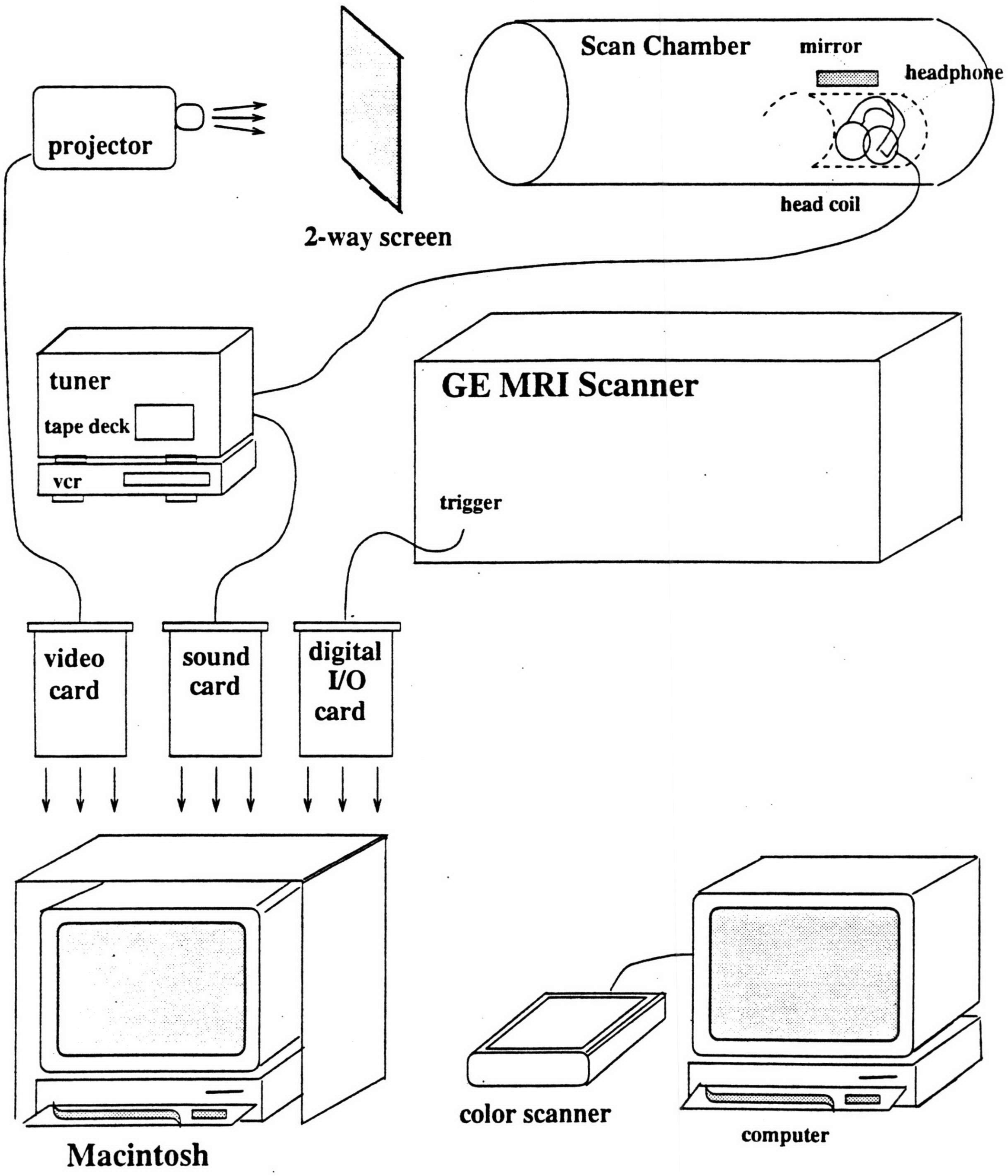


Fig 2-6. System apparatus

2.5 Experimental Implementation

. The system was implemented using a cognitive experiment. The experiment had two goals: 1) to utilize the stimulus presentation system, and 2) to observe functional activation of the amygdala.

2.5.1 Experimental Methods

As discussed earlier, the amygdala serves as an emotional center of the brain. Thus, to observe activation, it is necessary to generate periods of activity and non-activity in the amygdala. This means creating alternating periods of emotional and unemotional states (as shown in Fig 2-7). The stimuli used to create these states were emotionally charged and emotionally neutral pictures. Three sets of emotional stimuli were used. The first two sets are best described as horrific and pleasant (or nice). These pictures were selected for their apparent horrificness or pleasantness. The pictures were rated by an independent set of subjects using a scale from +5 to -5, with +5 representing horrific (or pleasant) as an accurate description of the picture, and -5 as an inaccurate description (this work was conducted by Dr. Deborah Fein, Professor of Psychology at the University of Connecticut). (Most of the pictures that were used received an average rating of +3.0 or higher.) The third set of pictures are referred to as facial pictures and have been used in other cognitive experiments. (These were developed by P. Eckman of the University of California Human Interaction Laboratory.) These pictures include faces expressing one of four emotions: happiness, sadness, anger, or fear. For the remainder of this discussion, these three sets of emotional stimuli will be referred to as nice, horror, and facial stimuli.

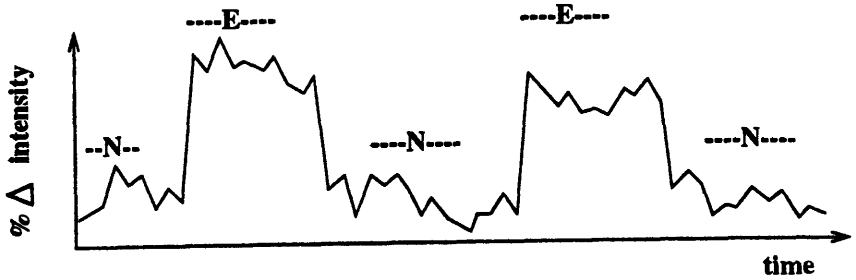


Fig 2-7. Idealized experimental results: alternating periods of activation and lull in response to emotional and unemotional stimuli.

These stimuli, which were used to stimulate emotional processing, were contrasted with emotionally neutral stimuli. Brick walls were used for neutral stimuli. Other stimuli (such as those depicting symbols, writing, faces, or animals) were not chosen because they were believed to be more likely to produce emotional responses. A blank screen was also chosen against, since it might provide a the lack of stimuli, instead of emotionally neutral stimuli.

A paradigm was designed in cooperation with Dr. Deborah Fein in order to achieve the idealized results shown in Fig 2-7. A single experimental run consisted of a series of alternating periods of emotionally charged and emotionally neutral stimuli, as shown in Fig 2-8. Periods in which neutral pictures were presented to the subject (*N*) were followed by periods in which pictures from one of the three emotional sets were presented (*E*). During each neutral (*N*) period, four neutral stimuli were presented. During the emotional (E) period, four emotional stimuli were presented, all of which were from only one of the three emotional categories.

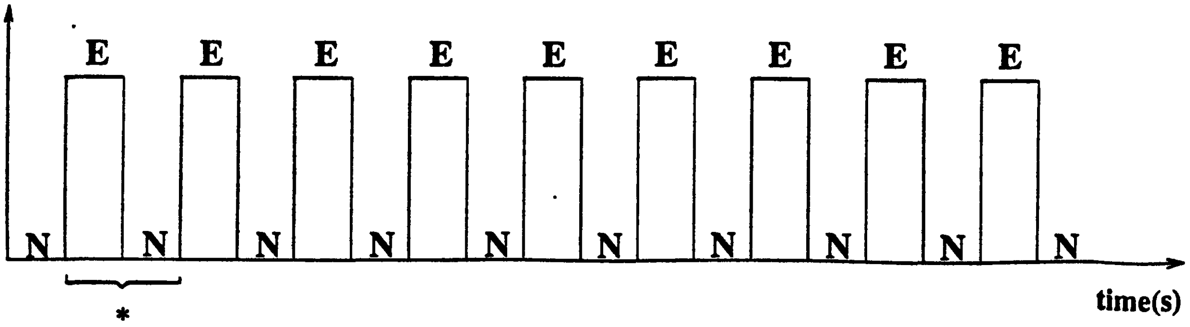


Fig 2-8. Alternating periods of emotionally active (*E*) and emotionally neutral (*N*) stimuli.

Fig 2-9 focuses upon one pair of the alternating periods shown in Fig 2-8 (denoted by an *). During the experiment, scanning of a single slice of the brain occurred every three seconds, as denoted by either the upward or downward impulses in Fig 2-9. For each experimental run, two slices of the brain were observed-- a posterior slice and an anterior slice. The posterior slice was scanned once every three seconds, and is denoted by the upward impulses (in Fig 2-9); the posterior slice was also scanned once every three seconds, and is denoted by the downward impulse. Scanning of the anterior slice occurred 1.5 seconds after the posterior slice. A picture was presented by the stimulus presentation program *immediately following* every other posterior scan, denoted by the impulses with circles in Fig 2-9. The darkened circles represent scans immediately followed by the presentation of an emotionally charged stimulus; the empty circles represent scans that are immediately followed by the presentation of neutral stimuli. The upper and lower dashed lines indicate periods during which emotionally charged and emotionally neutral stimuli were within the subject's view.

The script files for these experiments are contained in Appendix C.2. From these scripts, note that in each experiment, stimuli is presented immediately following every four scans (i.e., scan numbers 1, 5, 9, 13...). This is because the stimulus presentation program cannot distinguish between a posterior scan and an anterior scan. Thus, a posterior scan is performed first, immediately followed by a stimulus (note that the delays in each script are set to zero seconds). An anterior scan occurs 1.5 seconds later, followed by a second posterior scan, then a second anterior scan. The third posterior scan is followed by the presentation of a different stimulus, leaving the four subsequent scans (two posterior and two anterior) to occur while the subject is viewing the second picture. This process continues to repeat itself, alternating between neutral stimuli and emotional stimuli.

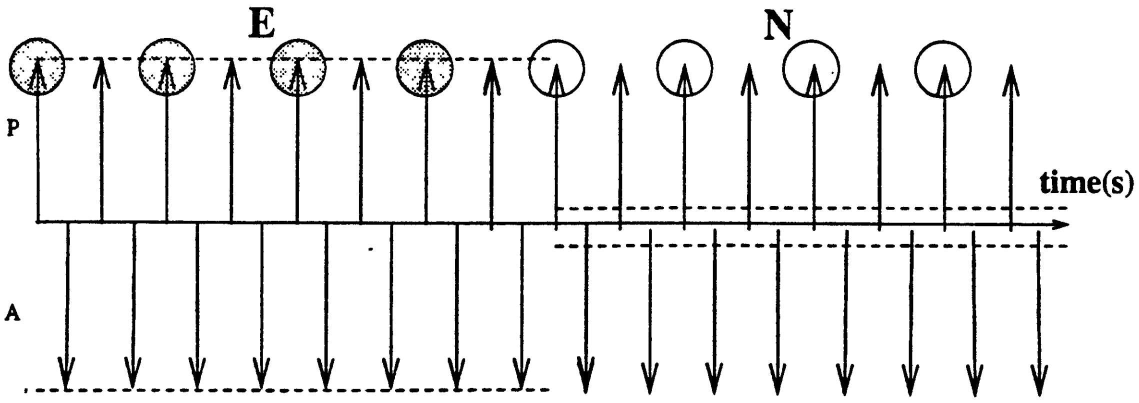


Fig 2-9. One period of presentation of emotionally neutral stimuli and emotionally active stimuli during the scanning of one slice of the brain. Impulses (arrows) indicate scans of one slice of the brain; circles indicate the presentation of stimuli immediately following the scan. Dark circles indicate emotional stimuli, while empty circles correspond to neutral stimuli.

Five experimental scripts were created (contained in Appendix C.2), each using one of the three sets of emotional stimuli along with the single set of neutral stimuli. Three of the scripts were named *FACE*, *NICE*, and *HORROR*. The stimuli in these scripts alternated between equal durations of emotional stimuli and neutral stimuli, as shown in Fig. 2-8. Each script followed the presentation of neutral stimuli with ten pairs of alternating periods of emotional and neutral stimuli, as shown in Fig. 2-9. The other two experimental scripts, *NICE-VARIANT* and *HORROR-VARIANT*, are variations of two of the aforementioned scripts. The primary difference in these scripts is that they contain longer periods of neutral stimuli, and the lengths of these periods vary in duration: instead of presenting four consecutive neutral pictures (as *NICE* and *HORROR* do), the variants may present six to twelve neutral pictures. Similar to *NICE* and *HORROR*, the variants present four consecutive emotional pictures in a period, but the number of emotional periods is cut (from [9]) to five (as shown in Fig 2-10). The scanning duration remains unchanged, as does the interval of time between scans (three seconds between same slice scans); but the ratio of the number of scans following emotional stimuli to the number of scans following neutral stimuli has shifted from roughly 1:1 to roughly 1:3 for the variants.

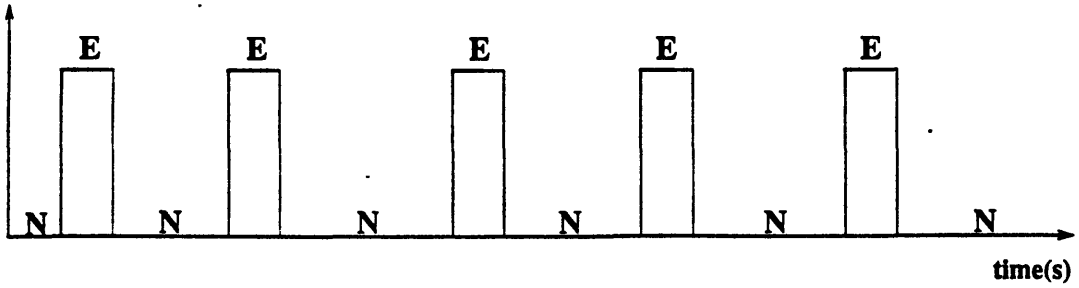


Fig 2-10. Schematic of the alternating periods of emotional and neutral stimuli for variant paradigms.

Experiments were performed on ten subjects, using either one, two, or three of the scripts. Data collection was done in part by Abigail Baird and Dr. Deborah Fein; scanner operation was performed by Anne Smith and Eileen Connolly, members of the McLean Hospital MRI staff. For subjects 2, 3, 4, 5, and 8, data was collected during exposure to the non-variant scripts; for subjects 6 and 7, data was collected using the variant scripts and the *FACE* script. Data for subjects number 2, 3, 4, and 5 were collected using a surface coil; data for subjects 6, 7, 8, 9, 10, and 11 were collected using a head coil, which allows the attainment of better signal-to-noise ratios. Each script and all pictures were tested in advance using the non-gating simulation program.

Instructions were given both before the subject entered the scanner and while the subject was inside. The subject was told before entering the scanner to observe the mirror just above the eyes (which is directed towards the projector screen). The subject was then requested to observe the screen without head movements as pictures were presented every few seconds. For the horrific and pleasant stimuli, the subject was told to watch and pay attention to the pictures, and to react as naturally as possible. For the face pictures (which were always presented as the last set), the subject was told to observe the emotional expression of the projected face and wiggle his or her foot when that expression was fear, with the anticipation that discrimination of emotional stimuli will produce activation.

Scout scans were used to select a posterior and anterior slice of the amygdala. Literature on structural images and their corresponding fMRI images were used to identify those slices that contained the amygdala, similar to those shown in section 1.3. The scout slices for each subject are contained in Appendix D.

After giving the subject instructions and focusing the projector, functional imaging followed. Scans for each slice were taken every three seconds; scans alternated every 1.5 seconds between anterior and posterior slices. Scan slices measured a 256x192 slice area, each slice being 7 mm apart. Scans were single gradient echo, TE=40 msec, TR=300 msec, 40 mm FOV with a flip angle of 75°.

2.5.2 Methods of Data Analysis

Data files were saved for each experimental run of each subject. Data was initially analyzed with the program Genslice on a Sun Sparc Station. For each experimental run, the program produced a 256x192 grayscale map for one of the two slices. The values observed on the grayscale map corresponded to the intensity of each voxel for the first scan. (Subsequent scans can also be viewed. Though the changes in the structural image are subtle, switching to different scans can provide more information.) The contrast and intensity of the image can be altered manually, to give greater definition between the structures.

Using the cursor, one of the pixels in the structural image was selected. For any pixel selected, the data stream corresponding to that pixel during the experiment was viewed. The user can save a single pixel or a block of averaged pixels for later statistical study. Using reference materials and the high-definition scout slices, the region corresponding to what was perceived to be the right amygdaloid complex was determined. From this region, the data streams for four pixels were saved per slice per paradigm. Other data points close to but outside of the perceived amygdaloid complex were also selected and analyzed, but are not included in this report.

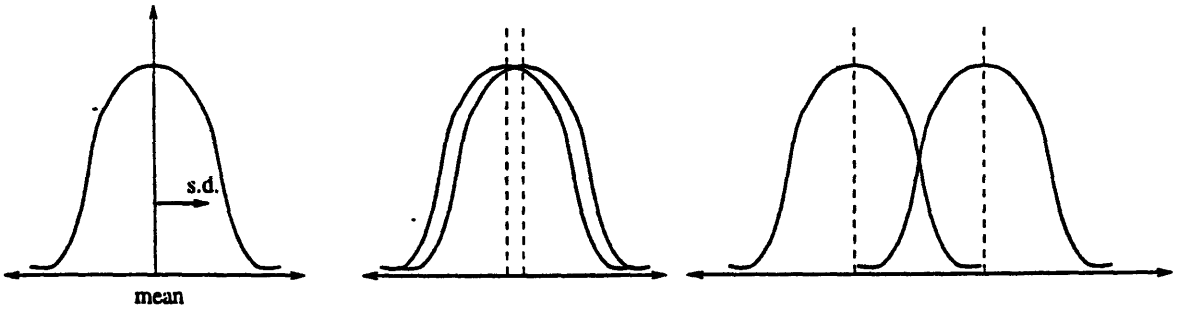
Saved files were incorporated into Microsoft Excel spreadsheets for final analysis. Data streams were then normalized with reference to baseline such that:

(1) Intensity, I = data point/(average of data points in data stream).

Thus, each data point represents ± its % above and below the average of the data stream.

Two-sample t-test for single data streams

For each normalized data stream, the data points were separated into two distinct groups: those that were sampled after the subject had been viewing an emotional stimulus, and those that occurred after the subject had been viewing a neutral stimulus. (Note that changes in stimuli occur immediately after a scan, so that with the exception of the first, scans occur after the subject has been viewing a stimulus for 1.5 or 4.5 seconds [for anterior scans] or for 3.0 or 6.0 seconds [for posterior scans].) For the non-variant data sets, scans occurring with and before the first two pictures and scans with or after the last two pictures were excluded from this analysis, effectively cutting the last neutral intervals from the previously specified eight scans to only four scans). For the variant data sets, the first five and last five data sets were not included. This was done for three reasons. State during initial scans may be influenced by the fact that a scanning sequence (which can be heard by the subject) had begun, or that stimuli had begun to appear; thus these stimuli are not truly neutral. For the nonvariants, it gives an equal number of points in each group ($n_{emotional} = n_{neutral}$) and balances the distribution of emotional and neutral scans over the time course of data collection. For variant data streams, the ratio of $n_{emotional}:n_{neutral}$ was 1:3.



(a) (b) (c)

Fig 2-11. (a) A sample data distribution. Examples of two data sets whose means were statistically the (b) same and (c) statistically different.

The two distinct groups in each data stream, one of data points following emotional stimuli and the other following neutral stimuli, constitute two distributions of data points. Thus for each distribution, n (the number of data points), $\bar{x}$ (the sample mean), and s (the sample standard deviation) were calculated, as shown in Fig 2-11(a) (See also Fig 2-12.) Further, for each sample pair, it was determined whether the two-sample distributions were statistically different (Fig 2-11(c) as opposed to 2-11(b)) using the two-sided t-test method [55]. (Note that a t-distribution, and thus a t-test, is used since it is not known whether the distributions are normal.) Thus for the any single datastream, the population means of the two distributions (emotional and neutral) were determined to be unequal ($\mu_{\text{emotional}} \neq \mu_{\text{neutral}}$, where μ is the the population mean) at a confidence interval of α when [55]

$$- t_{df,\,1\text{-}\alpha/2} > \frac{\bar{x}_e - \bar{x}_n}{\sqrt{({}^{S_n}/_{n_n})^2 + ({}^{S_e}/_{n_e})^2}} > t_{df,\,1\text{-}\alpha/2}$$

Degrees of freedom, *df*, were calculated using the following formula [55]:

$$df = \frac{S_n^2/_{n_n} + S_e^2/_{n_e}}{(S_n^2/_{n_n})^2/_{(n_n - 1)} + (S_e^2/_{n_e})^2/_{(n_e - 1)}}$$

where $t_{df,\,1\text{-}n/2}$ corresponds to a value in a t-distribution. For approximately equal variances, the number of degrees of freedom is equal to $n_1 + n_2 - 2$ [55]. This value

(142) served as a good estimate (± 4) of the degrees of freedom for the non-variant data sets. (Note that for values of the degrees of freedom this high, the t distribution very closely approximates the normal distribution.) For the variant data streams, equal variance was not assumed; almost all degrees of freedom values were above 60. Thus, the lower, conservative value of 60 was used for all variant two-sample t-tests. For each pair of data sets (emotional and neutral) within a data stream, it was determined whether or not the two samples were statistically different at confidence intervals of 95%. Confidence values were determined using a conservative curve-fit interpolation.

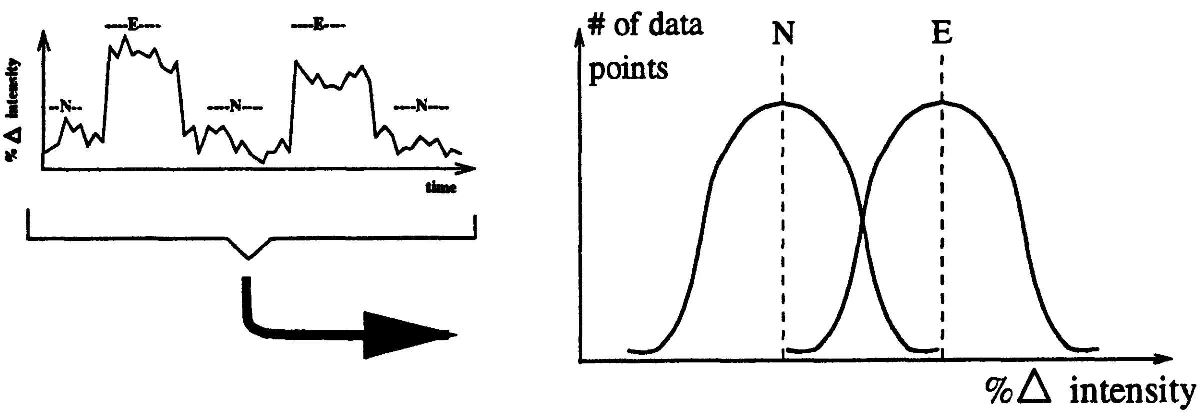


Fig 2-12. The two different stimulation periods in a data stream (those following observation of emotioal stimuli and those following observation of neutral stimuli) render two distributions that can be compared with a two-sample t-test.

Also of interest was whether or not there were significant delays in the creation of active periods. Thus, for each data stream, two more pairs of data sets were identified by shifting +1 and +2 scans to the right. In other words (for *shift* = +1), in an interval of consecutive emotional scans, the first scan was now incorporated with the neutral data set; similarly the (previously neutral) scan that followed the emotional scans was now incorporated into the emotional data set (as shown in Fig 2-13). The new pairs of data sets (referred to as *shift* = +1 and *shift* = +2) were also analyzed using two-sample t-tests in order to determine their level of confidence.

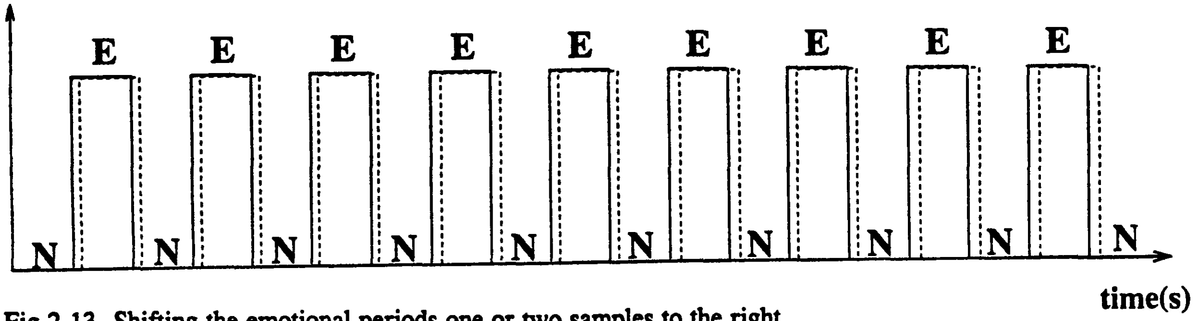


Fig 2-13. Shifting the emotional periods one or two samples to the right.

Paired t-test analysis comparing the difference between means of multiple data streams

A second analysis built upon the previous analysis. For each data stream, a pair of data were calculated, $\bar{x}_e$ and $\bar{x}_n$, the average intensity of the emotional and neutral periods, respectively. A paired t-test was used in order to determine if the mean difference between these two values (over multiple datastreams) was statistically different from zero. Note that the pairing of data points provided a more powerful test than if the data were unpaired [55].

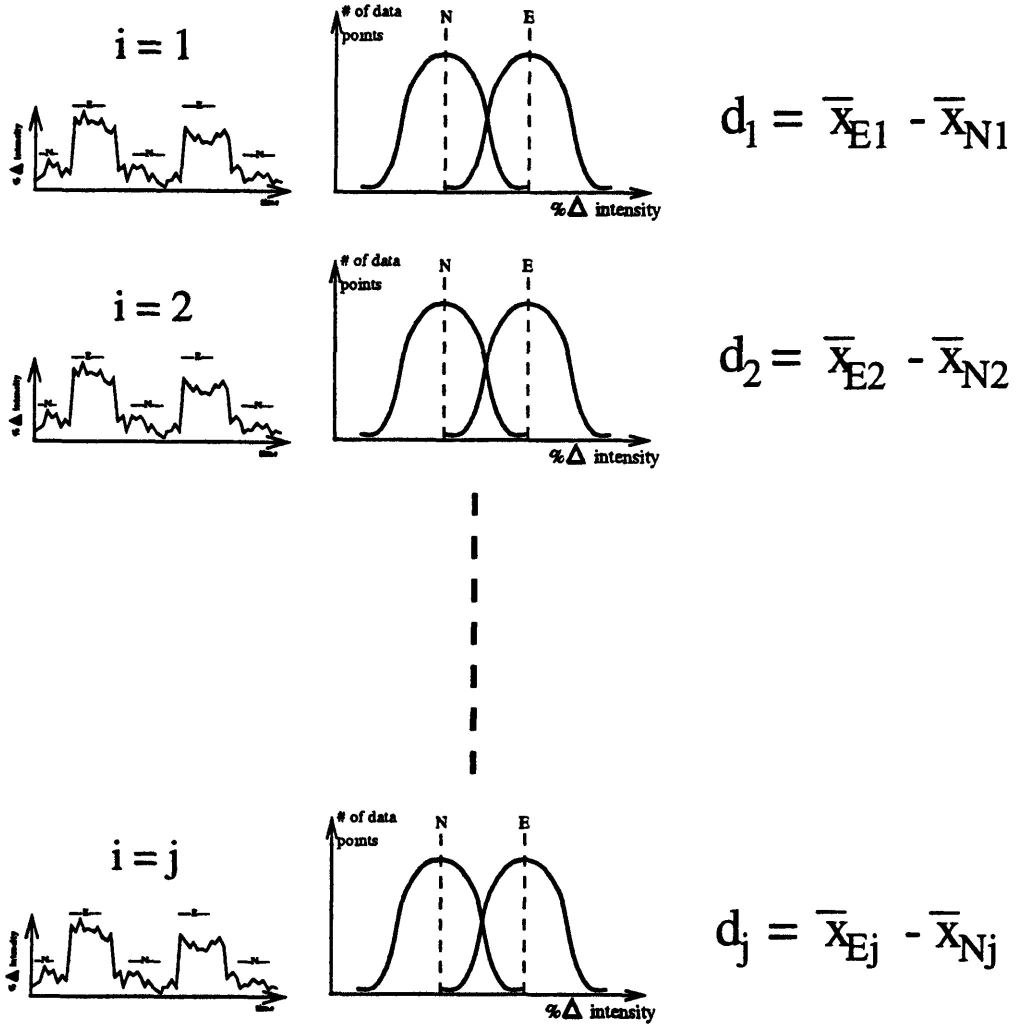


Fig 2-14. Paired t-test across data streams and subjects. For each data stream the average intensity of the emotional and neutral periods were calculated, forming a data pair. The difference between the two sample means was calculated for each data stream, forming a new data set containing *j* differences. A paired t-test was used in order to assess if the mean of the differences, *d*, was statistically different from zero.

From each data stream's pair of means, the value d_i was determined where

$$d_i = \bar{x}_{e,i} - \bar{x}_{n,i}$$

for the *i*th data stream. The d_i values were then grouped across stimulus sets (i.e., either *HORROR* only, or *NICE* only, or *FACE* only) within a single slice (posterior or anterior) to produce new distributions of data points. Each data point represented the difference between sample means of the two emotional periods of a single data stream. A paired t-test was performed in order to determine if the mean of this distribution was statistically different from zero [55]; this was true when

$$-t_{nd-1,\, 1-\alpha/2} > t' = \frac{\bar{d} - 0}{s_d/\sqrt{n_d}} > t_{nd-1,\, 1-\alpha/2}$$

where n_d (equivalent to j in figure 2-14) is the number of data points in the distribution d_i, while s_d and $\bar{d}$ were the standard deviation and mean of the distribution d_i (for i = 1 to n_d). Different t-tables were used since a different number of subjects participated in each stimulus set.

Averaging over data streams

Since the presentation of stimuli was identical for every subject when a particular paradigm was used, data streams were averaged across subjects for a given paradigm and a given slice. For example, the first data point from each (relevant) data stream was grouped to form a distribution from which a mean and standard deviation were determined. When done for each data point, a stream of averages (along with their respective standard deviations) was determined. Data streams were grouped across subjects for each paradigm and slice.

Paired t-test comparison of a first and second scan during stimulus viewing

As mentioned, for each picture a subject viewed, two scans were taken (after 3.0 and 6.0 seconds of viewing for posterior scans, and after 1.5 and 4.5 seconds of viewing for anterior scans). Data analysis was performed in order to compare these two time frames when emotional pictures were viewed. For each data stream, scans during an emotional period were divided into two distributions. The first distribution contained the first scans during the viewing of a picture; the second distribution contained second scans during the viewing of a picture. For each distribution, a mean was calculated, giving a new pair of means for each data stream. Paired t-tests were performed across subjects in order to determine whether there was a statistical difference between first scans and second scans during a picture. (Paired t-test analyses are described on page 44.)

Chapter 3

Results

3.1 Verification of System Performance

The programs written in section 2.3.1 to verify hardware performance accomplished the tasks they were designed to perform. The program to test the counter-timer board's ability to count scans (reviewed in appendix B.3) correctly counted a preset number of scans during a variety of imaging sequences. The program to test the presentation of visual stimuli (contained in appendix B.4) displayed pictures on the projector screen, which were able to be viewed by subjects. Aspects of both programs were incorporated into the two final programs used by the experimenters (see sections 2.3.2 and 2.4, and appendix B.1 and B.2).

Performance of both these programs (one to simulate the presentation of stimuli during an experiment, the other to gate off the scanner and present stimuli during an experimental run), as well as the performance of the overall (gated stimuli presentation) system design were demonstrated through experimental implementation with cognitive paradigms. Tasks that were designed to elicit neural activity in the amygdala are described in section 2.5.

The program to simulate the presentation of stimuli was useful as a tool in observing the sequence of stimuli presentation without the use of scanner equipment. The program correctly displayed each stimulus stipulated in the script and in the correct order. No deviations from the predetermined (scripted) time that each stimuli was to be presented were observed by the users when compared with a stopwatch. The program allowed the experimenter to note dissatisfactions with the stimuli presentation, allowing for corrections to be made before an experimental run.

The gated stimulus presentation program gated off the scanner, counted scans, and presented stimuli as prescribed by the script file. It allowed for the presentation of identical paradigms to multiple subjects. No deviations were observed from the simulated presentation: sequence and timing corresponded to scans and in the correct order of presentation. Mechanical and electrical delays in the presentation of stimuli were not measured; however, in both programs, delays in the presentation of stimuli were not detectable to the naked eye.

3.2 Experimental Data

Stimuli presentation gated off a scanning sequence was performed on ten subjects, as described in the experimental protocols of section 2.5.1. Four pixels per slice per paradigm and slice were selected from the amygdala as described in section 2.5.2.

Two-sample t-tests for a single data stream

For each collected data stream, data points were separated into two distributions--those following the viewing of emotional stimuli and those following neutral stimuli (as shown in Fig 2-12). Means and standard deviations were calculated for both distributions and compared using a two-sided, two-sample t-test. Results are shown in the tables below. Distributions were realigned by shifting the emotional and neutral periods by one and two samples (as shown in Fig 2-13); two-sided t-tests were also calculated for these new data sets and are listed in the tables below. No distinct trends were apparent.

Posterior	NICE	HORROR	FACE	NICE/VAR	HOR/VAR
#2 male	N/A	Xe > Xn 85% Xe > Xn 96% Xe = Xn Xe = Xn	N/A	N/A	N/A
#3 male	Xe = Xn Xe < Xn 66% Xe > Xn 82% Xe < Xn 54%	Xe = Xn Xe > Xn 65% Xe < Xn 69% Xe = Xn	Xe = Xn Xe = Xn Xe > Xn 92% Xe = Xn	N/A	N/A
#4 female	Xe = Xn Xe > Xn 75% Xe < Xn 62% Xe > Xn 53%	Xe > Xn 67% Xe = Xn Xe > Xn 71% Xe > Xn 97%	Xe = Xn Xe = Xn Xe = Xn Xe = Xn	N/A	N/A
#5 female	Xe > Xn 78% Xe = Xn Xe = Xn Xe = Xn	Xe < Xn 92% Xe = Xn Xe = Xn Xe < Xn 99%	Xe = Xn Xe = Xn Xe > Xn 70% Xe > Xn 66%	N/A	N/A
#6 female	N/A	N/A	Xe < Xn 59% Xe = Xn Xe > Xn 62% Xe = Xn	Xe < Xn 90% Xe = Xn Xe = Xn Xe = Xn	Xe = Xn Xe > Xn 75% Xe < Xn 75% Xe > Xn 91%
#7 female	N/A	N/A	Xe > Xn 92% Xe > Xn 86% Xe = Xn Xe = Xn	Xe = Xn Xe < Xn 71% Xe > Xn 76% Xe = Xn	Xe > Xn 96% Xe = Xn Xe > Xn 64% Xe > Xn 98%
#8 male	Xe > Xn 76% Xe = Xn Xe = Xn Xe > Xn 84%	Xe > Xn 78% Xe = Xn Xe = Xn Xe = Xn	N/A	N/A	N/A
#9 female	Xe = Xn Xe = Xn Xe < Xn 77% Xe = Xn	Xe = Xn Xe = Xn Xe = Xn Xe = Xn	N/A	N/A	N/A
#10 male	Xe < Xn 78% Xe = Xn Xe = Xn Xe < Xn 77%	Xe > Xn 74% Xe < Xn 80% Xe = Xn Xe > Xn 71%	N/A	N/A	N/A
#11 male	Xe = Xn Xe = Xn Xe = Xn Xe = Xn	Xe < Xn 74% Xe = Xn Xe = Xn Xe = Xn	N/A	N/A	N/A

Posterior slice, shift=0

Fig 3-1. Table containing results of two-sample t-tests where shift=0 for posterior slices. Columns correspond to one of the five paradigms; rows correspond to a particular subject. Blocks contain the results of four data streams. If $\bar{x}_{emotional}$ was calculated as greater or less than $\bar{x}_{neutral}$ at above a 50% confidence level, the inequality is stated along with the confidence value. If less than 50%, $\bar{x}_{emotional} = \bar{x}_{neutral}$ is stated.

<u>ANTERIOR</u>	NICE	HORROR	FACE	NICE/VAR	HOR/VAR
#2 male	N/A	Xe > Xn 93% Xe > Xn 68% Xe > Xn 97% Xe > Xn 84%	N/A	N/A	N/A
#3 male	Xe > Xn 81% Xe = Xn Xe < Xn 92% Xe = Xn	Xe = Xn Xe > Xn 70% Xe > Xn 81% Xe > Xn 63%	Xe < Xn 79% Xe > Xn 71% Xe = Xn Xe = Xn	N/A	N/A
#4 female	Xe > Xn 77% Xe > Xn 93% Xe = Xn Xe > Xn 55%	Xe < Xn 74% Xe > Xn 87% Xe = Xn Xe = Xn	Xe > Xn 91% Xe < Xn 79% Xe = Xn Xe = Xn	N/A	N/A
#5 female	Xe < Xn 77% Xe = Xn Xe < Xn 59% Xe = Xn	Xe > Xn 91% Xe = Xn Xe = Xn Xe < Xn 51%	Xe < Xn 76% Xe = Xn Xe < Xn 76% Xe = Xn	N/A	N/A
#6 female	N/A	N/A	Xe > Xn 87% Xe = Xn Xe = Xn Xe = Xn	Xe > Xn 86% Xe < Xn 94% Xe > Xn 58% Xe = Xn	Xe < Xn 90% Xe = Xn Xe > Xn 94% Xe > Xn 99%
#7 female	N/A	N/A	Xe = Xn Xe = Xn Xe = Xn Xe > Xn 92%	Xe > Xn 86% Xe = Xn Xe < Xn 70% Xe < Xn 49%	Xe < Xn 77% Xe = Xn Xe = Xn Xe = Xn
#8 male	Xe > Xn 51% Xe = Xn Xe > Xn 51% Xe = Xn	Xe = Xn Xe = Xn Xe = Xn Xe = Xn	N/A	N/A	N/A
#9 female	Xe = Xn Xe = Xn Xe = Xn Xe = Xn	Xe = Xn Xe < Xn 84% Xe = Xn Xe < Xn 56%	N/A	N/A	N/A
#10 male	Xe < Xn 79% Xe = Xn Xe = Xn Xe = Xn	Xe > Xn 55% Xe > Xn 99% Xe < Xn 69% Xe = Xn	N/A	N/A	N/A
#11 male	Xe = Xn Xe = Xn Xe > Xn 79% Xe = Xn	Xe < Xn 85% Xe < Xn 67% Xe = Xn Xe < Xn 68%	N/A	N/A	N/A
Anterior slice, shift=0					

Fig 3-2. Table containing results of two-sample t-tests where shift=0 for anterior slices. Columns correspond to one of the five paradigms; rows correspond to a particular subject. Blocks contain the results of four data streams. If $\bar{x}_{emotional}$ was calculated as greater or less than $\bar{x}_{neutral}$ at above a 50% confidence level, the inequality is stated along with the confidence value. If less than 50%, $\bar{x}_{emotional} = \bar{x}_{neutral}$ is stated.

POSTERIOR	NICE	HORROR	FACE	NICE/VAR	HOR/VAR
#2 male	N/A	Xe > Xn 74% Xe > Xn 98% Xe = Xn Xe = Xn	N/A	N/A	N/A
#3 male	Xe = Xn Xe < Xn 69% Xe > Xn 73% Xe < Xn 65%	Xe = Xn Xe > Xn 81% Xe < Xn 58% Xe = Xn	Xe = Xn Xe = Xn Xe > Xn 81% Xe = Xn	N/A	N/A
#4 female	Xe = Xn Xe > Xn 68% Xe < Xn 93% Xe > Xn 50%	Xe = Xn Xe = Xn Xe = Xn Xe > Xn 96%	Xe = Xn Xe = Xn Xe = Xn Xe = Xn	N/A	N/A
#5 female	Xe = Xn Xe > Xn 82% Xe = Xn Xe = Xn	Xe < Xn 50% Xe = Xn Xe = Xn Xe < Xn 97%	Xe = Xn Xe = Xn Xe > Xn 82% Xe > Xn 94%	N/A	N/A
#6 female	N/A	N/A	Xe = Xn Xe = Xn Xe > Xn 94% Xe = Xn	Xe < Xn 82% Xe = Xn Xe = Xn Xe < Xn 50%	Xe > Xn 53% Xe = Xn Xe < Xn 84% Xe > Xn 92%
#7 female	N/A	N/A	Xe > Xn 63% Xe > Xn 81% Xe = Xn Xe > Xn 59%	Xe = Xn Xe = Xn Xe = Xn Xe > Xn 65%	Xe > Xn 95% Xe = Xn Xe > Xn 56% Xe > Xn 94%
#8 male	Xe = Xn Xe = Xn Xe > Xn 51% Xe > Xn 66%	Xe > Xn 93% Xe > Xn 85% Xe = Xn Xe > Xn 87%	N/A	N/A	N/A
#9 female	Xe = Xn Xe > Xn 55% Xe < Xn 59% Xe > Xn 81%	Xe = Xn Xe > Xn 68% Xe = Xn Xe = Xn	N/A	N/A	N/A
#10 male	Xe < Xn 84% Xe < Xn 62% Xe < Xn 77% Xe < Xn 97%	Xe > Xn 84% Xe = Xn Xe = Xn Xe > Xn 88%	N/A	N/A	N/A
#11 male	Xe = Xn Xe < Xn 57% Xe < Xn 79% Xe = Xn	Xe = Xn Xe = Xn Xe > Xn 92% Xe > Xn 64%	N/A	N/A	N/A

Posterior slice, shift=1

Fig 3-3. Table containing results of two-sample t-tests where shift=1 for posterior slices. Columns correspond to one of the five paradigms; rows correspond to a particular subject. Blocks contain the results of four data streams. If $\overline{x}_{emotional}$ was calculated as greater or less than $\overline{x}_{neutral}$ at above a 50% confidence level, the inequality is stated along with the confidence value. If less than 50%, $\overline{x}_{emotional} = \overline{x}_{neutral}$ is stated.

ANTERIOR	NICE	HORROR	FACE	NICE/VAR	HOR/VAR
#2 male	N/A	Xe > Xn 79% Xe > Xn 74% Xe > Xn 98% Xe > Xn 65%	N/A	N/A	N/A
#3 male	Xe > Xn 70% Xe > Xn 88% Xe < Xn 69% Xe = Xn	Xe > Xn 74% Xe = Xn Xe > Xn 74% Xe = Xn	Xe < Xn 74% Xe > Xn 77% Xe = Xn Xe = Xn	N/A	N/A
#4 female	Xe > Xn 93% Xe > Xn 60% Xe > Xn 77% Xe > Xn 88%	Xe < Xn 61% Xe > Xn 94% Xe < Xn 66% Xe > Xn 72%	Xe > Xn 91% Xe = Xn Xe > Xn 60% Xe = Xn	N/A	N/A
#5 female	Xe < Xn 91% Xe = Xn Xe = Xn Xe = Xn	Xe > Xn 95% Xe > Xn 77% Xe > Xn 77% Xe > Xn 94%	Xe < Xn 77% Xe = Xn Xe < Xn 79% Xe < Xn 77%	N/A	N/A
#6 female	N/A	N/A	Xe > Xn 64% Xe < Xn 52% Xe = Xn Xe = Xn	Xe > Xn 97% Xe < Xn 97% Xe > Xn 78% Xe = Xn	Xe < Xn 95% Xe = Xn Xe > Xn 97% Xe > Xn 98%
#7 female	N/A	N/A	Xe = Xn Xe = Xn Xe > Xn 57% Xe > Xn 94%	Xe > Xn 92% Xe = Xn Xe = Xn Xe < Xn 86%	Xe < Xn 81% Xe = Xn Xe > Xn 50% Xe = Xn
#8 male	Xe = Xn Xe < Xn 52% Xe = Xn Xe = Xn	Xe = Xn Xe = Xn Xe < Xn 94% Xe > Xn 81%	N/A	N/A	N/A
#9 female	Xe > Xn 62% Xe < Xn 75% Xe = Xn Xe = Xn	Xe = Xn Xe = Xn Xe > Xn 57% Xe = Xn	N/A	N/A	N/A
#10 male	Xe < Xn 83% Xe < Xn 87% Xe = Xn Xe > Xn 91%	Xe = Xn Xe > Xn 99% Xe = Xn Xe < Xn 58%	N/A	N/A	N/A
#11 male	Xe = Xn Xe > Xn 60% Xe > Xn 56% Xe = Xn	Xe < Xn 80% Xe < Xn 64% Xe = Xn Xe < Xn 87%	N/A	N/A	N/A
Anterior slice, shift=1					

Fig 3-4. Table containing results of two-sample t-tests where shift=1 for anterior slices. Columns correspond to one of the five paradigms; rows correspond to a particular subject. Blocks contain the results of four data streams. If $\overline{x}_{emotional}$ was calculated as greater or less than $\overline{x}_{neutral}$ at above a 50% confidence level, the inequality is stated along with the confidence value. If less than 50%, $\overline{x}_{emotional} = \overline{x}_{neutral}$ is stated.

POSTERIOR	NICE	HORROR	FACE	NICE/VAR	HOR/VAR
#2 male	N/A	Xe = Xn Xe > Xn 92% Xe = Xn Xe = Xn	N/A	N/A	N/A
#3 male	Xe = Xn Xe < Xn 50% Xe > Xn 73% Xe = Xn	Xe < Xn 50% Xe > Xn 65% Xe < Xn 71% Xe = Xn	Xe = Xn Xe > Xn 74% Xe = Xn Xe > Xn 61%	N/A	N/A
#4 female	Xe > Xn 73% Xe > Xn 86% Xe < Xn 68% Xe > Xn 95%	Xe = Xn Xe = Xn Xe = Xn Xe > Xn 84%	Xe = Xn Xe = Xn Xe > Xn 57% Xe = Xn	N/A	N/A
#5 female	Xe = Xn Xe > Xn 91% Xe < Xn 54% Xe = Xn	Xe = Xn Xe = Xn Xe = Xn Xe < Xn 66%	Xe > Xn 83% Xe = Xn Xe = Xn Xe > Xn 75%	N/A	N/A
#6 female	N/A	N/A	Xe = Xn Xe < Xn 65% Xe > Xn 99% Xe > Xn 70%	Xe < Xn 96% Xe > Xn 69% Xe = Xn Xe < Xn 77%	Xe = Xn Xe > Xn 67% Xe < Xn 67% Xe > Xn 85%
#7 female	N/A	N/A	Xe > Xn 79% Xe > Xn 93% Xe = Xn Xe > Xn 96%	Xe < Xn 81% Xe = Xn Xe = Xn Xe = Xn	Xe > Xn 96% Xe = Xn Xe = Xn Xe > Xn 96%
#8 male	Xe = Xn Xe < Xn 78% Xe > Xn 91% Xe > Xn 76%	Xe > Xn 93% Xe = Xn Xe = Xn Xe > Xn 96%	N/A	N/A	N/A
#9 female	Xe > Xn 55% Xe > Xn 93% Xe < Xn 57% Xe > Xn 97%	Xe > Xn 80% Xe > Xn 80% Xe = Xn Xe = Xn	N/A	N/A	N/A
#10 male	Xe < Xn 94% Xe < Xn 55% Xe < Xn 83% Xe < Xn 98%	Xe > Xn 51% Xe = Xn Xe < Xn 86% Xe > Xn 83%	N/A	N/A	N/A
#11 male	Xe > Xn 57% Xe = Xn Xe < Xn 64% Xe = Xn	Xe = Xn Xe < Xn 50% Xe > Xn 85% Xe > Xn 77%	N/A	N/A	N/A
Posterior slice, shift=2					

Fig 3-5. Table containing results of two-sample t-tests where shift=2 for posterior slices. Columns correspond to one of the five paradigms; rows correspond to a particular subject. Blocks contain the results of four data streams. If $\bar{x}_{emotional}$ was calculated as greater or less than $\bar{x}_{neutral}$ at above a 50% confidence level, the inequality is stated along with the confidence value. If less than 50%, $\bar{x}_{emotional} = \bar{x}_{neutral}$ is stated.

ANTERIOR	NICE	HORROR	FACE	NICE/VAR	HOR/VAR
#2 male	N/A	Xe > Xn 78% Xe > Xn 84% Xe > Xn 98% Xe = Xn	N/A	N/A	N/A
#3 male	Xe > Xn 79% Xe > Xn 54% Xe = Xn Xe > Xn 54%	Xe > Xn 88% Xe = Xn Xe > Xn 80% Xe > Xn 52%	Xe < Xn 54% Xe > Xn 80% Xe = Xn Xe = Xn	N/A	N/A
#4 female	Xe > Xn 95% Xe > Xn 85% Xe = Xn Xe > Xn 84%	Xe < Xn 55% Xe > Xn 69% Xe < Xn 91% Xe > Xn 62%	Xe > Xn 72% Xe < Xn 65% Xe = Xn Xe > Xn 54%	N/A	N/A
#5 female	Xe < Xn 74% Xe = Xn Xe = Xn Xe = Xn	Xe > Xn 96% Xe > Xn 82% Xe = Xn Xe > Xn 77%	Xe < Xn 93% Xe = Xn Xe < Xn 88% Xe < Xn 83%	N/A	N/A
#6 female	N/A	N/A	Xe = Xn Xe = Xn Xe = Xn Xe = Xn	Xe > Xn 96% Xe < Xn 79% Xe > Xn 69% Xe = Xn	Xe < Xn 73% Xe = Xn Xe > Xn 96% Xe > Xn 98%
#7 female	N/A	N/A	Xe = Xn Xe = Xn Xe = Xn Xe > Xn 91%	Xe > Xn 86% Xe = Xn Xe = Xn Xe < Xn 93%	Xe < Xn 95% Xe > Xn 55% Xe = Xn Xe = Xn
#8 male	Xe = Xn Xe = Xn Xe = Xn Xe = Xn	Xe < Xn 78% Xe = Xn Xe < Xn 84% Xe > Xn 84%	N/A	N/A	N/A
#9 female	Xe < Xn 56% Xe < Xn 80% Xe < Xn 66% Xe = Xn	Xe = Xn Xe = Xn Xe = Xn Xe < Xn 70%	N/A	N/A	N/A
#10 male	Xe < Xn 52% Xe < Xn 91% Xe > Xn 94% Xe > Xn 96%	Xe = Xn Xe > Xn 99% Xe = Xn Xe = Xn	N/A	N/A	N/A
#11 male	Xe = Xn Xe > Xn 62% Xe > Xn 51% Xe = Xn	Xe = Xn Xe < Xn 52% Xe > Xn 68% Xe = Xn	N/A	N/A	N/A

Anterior slice, shift=2

Fig 3-6. Table containing results of two-sample t-tests where shift=2 for anterior slices. Columns correspond to one of the five paradigms; rows correspond to a particular subject. Blocks contain the results of four data streams. If $\bar{x}_{emotional}$ was calculated as greater or less than $\bar{x}_{neutral}$ at above a 50% confidence level, the inequality is stated along with the confidence value. If less than 50%, $\bar{x}_{emotional}=\bar{x}_{neutral}$ is stated.

Paired t-test analysis comparing the difference between means of multiple data streams

Paired t-tests were performed across subjects and data streams using the mean intensity of a data stream's emotional periods and the mean intensity of the data stream's neutral periods as a data pair (see Fig 2-14). For a particular slice, stimulus set, and shift, a *mean* value of $\bar{x}_{emotional}$ - $\bar{x}_{neutral}$, the difference between individual pairs, was determined. A paired t-test was performed in order to determine if the mean value of the differences was statistically different from zero. Fig 3-7 shows the results of t-tests for different slices, stimulus sets, and shift.

	Anterior			**Posterior**			
	mean($\bar{x}_e$-$\bar{x}_n$)	t -value	confidence	mean($\bar{x}_e$-$\bar{x}_n$)	t -value	confidence	
NICE	+ 0.00062	+ 0.7969	58.2%	- 0.0004	- 0.6787	<50%	shift = 0
n_d = 36	+ 0.00087	+ 0.9156	65.6%	- 0.0008	- 1.1601	77.3%	shift = 1
	+ 0.00116	+ 1.3006	79.90%	- 0.0004	- 0.4749	<50%	shift = 2
	mean($\bar{x}_e$-$\bar{x}_n$)	t -value	confidence	mean($\bar{x}_e$-$\bar{x}_n$)	t -value	confidence	
HORROR	+ 0.00164	+ 1.70464	90.0%	+ 0.00215	+ 2.65597	98.4%	shift = 0
n_d = 40	+ 0.00181	+ 1.70602	90.0%	+ 0.00257	+ 3.27518	99.7%	shift = 1
	+ 0.00194	+ 2.08096	96.7%	+ 0.00215	+ 2.77086	98.6%	shift = 2
	mean($\bar{x}_e$-$\bar{x}_n$)	t -value	confidence	mean($\bar{x}_e$-$\bar{x}_n$)	t -value	confidence	
FACE	+ 0.00026	+ 0.27352	<50%	+ 0.00147	+ 2.25544	96.0%	shift = 0
n_d = 20	+ 0.00021	+ 0.21533	<50%	+ 0.00158	+ 2.23409	93.0%	shift = 1
	- 0.00009	- 0.09300	<50%	+ 0.00294	+ 3.24718	98.60%	shift = 2

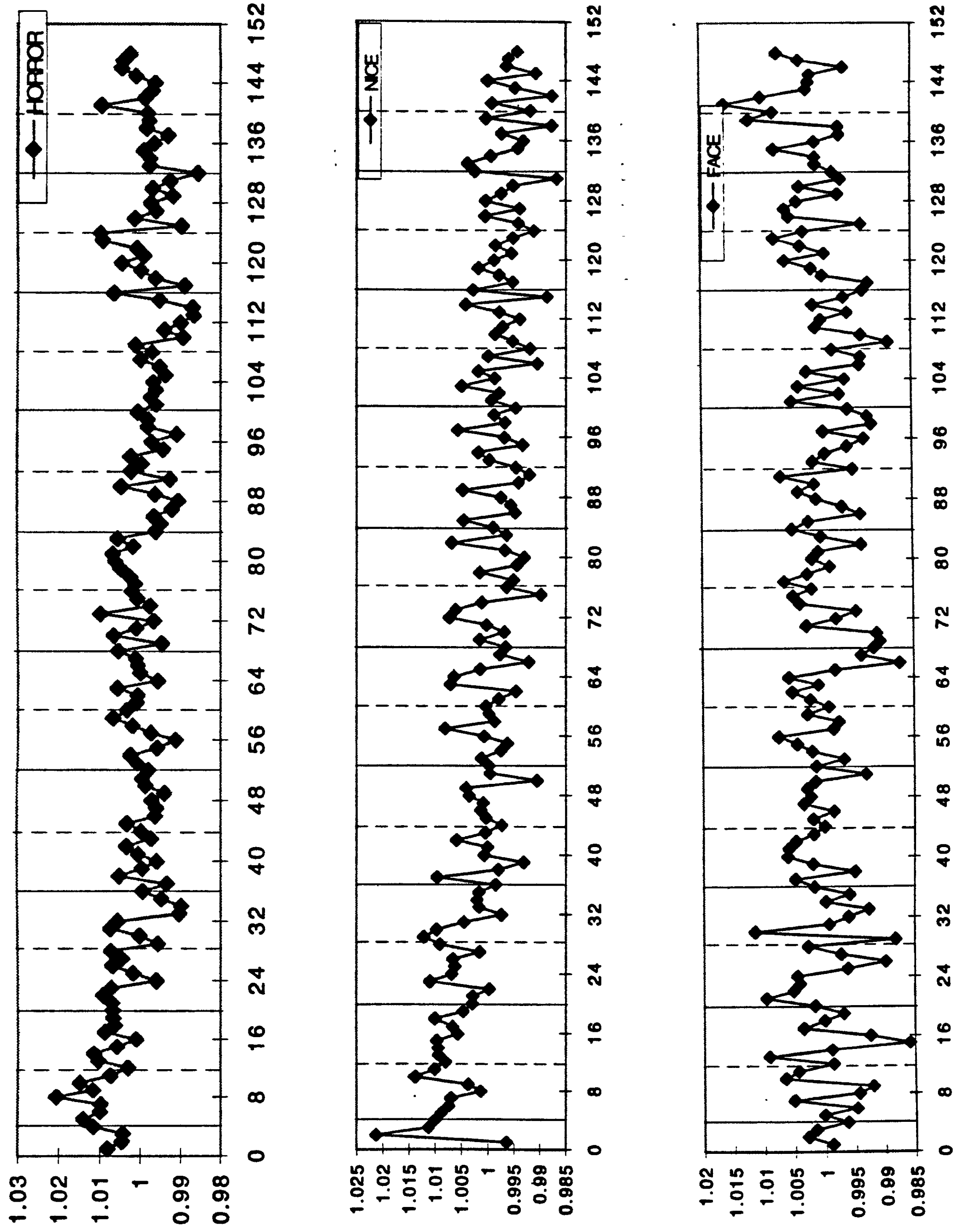
Fig 3-7. Results of t-tests for different stimulus sets, slices, and shifts. The number of data stream ($\bar{x}_e$ - $\bar{x}_n$) differences is denoted as n_d for each stimulus set. Mean ($\bar{x}_e$ - $\bar{x}_n$) denotes the average difference between a data stream's emotional and neutral periods. The corresponding t-value and two-sided confidence interval is listed alongside of it.

Note that none of the t-tests showed ($\bar{x}_{emotional} - \bar{x}_{neutral}$) being greater than zero. All stimuli/slice categories, with the exception of *NICE*/posterior and *FACE*/anterior, showed trends towards the mean ($\bar{x}_{emotional} - \bar{x}_{neutral}$) being less than zero. Both anterior and posterior *HORROR* as well as *FACE*/posterior showed the mean ($\bar{x}_{emotional} - \bar{x}_{neutral}$) being greater than zero at (two-sided) confidences of over 90%. Of these, *HORROR*/posterior was the largest, showing average differences of over 0.2% with respect to baseline. Most shifts peaked at shift=+1 or +2.

Averaging over data streams

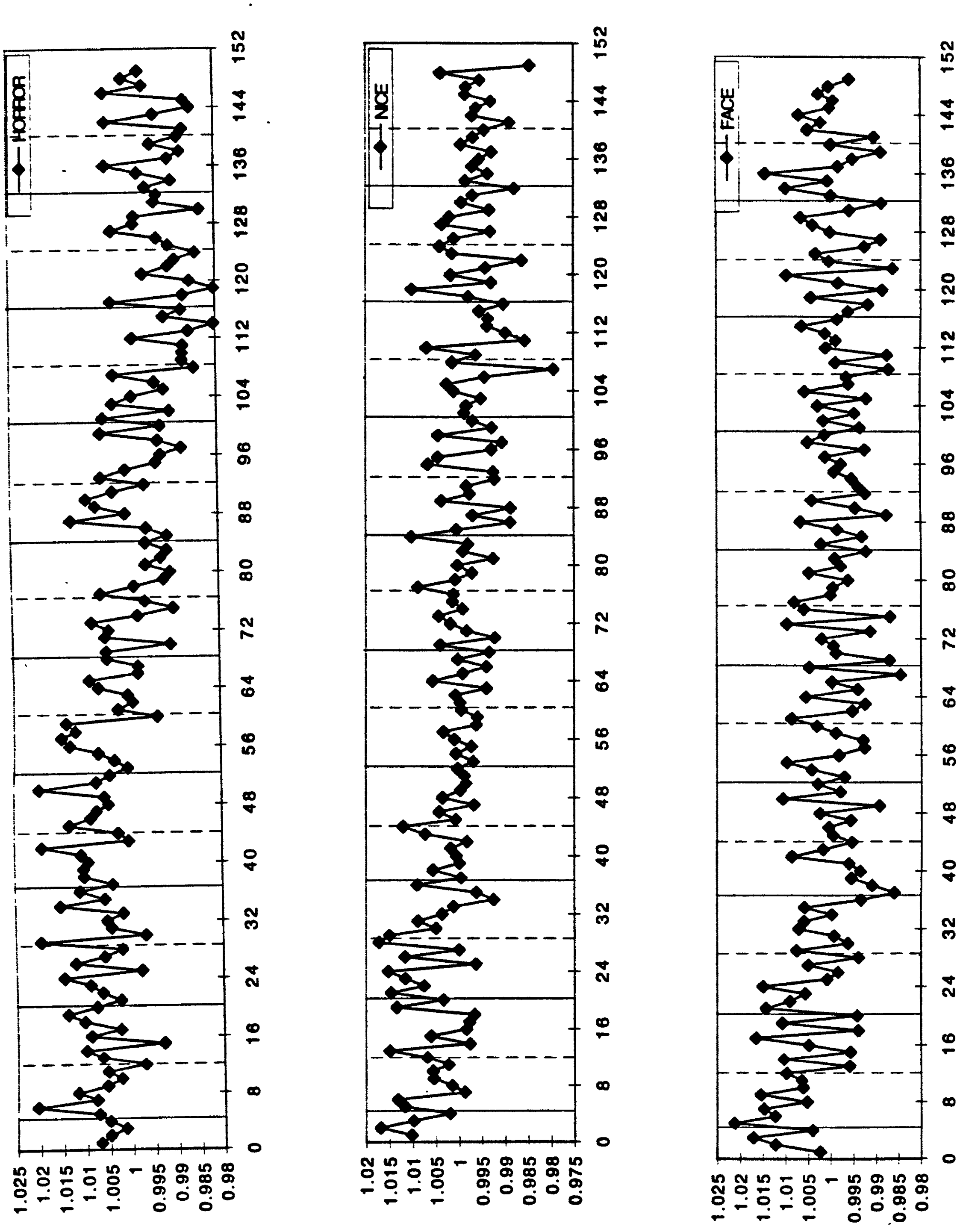
Non-variant data streams were averaged over subjects and data streams, calculating the average value and standard deviation of the n^{th} data point for a given paradigm and slice. Results are shown in Fig 3-8 and 3-9.

Though subtle, several trends were detectable in the progression of intensity during the course of a paradigm. Both anterior and posterior slices for the *HORROR* paradigm showed a considerable increase in signal intensity during the first interval of emotional stimuli. Other changes in intensity of this magnitude occurred at other points within these and other paradigms. The posterior slice of the *FACE* paradigm showed large changes in signal over the first two periods of emotional stimuli. With the exception of *FACE*/posterior, all streams appeared to have downward slopes. Finally, note that changes in signal intensity across the time course of the paradigms were generally smaller than the standard deviations of individual data points.



(a) (b) (c)

Fig 3-8. Non-variant, posterior slice of data stream averages for (a) *HORROR*, (b) *NICE*, and (c) *FACE* paradigms. The horizontal axes indicate posterior scan number (not including the first five scans); the vertical axes indicate changes in signal intensity with respect to baseline. Though not shown, the averages and ranges of standard deviations were: 0.031 and [0.018, 0.057] for *HORROR*; 0.031 and [0.210, 0.047] for *NICE;* and 0.027 AND [0.014, 0.043] for *FACE*. Solid vertical lines indicate initial viewing of emotional stimuli; dashed lines indicate initial viewing of neutral stimuli.



(a) (b) (c)

Fig 3-9. Non-variant, anterior slice of data stream averages for (a) *HORROR*, (b) *NICE*, and (c) *FACE* paradigms. The horizontal axes indicate posterior scan number (not including the first four scans); the vertical axes indicate changes in signal intensity with respect to baseline. Though not shown, the averages and ranges of standard deviations were: 0.031 and [0.017, 0.057] for *HORROR*; 0.027 and [0.014, 0.043] for *NICE;* and 0.031 AND [0.021, 0.047] for *FACE*. Solid vertical lines indicate initial viewing of emotional stimuli; dashed lines indicate initial viewing of neutral stimuli.

Paired t-test comparison of a first and second scan during stimulus viewing.

For each data stream, emotional scans were divided into two groups-- the first and second scan during the viewing of a picture from an emotional set. The averages of these two groups were also determined. Paired t-tests were performed on the differences between the two averages. The results are listed in the table below. With the exception of *HORROR*/anterior, the later (second) scans seemed to be higher than the firs scans, similar to shift trends found in paired t-tests comparing the difference between data stream means.

	Anterior			**Posterior**		
NICE $n_d = 36$	mean($\bar{x}_1$-$\bar{x}_2$)	t -value	confidence	mean($\bar{x}_1$-$\bar{x}_2$)	t -value	confidence
	- 0.0016	- 1.4576	84.8%	- 0.0014	- 1.4920	85.5%
HORROR $n_d = 40$	mean($\bar{x}_1$-$\bar{x}_2$)	t -value	confidence	mean($\bar{x}_1$-$\bar{x}_2$)	t -value	confidence
	+ 0.0023	+ 2.0952	95.6%	- 0.0004	- 0.3445	<50%
FACE $n_d = 20$	mean($\bar{x}_1$-$\bar{x}_2$)	t -value	confidence	mean($\bar{x}_1$-$\bar{x}_2$)	t -value	confidence
	- 0.0035	- 2.7721	98.5%	- 0.0021	- 1.6944	88.0%

Fig 3-10. The mean difference between the averages of the first and second scans when viewing an emotional stimulus are listed for different stimulus sets and slices. The results of paired t-tests were compared to the value zero.

Chapter 4

Discussion and Conclusions

Stimulus Presentation System: Possible Improvements

The stimulus presentation system, along with the programs to deliver stimuli and simulate the presentation of stimuli, both served as effective tools for preparing and running experiments that presented gated audio and visul stimuli during fMRI. The system was relatively easy for other users to learn and use. It allowed for the reuse of paradigms over multiple subjects and the coordination of the timing of stimulus presentation with collection of experimental data. The system appeared sufficiently fast when observing the timing of stimuli presentation. Finally, it effectively presented designed paradigms used in the studies of this report.

The system does have its limitations and inconveniences. The user cannot abort and restart the presentation program or make paradigm changes during an experiment. Only a small library of sound files are currently available. Also, a great deal of time must be devoted to scanning in pictures, cropping them, and converting them to PICT files. The program also has other quirks. For example, filenames must be limited to less than thirty characters, and stimuli and the script must be contained within a certain directory.

Though the time needed for the presentation of stimuli (i.e., system delays) was not detectable to the naked eye, they did exist. Though no measure of these delays were made, they can be discussed. The time from scan initiation to stimulus presentation can be broken into three parts. The first is scan recognition, performed by thje counter-timer hardware. This delay is relatively small (under 0.5 msec), according to the hardware specifications contained in appendix A.4. The second delay involves presentation decision. This is effectively the amount of time it takes the program to perform two lines of code containing "while" commands. The first line (or loop) accesses a register on the counter-timer board in order to see if the number of scans has changed since the previous register access. If the count has changed, the second line decides if a stimulus should be presented on this scan by comparing an integer in an array to the register value. The largest set of delays resulted from the presentation of stimuli. These delays varied, since it is likely to take different amounts of time to present different stimuli. However, as previously mentioned, observations of presentation timing did not pose any delays to the experimenters.

Several options are available for upgrading system performance. The first involves recompiling the programs in a Power Mac native version of Think C for the Macintosh. The purchase of a removable hard drive is also recommended, allowing the faster transportation of stimulus files in and out of the Macintosh used to deliver stimuli. Other upgrades could be made to boost system performance, such as more RAM or perhaps a faster video card. However, at the present time, the system appears to be sufficiently fast. Finally, concatenating counter-timers does not appear necessary, since experiments using over 64,000 changes in stimuli are unlikely.

Other experiments using the stimulus presentation system or its components are either planned or currently underway. One project currently underway uses a non-ferrous air pressure joystick (connected to one of the two 8-bit digital ports of the counter-timer board) along with the video projection system to observe ethanol-induced changes in motor activity.; A second experiment that is currently in the planning stages, anticipates adding another stimulus type (such as the system's current picture and sound capabilities) to the presentation program that would control an optical array through the second 8-bit digital port.

Amygdalar Activity

No strong trends were apparent when comparing the average intensities of emotional periods with those of neutral periods for single data streams. There are several possible explanations. One is that the signal to noise ratio in subcortical regions, as expected, is relatively weak. Other factors include errors and randomness in pixel selection. It was not uncommon for adjacent pixels to show different values for $\bar{x}_{emotional}$ - $\bar{x}_{neutral}$. This could have been the result of motion artifact at the borders of structures, the selection of pixels lying outside of the amygdalar region, or a weakness in experimental protocols. To avoid this, head coils are recommended for future studies of this region. The uniform signal-to-noise ratio throughout the slice provided better maps for pixel selection. Signal-to-noise ratios also appeared higher using the head coil. Finally, had the head coil been used in all experiments, the study of both amygdaloid bodies would have been possible.

However some subtle trends were apparent. Some subjects did appear to be consistently active or inactive for a particular paradigm. In general, $\bar{x}_{emotional}$ was greater more often than $\bar{x}_{neutral}$. This was further demonstrated when the average $\bar{x}_{emotional}$ - $\bar{x}_{neutral}$ was calculated over subjects and analyzed with a paired t-test. The paired t-tests showed trends towards the mean of $\bar{x}_{emotional,\, i}$ being greater than the mean of $\bar{x}_{neutral,\, i}$ for four of the six slice/emotional sets (anterior or posterior; and either *NICE*, *HORROR*, or *FACE*). Both *HORROR* slices, as well as the posterior *FACE* slice, showed the mean of $\bar{x}_{emotional,\, i}$ - $\bar{x}_{neutral,\, i}$ being greater than zero at statistically significant levels, with changes in signal intensity at levels of ± 0.2% of baseline. The most pominent of these were data streams from the posterior *HORROR* slices, all showing increases in signal intensity during emotional periods greater than +0.20%, with shift= +1 being the highest at +2.5%. With the exception of the *NICE* stimuli (which had no confidence values above 80%), greater confidence intervals were found for posterior slices than anterior slices. This might have been because the posterior slices typically gave a slightly larger, more defined amygdalar region from which to choose pixels.

As expected, no t-tests revealed significant trends towards the mean of $\bar{x}_{emotional,i} - \bar{x}_{neutral,\,i}$ being less than zero. This indicated that, on average, the amygdala was not more active in processing the neutral stimuli (repeated pictures of bricks) than it was in processing the emotional sets of stimuli (happy faces, faces expressing an emotion, or horrific images). This is the best evidence of the reliability of the study. However, these changes in signal intensity are considerably small (roughly an order of magnitude) in comparison to cortical studies. Whether this reflects the role of the amygdala role in processing the experimental stimuli, the feasability of subcortical experiments, or artifact and error in pixel selection is unknown.

Both the paired t-test data and the comparison of first and second scans during an emotional viewing indicated that there may have been delays in changes to signal intensity after the initial viewing of emotional stimuli. Specifically, a higher difference between the intensity of emotional periods and neutral periods was observed when shifting the periods 3.0 to 6.0 seconds later. This may have been a result of processing the end of the viewing period, or delays in activation or inactivation of the amygdala.

Finally, although the averages of data streams did present some trends (most notably a large peak during the first four horror pictures, as well as a downward drift for several stimuli streams), there did not appear to be any dominant patterns between emotional and neutral periods. Though the most likely candidate would be a square wave, the average data streams did not indicate any particular curve with which to correlate.

Appendix A

The Macintosh, C Programming, and Hardware

A.1 The Macintosh Interface, Toolbox, and Resources

Since the first Macintosh was introduced in 1984, the Macintosh line of computers have changed the way personal computers are used as tools. The Macintosh computer employs a graphical user interface and relies heavily on a mouse for accomplishing tasks. The most recent user interface, System 7, builds on Macintosh's unique use of menu bars and icons with added sound and graphics capabilities.

To create this user interface, the Macintosh uses a collection of thousands of procedures and functions known as the Macintosh Toolbox. These routines, defined in the Macintosh read-only memory, allows programmers to manipulate the Macintosh environment. The routines in the Toolbox allow the user to do many higher-level tasks, including opening a window, creating and manipulating a menu bar, displaying a picture, playing a system sound, selecting a directory, choosing a file, and "cut and paste" routines. Because these functions are common to many applications, the Toolbox standardizes application operations and avoids redundant programming.

Aside from the interface and the Toolbox, the third aspect that makes programming on the Macintosh unique are resource files. Resource files can be created within the program ResEdit, which accompanies Think C. Resources provides an efficient way of using the standardized Toolbox commands to manipulate data that is unique to a program. For example, with ResEdit, the programmer can create a window in a resource file. The resource file contains all information about the window (e.g., type, size, placement, scroll bars, dialogue, and sub-windows). The program uses the Toolbox commands to open, manipulate, and close the window. Resource files can contain other information relevant to a program, such as custom menu bars, picture files (in the Toolbox "PICT" file standard), and buttons. The resource file is compiled with the rest of the code when creating a program.

A.2 Think C Programming

Think C is a C programming environment for the Macintosh. Programming in Think C requires the creation of a minimum of two files: the project file (for example, MRIgsp.π) and a source code file (for example, MRIgsp.c). Other files may also be used, such as a resource file (MRIgsp.π.rsrc) as well as other files containing source code (all ending with .c).

The project file is unique to Think C. Like many other Macintosh applications, it is not "written", but is interfaced graphically. Think C asks the user to create a project file by indicating what files will be used to make the program. For example, the user will indicate the source code file containing the main subroutine (MRIgsp.c); the user may also indicate other resource code files (for example files that accompany a particular add-on board); or the user may include library files that accompany Think C (such as ANSI or MacTraps, which give access to the Toolbox). Note that header files can be utilized using the C command "# include" within the code. The source code of course should be written in the C standard.

When the project file, resource files, and all source code files are complete, the files are linked and compiled. Think C provides that if the total size of the compiled code is too large, it can be segmented, separating the library and source code files into different segments. It can be tested using the Think C debugger, or used to create a stand-alone application.

A.3 The NB-TIO-10 Counter-Timer Input/Output board for the Macintosh.

The NB-TIO-10 is a timing I/O board for Macintosh NuBus computers. The board contains 10 counter-timer lines and a pair of 8-bit digital lines. The counter-timer can perform a wide range of pulse counting and wave generation experiments. Of particular interest are its event counting features and the counter-timer chips used to perform them.

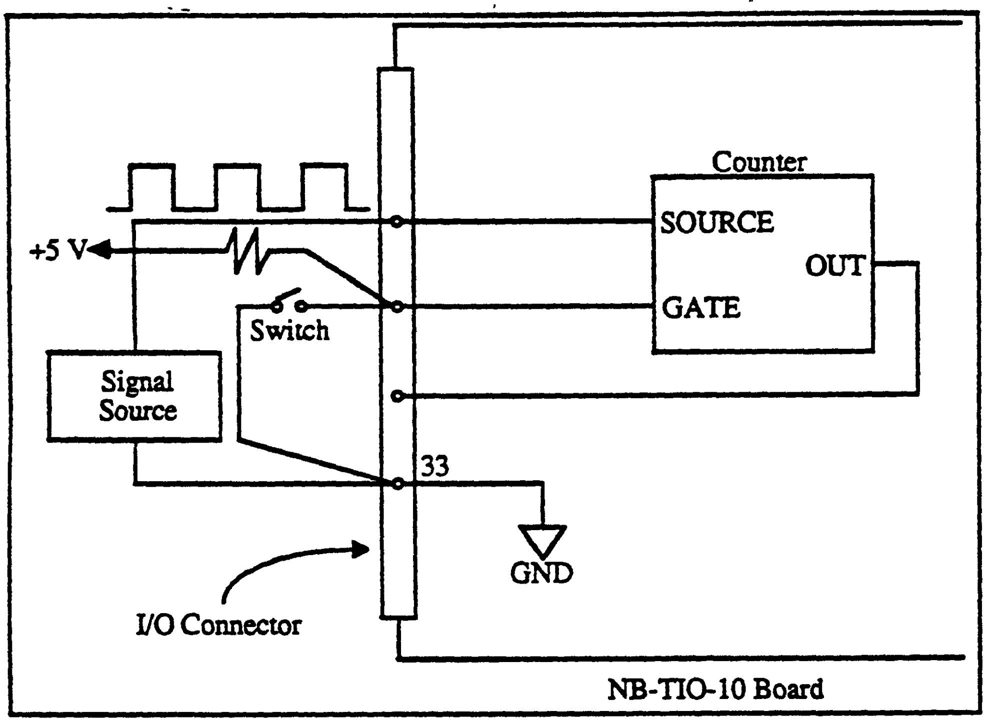


Fig A-1. Event-counting operation with internal switch gating (adapted from NB-TIO-10 user manual). Note that the switch can be controlled internally by the computer. Also note that the counter will increment a register for each event (pulse).

Above is a brief schematic of the counter-timer used in event counting operations. Fundamentally, the model is an edge-triggered latch. The gate is controlled by software in the computer. When it is opened, it begins event counting. Every time there is an edge from the source, the counter recognizes the edge and increment its 16-bit 65,536-count register by one. The register, through programming, can be accessed by the computer to obtain its value (or count). [56]

A summary of the board performance is shown below. Of particular note is t_{out}, the time it takes for the gate to recognize the edge (the change in voltage from the source) to the moment that it increments the edge. Other board specifications include voltage ratings: minimum and maximum input logic high voltage settings are 2.0 V and 5.25 V, respectively; minimum and maximum logic low voltage ratings are 0.0 V and 0.8 V, respectively. [56]

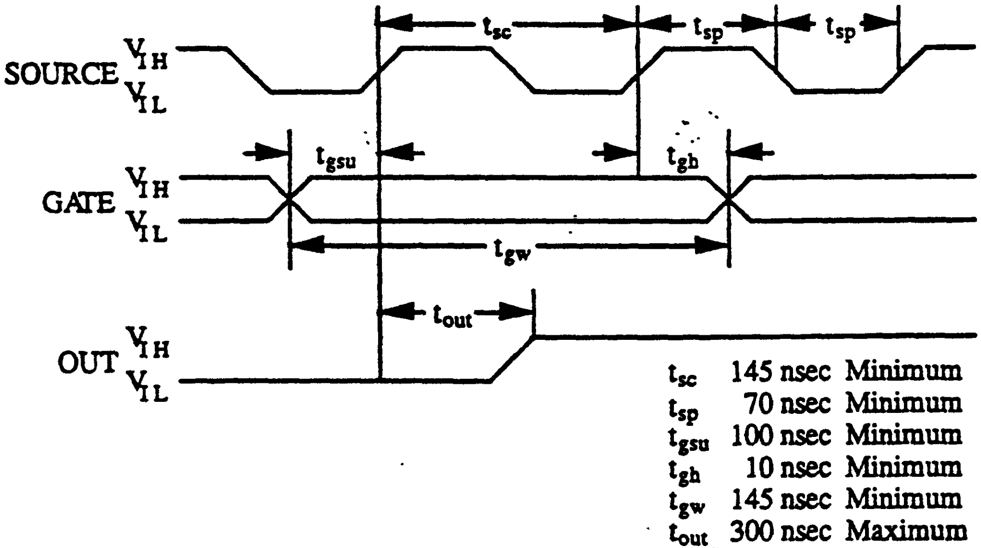


Fig A-2. Timing signal relationships (taken from the NB-TIO-10 manual).

Through the use of NI-DAQ software that accompanies the board, functions are available to operate the board through Think C fairly easily. These functions will not be delved into here, but suffice it to say that functions (or combinations of functions and parameters) are available for resetting the board, to configure the board for event counting, and accessing the counter for its register value (without interrupting board operation). [57]

A.4 Shielding and Television Monitor

The Macintosh monitor, like most computers and television monitors, emits electron beams in a vacuum tube that bombard luminescent phosphorous dots on the face of the picture tube. The path of charged particles is altered by magnetic fields. This creates a problem when using computer monitors around the strong magnetic fields of MRI machinery: the path of the electron beam can be skewed as it moves through the static field. A conventional method for isolating a region from magnetic fields is with the use of a conductor. Currents induced in a conductor tend to shield magnetic fields from a region when the conductor is thick compared to the effective skin depth of the surrounding fields [58]. This technique is employed by the MRI lab at McLean Hospital in order to isolate equipment from magnetic fields, and to assure clear viewing of monitors.

Appendix B

Source Code

B.1 Source Code for Stimulus Presentation Program

This section gives an overview of the gated stimulus presentation program, created with Think C version 6.0. The final program, when compiled, was used for the gated presentation of stimuli to subjects in the scanner during an experiment.

A resource file was created with ResEdit (which accompanies Think C). The resource file contains a plain box window, 640x480 pixels, with its right border matched up with the left border of the viewing screen. This is the location of the viewing area presented by the Radius video board, which feeds motion video to the Resonance Technologies projector.

The resources and library files (and their corresponding segmentation) used in creating the program are listed below.

Segment 2:
    ChkErr.c
    MacTraps
    MRIgsp.c
    NI_DAQ_MACL1

Segment 3:
    ANSI

The files were too large to be contained in one segment so ANSI (the largest file) was placed in a separate segment. ChkErr.c and NI_DAQ_MACL1 were used to allow the program to access the board and correctly gate off the scanner. MacTraps gives access to the Macintosh ToolBox, while ANSI was useful for displaying text and working with strings. With the exception of MRIgsp.c, all files were resources provided by Think C version 6.0. MRIgsp.c, the heart of the program, was written for the experiments in this thesis and is listed below:

```
#include <GestaltEqu.h>
#include <stdio.h>
#include <ctype.h>
#include <stdlib.h>
#include <pascal.h>
#include <Events.h>
#ifndef TC4
#include <Events.h>
#else
#include <EventMgr.h>
#endif
#include "NI_DAQ_MAC.h"

char DscriptName[30];
FILE *fp;

#define MAXLINE 100
#define MAXPRESET 25
#define MAXSCRIPT 1000
#define MAXCHAR 10000
#define kBaseResID     128
#define kMoveToFront   (WindowPtr)-1L
#define kEmptyString   "\p"
#define kNilFilterProc nil
#define kErrorAlertID  kBaseResID
#define kPICTHeaderSize 512
#define kEmptyTitle                  kEmptyString
#define kVisible               true
#define kNoGoAway                    false
#define kNilRefCon                   (long)nil
#define CTR1                         1          /* IDs counter #1*/
#define CONT                         0          /*  no rollover @65k; "overflow"
registered */
#define TIMEBASE1                    6          /* time base is counter 1 */
#define  DEV                         3          /*ID NB-TIO-10  */
#define OUT                          0          /* on reset, does not change counter
output */
#define EDGE                         0          /*   count RISING edge*/
#define OUTTYPE                      1          /* TC pulse output (could be 0) */
#define OUTPOLAR                     0          /* output pos logic  (could be 1) */
#define GATEMODE                     0          /* no gating -- computer controlled */
#define CONTrollover                 0          /* rollover @65k; counter 1 */

char *optionPtr[MAXSCRIPT];
```

```
int when[MAXSCRIPT];
int delay[MAXSCRIPT];
char type[MAXSCRIPT];
char *prePtr[MAXPRESET];


char header[] = "Maxwell:MRIpresents:visual stimuli:";

int templength;
int valuenow;
short     gOldMBarHeight;

void ToolBoxInit(void);
PicHandle  LoadPICTFile(char *myfilename);
void  WindowInit(void);
void  WindowInit2(void);
void  DrawMyPicture(PicHandle picture);
void  CenterPict(PicHandle picture, Rect *destRectPtr);



int my_kbhit()
{         EventRecord  keyEvent;
          return(EventAvail(keyDownMask, &keyEvent));
          }


/***********************ToolBoxInit()************************/

void  ToolBoxInit(void)
{
InitGraf( &thePort);
InitFonts();
InitWindows();
InitMenus();
TEInit();
InitDialogs(nil);
InitCursor();
}


/************************************************/

PicHandle  LoadPICTFile(char *myfilename)
{
short              srcFile;
PicHandle          picture;
char               pictHeader[ kPICTHeaderSize];
long               pictSize, headerSize;
long               feature;
OSErr     err;

            err = FSOpen(CtoPstr(myfilename),fsRdPerm,&srcFile);
            err = GetEOF(srcFile,&pictSize);
```

```
            headerSize = kPICTHeaderSize;
            err = FSRead(srcFile,&headerSize, pictHeader);
            pictSize -= kPICTHeaderSize;
            picture=(PicHandle)NewHandle(pictSize);
            HLock((Handle)picture);
            err = FSRead(srcFile,&pictSize, *picture);
            HUnlock((Handle)picture);
            FSClose(srcFile);
            return(picture);
}

/**********************DrawMyPicture***********************/

void  DrawMyPicture(PicHandle picture)
{
Rect    pictureRect;
WindowPtr window;

window = FrontWindow();
pictureRect = window->portRect;

CenterPict(picture, &pictureRect);
DrawPicture(picture, &pictureRect);
}

/***********************CenterPict**********************/

void  CenterPict(PicHandle picture, Rect *destRectPtr)
{
Rect   windRect, pictRect;

windRect = *destRectPtr;
pictRect = (**(picture)).picFrame;
OffsetRect(&pictRect, windRect.left - pictRect.left,
                                        windRect.top - pictRect.top);
OffsetRect(&pictRect, (windRect.right - pictRect.right)/2,
                                        (windRect.bottom - pictRect.bottom)/2);
*destRectPtr = pictRect;
}



/****************************************************/

char GetCH(void)
{
        short      theChar;
        theChar = getchar();
        fflush(stdin);
        return theChar;
}

/*****************************************************/
```

```
int intro(void)
{
        char c;

        printf("This program executes a scriptable paradigme that has already\n");
        printf("been created with the use of the an editor or the MRI stimuli\n");
        printf("presentation program.  Run this program only when you are ready\n");
        printf("to make a (trial) scan.  If you have not ceated a paradigme script\n");
        printf("or are not ready to run a scan, exit now and execute the MRI\n");
        printf("presents program in the MRI folder.\n");
        printf("\n");

        while(1) {
                printf("Do you wish to continue[y/n]->");
                c = GetCH();
                printf("\n");
                        switch (c) {
                         case 'y':
                         case 'Y':
                                 return 0;
                         case 'n':
                         case 'N':
                                 return 1;
                         default:
                                 break;  };
                         }
        printf("\n");
        printf("\n");
}

/*******************************************/

int continues(void)
{
        char c;
        printf("\n");
        printf("\n");
        printf("\n");
        printf("Very good.  At this point, the subject should enter the scanner and all\n");
        printf("preliminary scans should be made [orientation of the head, etc.].\n");
        printf("\n");
        printf("The scanner should be ready to begin [but not yet running] the
experimental\n");
        printf("scans.  Your final steps are:\n");
        printf("  1. Type b at the prompt.\n");
        printf("  2. Turn on the projector and focus [if necessary].\n");
        printf("  3. Begin scanning.\n");
        printf("\n");

        while(1) {
                printf("Type b if you are ready to begin these steps, or a to abort.->");
                c = GetCH();
                printf("\n");
                        switch (c) {
```

```
                        case 'b':
                        case 'B':
                                return 0;
                        case 'a':
                        case 'A':
                                return 1;
                        default:
                                break; };
                         }
        printf("\n");
        printf("\n");
}

/************************filename*********************/

int filename(void)
{
int i;
char key[30] = "quit";
extern char DscriptName[];
int p;

printf("The script you will be using should be placed in the folder\n");
printf("Maxwell:MRIpresents:paradimes, and the name of the script should not\n");
printf("20 characters.\n");
printf("Enter the name of the script you wish to use at the prompt, or if you\n");
printf("wish to exit, type \"quit\"\n");
printf("-->");
scanf("%s", DscriptName);
printf("you chose the file named:%s\n", DscriptName);

  for(i=0; i<30; ++i)
          {if (DscriptName[i] != key[i])
          {p=0; break;}
                  else {if (i == 29)
                    p=1;}
                 }
return p;
}

/**********************************************/

void board(void)
{
        /**** to be completed ****/

        /***initializes & readies board***/
;
}

/**********************************************/

int getScriptLength(void)
{
```

```
extern char DscriptName[];
int p = 0; int linetotal = 0;
char c;
while((c=getc(fp)) != EOF)
        {++p;
        if (c == ';')
                ++linetotal;}
templength = p + linetotal;
return linetotal;
}

/************************izit***********************/

int izit(char check[], char newone[])
{
int i = 0; int p = 0;
while(check[i] == newone[i])
        {i++;
        if (check[i] == '\0' && newone[i] == '\0')
                {p=1;break;}}
return p;
}

int izit2(char check[], char newone[], int cstart)
{
int p = 0; int i = 0;
for(;check[i] == newone[cstart];i++, cstart++)
{if (check[i] == '\0' && newone[cstart] == '\0')
        {p=1;break;}}
return p;
}


/***********************tillThat*************************/

int tillThat2(char linus[], char checker, int start)
{int z = start;
while(linus[start] != checker)
        ++start;
return start-z;}

/***********************stoi*************************/

int stoi(char s[])
{int i = 0; int n = 0;
for (; isdigit(s[i]); i++)
        {n = 10 * n + (s[i] - '0');}
return n;}

int stoi3(char s[], int begin, int lim)
{int n = 0;
for (; begin <= lim; begin++)
        {n = 10 * n + (s[begin] - '0');}
return n;}
```

```
/**********************prepscript**********************/

int prepScript(long linetotal)
{
extern char DscriptName[];
int presetnum = 0; int optionnum = 0;
int i; char c; int k; int b; int n; int l;
int whichway = 0;
char *tempPtr; char *temp2Ptr;
char Dbegin[10] = "BEGIN";
char temp[MAXCHAR]; int j = 0; int count = 0;
int z; int dogear;

for(i=0; i < linetotal; ++ i)
        {
        while((c = getc(fp)) != ';')
                {if (isspace(c))
                        ;
                else
                        {temp[j] = c; j++; count++;}
                }

        temp[j] = '\0';
        dogear = j - count;

        /*** scan# =delay(ms)# = single-letter-ID = file.name   ***/

        if (whichway == 0)
                {if (izit2(Dbegin, temp, dogear))
                        {whichway = 1; n = i;}
                else
                        {prePtr[presetnum] = &temp[dogear];
                         presetnum++;}
                }
        else
                {k = tillThat2(temp,'=', dogear);  /** count from beginning to first = **/
                 l = tillThat2(temp,'=', dogear + k + 1); l = l + 1 +k;  /**count from
beginning to second = **/
                 /**printf("dogear = %d, k = %d, l = %d\n", dogear, k, l);**/
                /***/
                when[optionnum] =  stoi3(temp, dogear, dogear + k - 1);
                /***/
                delay[optionnum] =  stoi3(temp, dogear + k +1, dogear + l - 1);
                /***/
                type[optionnum] = temp[dogear + l + 1];
                /***/
                optionPtr[optionnum] = &temp[dogear + l + 1 + 2];
                /***/
                printf("%d\t%d\t%d\t%c\t%s\n", optionnum, when[optionnum],
delay[optionnum], type[optionnum], optionPtr[optionnum]);
                optionnum++;
                }
```

```
        j++; count = 0;
        }

return (linetotal - n -1);
}

/**********************scanmessage*************************/

void scanmessage(void)
{
long p;

printf("\n");
printf("BEGIN SCANNING AFTER SCREEN TURNS BLANK");
printf("\n");
printf("\n");
for (p=0; p < 300000; ++p)
        ;
}

/****************************makeWindow***************************/

void makeWindow(void)
{
WindowPtr   window;
Rect              totalRect, mBarRect;
RgnHandle         mBarRgn;

gOldMBarHeight = MBarHeight;
MBarHeight = 0;
window = GetNewWindow(kBaseResID, nil, kMoveToFront);
SetRect( &mBarRect, screenBits.bounds.left,
                                  screenBits.bounds.top,
                                  screenBits.bounds.right,
                                  screenBits.bounds.top+gOldMBarHeight );
mBarRgn = NewRgn();
RectRgn(mBarRgn, &mBarRect);
UnionRgn(window->visRgn,mBarRgn, window->visRgn);
DisposeRgn(mBarRgn);
SetPort(window);
FillRect( &(window->portRect), black);
PenMode(patXor);
}

/***************************CleanWindow****************************/

void cleanwindow(void)
{
WindowPtr window;

window = FrontWindow();

SetPort(window);
FillRect( &(window->portRect), black);
```

```
PenMode(patXor);
}

/*****************************wait******************************/

void wait(int time)
{
int T1;

T1 = clock();
while((clock() - T1) < (60 * time))
                    {;}
}

/*****************************boardOn*****************************/

void boardOn(void)
{
int16 errorB;

errorB = CTR_Reset (DEV,CTR1,OUT);              /* RESETS COUNTER 1*/
chkerr("CTR_Reset",errorB);

errorB = CTR_Config (DEV, CTR1, EDGE, GATEMODE, OUTTYPE, OUTPOLAR);
chkerr("CTR_Config",errorB);      /* SETS COUNTER 1 CONFIGURATION */

errorB = CTR_EvCount (DEV, CTR1, TIMEBASE1, CONTrollover);
chkerr("CTR_EvCount",errorB);  /* STARTS COUNTING EVENTS */
}

/*****************************boardOff*****************************/

void boardOff(void)
{
int16 errorB;
errorB = CTR_Stop(DEV, CTR1);
chkerr("CTR_EvCount",errorB);
}

/*****************************showtime*****************************/


void showtime(void)
{
/**int PreValue = 0;**/ int CommandNumber = 0;
int c; int pickUP = 0;  int appNumber; int a;  int delaytime;
char *Dend = "end";
PicHandle picture;
char *picFileP;
char picFileA[255];
int exiterror= 0;
int i = 0; int j = 0;
int16 errorB;  int16 overflow;
long count; int COUNT;
```

```
makeWindow();

 errorB = CTR_EvRead(DEV, CTR1, &overflow, &count);
 chkerr("CTR_EvRead",errorB);
 COUNT = count;

while( exiterror == 0)
            {while(COUNT <  when[CommandNumber])
                        {errorB = CTR_EvRead(DEV, CTR1, &overflow, &count);
                         COUNT = count;}

             if (delay[CommandNumber] != 0)
                        {delaytime = delay[CommandNumber]/1000;
                         wait(delaytime);}
             switch(type[CommandNumber])
                        {case 'p': case'P':
                                    cleanwindow();
                                    if (pickUP == 1)
                                                {KillPicture(picture);}

                                                i=0; j=0;
                                                while (header[i] != '\0')
                                                            {picFileA[i] = header[i];
                                                             i++;}
                                                while ( *(optionPtr[CommandNumber] +j) != '\0')
                                                            {picFileA[i] = *(optionPtr[CommandNumber] +j);
                                                            i++; j++;}
                                                picFileA[i] = *(optionPtr[CommandNumber] +j);

                                    picFileP = &picFileA[0];

                                    picture = LoadPICTFile(picFileP);
                                    if( picture != nil)
                                                {DrawMyPicture(picture);}
                                    pickUP = 1;
                                    break;
                         case 'b': case 'B':
                                    SysBeep(8);
                                    break;
                         case 'c': case 'C':
                                    cleanwindow();
                                    break;
                         case 'e': case 'E':
                                    exiterror = 1;
                                    break;
                         default:
                                    /*goto error;*/
                                    SysBeep(8);SysBeep(8);
                                    printf("%c\n",c); /*DrawString("\pDatafile error:Type not
found!")*/;
                                    exiterror = 1;
                                    break;
                         }
```

```
            printf("Displaying stimuli number %d, @ scan # %d, %s\n", CommandNumber,
when[CommandNumber], optionPtr[CommandNumber]);
            CommandNumber++;
            if (exiterror == 1)
                        break;
            }

if(exiterror == 3)
            printf("keyboard hit -- program aborted");


MBarHeight = gOldMBarHeight;
}


/**********************main************************/

void main(void)
{
            int scriptLines; int NumExecutes;
            int exit;
            exit = 0;

            exit = intro();
            if (exit == 1)
                        {printf("hit <return> to exit"); return;}

            ToolBoxInit();

            exit = filename();
            if (exit == 1)
                        {printf("hit <return> to exit"); return;}
            if ((fp = fopen(DscriptName, "r")) == NULL)
                        {printf("cant open file\nhit <return> to exit"); return;}
            scriptLines = getScriptLength();
            if (scriptLines > MAXCHAR)
                        {printf("this file is too long!\nhit <return> to exit"); return;}
            fp = fopen(DscriptName, "r");
            NumExecutes = prepScript(scriptLines);

            exit = continues();
            if (exit == 1)
                        {printf("hit <return> to exit"); return;}

            boardOn();
            scanmessage();
            showtime();
            boardOff();

            MBarHeight = gOldMBarHeight;
            printf("done");

}
```

B.2 Source Code for Simulation of Presentation Program

This section gives an overview of the non-gated stimulus presentation simulation program, which was designed to present visual stimuli on the Macintosh computer monitor. This program, created with Think C version 6.0, was used to test paradigms and stimuli before experimental runs without the use of scanner machinery.

A resource file was created with ResEdit. The resource file contains a plain box window, 640x480 pixels, which completely fills the viewing area of the monitor. The resources and library files (and their corresponding segmentation) used in creating the program are listed below.

Segment 2:
- MacTraps
- MRItest.c

Segment 3:
- ANSI

The files were too large to be contained in one segment, so ANSI (the largest file) was placed in a separate segment. MacTraps gives access to the Macintosh ToolBox, while ANSI was useful for displaying text and working with strings. With the exception of MRItest.c, all files were resources provided by Think C version 6.0. MRItest.c, the heart of the program, was written for the experiments in this thesis and is listed below:

```
#include <GestaltEqu.h>
#include <stdio.h>
#include <ctype.h>
#include <stdlib.h>
#include <pascal.h>
#include <Events.h>

char DscriptName[30];
FILE *fp;
```

```
#define MAXLINE 100
#define MAXPRESET 25
#define MAXSCRIPT 1000
#define MAXCHAR 10000
#define kBaseResID	128
#define kMoveToFront	(WindowPtr)-1L
#define kEmptyString	"\p"
#define kNilFilterProc	nil
#define kErrorAlertID	kBaseResID
#define kPICTHeaderSize 512
#define kEmptyTitle				kEmptyString
#define kVisible			true
#define kNoGoAway				false
#define kNilRefCon				(long)nil

char *optionPtr[MAXSCRIPT];
int when[MAXSCRIPT];
int delay[MAXSCRIPT];
char type[MAXSCRIPT];
char *prePtr[MAXPRESET];

char header[] = "Maxwell:MRIpresents:visual stimuli:";

short intervalLength;
int templength;
int valuenow;
short	gOldMBarHeight;

void ToolBoxInit(void);
PicHandle LoadPICTFile(char *myfilename);
void WindowInit(void);
void WindowInit2(void);
void DrawMyPicture(PicHandle picture);
void CenterPict(PicHandle picture, Rect *destRectPtr);



int my_kbhit()
{	EventRecord keyEvent;
	return(EventAvail(keyDownMask, &keyEvent));
	}


/***********************ToolBoxInit()***********************/

void ToolBoxInit(void)
{
InitGraf( &thePort);
InitFonts();
InitWindows();
InitMenus();
TEInit();
InitDialogs(nil);
```

```
InitCursor();
}


/***********************************************/

PicHandle  LoadPICTFile(char *myfilename)
{
short              srcFile;
PicHandle          picture;
char               pictHeader[ kPICTHeaderSize];
long               pictSize, headerSize;
long               feature;
OSErr     err;

          err = FSOpen(CtoPstr(myfilename),fsRdPerm,&srcFile);
          err = GetEOF(srcFile,&pictSize);
          headerSize = kPICTHeaderSize;
          err = FSRead(srcFile,&headerSize, pictHeader);
          pictSize -= kPICTHeaderSize;
          picture=(PicHandle)NewHandle(pictSize);
          HLock((Handle)picture);
          err = FSRead(srcFile,&pictSize, *picture);
          HUnlock((Handle)picture);
          FSClose(srcFile);
          return(picture);
}

/**********************DrawMyPicture***********************/

void  DrawMyPicture(PicHandle picture)
{
Rect   pictureRect;
WindowPtr window;

window = FrontWindow();
pictureRect = window->portRect;

CenterPict(picture, &pictureRect);
DrawPicture(picture, &pictureRect);
}

/**********************CenterPict**********************/

void  CenterPict(PicHandle picture, Rect *destRectPtr)
{
Rect   windRect, pictRect;

windRect = *destRectPtr;
pictRect = (**(picture)).picFrame;
OffsetRect(&pictRect, windRect.left - pictRect.left,
                                        windRect.top - pictRect.top);
OffsetRect(&pictRect, (windRect.right - pictRect.right)/2,
                                        (windRect.bottom - pictRect.bottom)/2);
```

```
*destRectPtr = pictRect;
}



/*****************************************************/

char GetCH(void)
{
        short     theChar;
        theChar = getchar();
        fflush(stdin);
        return theChar;
}

/********************************************************/

int intro(void)
{
        char c;

        printf("This program executes a scriptable paradigme that has already\n");
        printf("been created with the use of the an editor or the Create-\n");
        printf("AParadigme program.THIS PROGRAM IS TO BE RUM ON THE MACINTOSH \n");
        printf("WITHOUT THE SCANNER RUNNING.  If you have not ceated a paradigme script\n");
        printf("or are not ready to run a scan, exit now and execute the MRI\n");
        printf("presents program in the MRI folder.\n");
        printf("\n");

        while(1) {
                printf("Do you wish to continue[y/n]->");
                c = GetCH();
                printf("\n");
                        switch (c) {
                         case 'y':
                         case 'Y':
                                return 0;
                         case 'n':
                         case 'N':
                                return 1;
                         default:
                                break;  };
                          }
        printf("\n");
        printf("\n");
}

/*******************************************/

int continues(void)
{
        char c; char howlongdigits[6];
        printf("\n");
        printf("\n");
```

```
        printf("\n");
        printf("Very good.\n");
        printf("\n");
        printf("IF the scanner were running, at what intervals [in miliseconds] \n");
        printf("would you want it to scan during this presentation?\n");
        printf("-->");
        scanf("%s", howlongdigits);
        intervalLength = stoi(howlongdigits);
        printf("\n");
        printf("you chose a script length of %d milliseconds. \n", intervalLength);
        printf("\n");

        while(1) {
                printf("Type b if you are ready to begin these steps, or a to abort.->");
                c = GetCH();
                printf("\n");
                        switch (c) {
                         case 'b':
                         case 'B':
                                return 0;
                         case 'a':
                         case 'A':
                                return 1;
                         default:
                                break;  };
                          }
        printf("\n");
        printf("\n");
}

/************************filename**********************/

int filename(void)
{
int i;
char key[30] = "quit";
extern char DscriptName[];
int p;

printf("The script you will be using should be placed in the folder\n");
printf("Maxwell:MRIpresents:paradimes, and the name of the script should not\n");
printf("20 characters.\n");
printf("Enter the name of the script you wish to use at the prompt, or if you\n");
printf("wish to exit, type \"quit\"\n");
printf("-->");
scanf("%s", DscriptName);
printf("you chose the file named:%s\n", DscriptName);

  for(i=0; i<30; ++i)
          {if (DscriptName[i] != key[i])
         {p=0; break;}
                 else {if (i == 29)
                   p=1;}
                 }
```

```
return p;
}

/*********************************************/

void board(void)
{
        /**** to be completed ****/

        /***initializes & readies board***/
;
}

/*********************************************/

int getScriptLength(void)
{
extern char DscriptName[];
int p = 0; int linetotal = 0;
char c;
while((c=getc(fp)) != EOF)
        {++p;
        if (c == ';')
                ++linetotal;}
templength = p + linetotal;
return linetotal;
}

/************************izit***********************/

int izit(char check[], char newone[])
{
int i = 0; int p = 0;
while(check[i] == newone[i])
        {i++;
        if (check[i] == '\0' && newone[i] == '\0')
                {p=1;break;}}
return p;
}

int izit2(char check[], char newone[], int cstart)
{
int p = 0; int i = 0;
for(;check[i] == newone[cstart];i++, cstart++)
{if (check[i] == '\0' && newone[cstart] == '\0')
        {p=1;break;}}
return p;
}


/***********************tillThat*************************/

/*int tillThat(char linus[], char checker)
{int zz=0;
```

```
while(linus[zz] != checker)
            ++zz;
return zz;}*/

int tillThat2(char linus[], char checker, int start)
{int z = start;
while(linus[start] != checker)
            ++start;
return start-z;}

/***********************stoi*************************/

int stoi(char s[])
{int i = 0; int n = 0;
for (; isdigit(s[i]); i++)
            {n = 10 * n + (s[i] - '0');}
return n;}

/*int stoi2(char s[], int lim)
{int i; int n = 0;
for (i=0; i <= lim; i++)
            {n = 10 * n + (s[i] - '0');}
return n;}*/

int stoi3(char s[], int begin, int lim)
{int n = 0;
for (; begin <= lim; begin++)
            {n = 10 * n + (s[begin] - '0');}
return n;}

/*************************************************/

void lookatfunction(char ss[])
{;
}


/*************************************************/

void DoIt(char ss[])
{;
}

/***********************prepscript**********************/

int prepScript(long linetotal)
{
extern char DscriptName[];
int presetnum = 0; int optionnum = 0;
int i; char c; int k; int b; int n; int l;
int whichway = 0;
char *tempPtr; char *temp2Ptr;
char Dbegin[10] = "BEGIN";
char temp[MAXCHAR]; int j = 0; int count = 0;
```

```
int z; int dogear;

for(i=0; i < linetotal; ++ i)
          {
          while((c = getc(fp)) != ';')
                    {if (isspace(c))
                              ;
                     else
                              {temp[j] = c; j++; count++;}
                     }

          temp[j] = '\0';
          dogear = j - count;

          /*** scan# =delay(ms)# = single-letter-ID = file.name   ***/

          if (whichway == 0)
                    {if (izit2(Dbegin, temp, dogear))
                              {whichway = 1; n = i;}
                     else
                              {prePtr[presetnum] = &temp[dogear];
                               presetnum++;}
                    }
          else
                    {k = tillThat2(temp,'=', dogear);  /** count from beginning to first = **/
                     l = tillThat2(temp,'=', dogear + k + 1); l = l + 1 +k;  /**count from beginning to second
= **/
                     /**printf("dogear = %d, k = %d, l = %d\n", dogear, k, l);**/
                    /***/
                    when[optionnum] =  stoi3(temp, dogear, dogear + k - 1);
                    /***/
                    delay[optionnum] =  stoi3(temp, dogear + k +1, dogear + l - 1);
                    /***/
                    type[optionnum] = temp[dogear + l + 1];
                    /***/
                    optionPtr[optionnum] = &temp[dogear + l + 1 + 2];
                    /***/
                    printf("%d\t%d\t%d\t%c\t%s\n", optionnum, when[optionnum], delay[optionnum],
type[optionnum], optionPtr[optionnum]);
                    optionnum++;
                    }

          j++; count = 0;
          }

return (linetotal - n -1);
}

/**********************scanready*************************/

void scanready(void)
{
long p;
```

```
printf("\n");
printf("BEGIN SCANNING AFTER SCREEN TURNS BLANK");
printf("\n");
printf("\n");
for (p=0; p < 300000; ++p)
        ;
}

/****************************makeWindow***************************/

void makeWindow(void)
{
WindowPtr   window;
Rect              totalRect, mBarRect;
RgnHandle         mBarRgn;

gOldMBarHeight = MBarHeight;
MBarHeight = 0;
window = NewWindow(nil, &(screenBits.bounds), kEmptyTitle,
                                        kVisible, plainDBox,kMoveToFront,
                                        kNoGoAway, kNilRefCon);
SetRect( &mBarRect, screenBits.bounds.left,
                                        screenBits.bounds.top,
                                        screenBits.bounds.right,
                                        screenBits.bounds.top+gOldMBarHeight );
mBarRgn = NewRgn();
RectRgn(mBarRgn, &mBarRect);
UnionRgn(window->visRgn,mBarRgn, window->visRgn);
DisposeRgn(mBarRgn);
SetPort(window);
FillRect( &(window->portRect), black);
PenMode(patXor);
}

/****************************CleanWindow****************************/

void cleanwindow(void)
{
WindowPtr window;

window = FrontWindow();

SetPort(window);
FillRect( &(window->portRect), black);
PenMode(patXor);
}

/******************************wait*********************************/

void wait(int time)
{
int T1;

T1 = clock();
```

```
while((clock() - T1) < (60 * time))
                    {;}
}

/*****************************showtime*****************************/


void showtime(void)
{
/**int PreValue = 0;**/ int CommandNumber = 0;
int To; int c; int pickUP = 0;  int appNumber; int a;  int delaytime;
char *Dend = "end";
PicHandle picture;
char *picFileP;
char picFileA[255];
int exiterror= 0;
int i = 0; int j = 0;

makeWindow();


To = clock();

while( exiterror == 0)
          {while((clock() - To) < (intervalLength *.06 * when[CommandNumber]))
                    {;}

          /*if(izit(Dend, optionPtr[CommandNumber]) == 1)
                    {exiterror = 2; break;}*/
          if (delay[CommandNumber] != 0)
                    {delaytime = delay[CommandNumber]/1000;
                     wait(delaytime);}
          switch(type[CommandNumber])
                    { case 'b': case 'B':
                              SysBeep(8);
                              break;
                    case 'c': case 'C':
                              cleanwindow();
                    case 'p': case'P':

                              cleanwindow();
                              if (pickUP == 1)
                                        {KillPicture(picture);}

                              /*picFile = comboString(header, optionPtr[CommandNumber]);*/

                                        i=0; j=0;
                                        while (header[i] != '\0')
                                                  {picFileA[i] = header[i];
                                                   i++;}
                                        while ( *(optionPtr[CommandNumber] +j) != '\0')
                                                  {picFileA[i] = *(optionPtr[CommandNumber] +j);
                                                  i++; j++;}
```

```
                picFileA[i] = *(optionPtr[CommandNumber] +j);

            picFileP = &picFileA[0];

            picture = LoadPICTFile(picFileP);
            if( picture != nil)
                {DrawMyPicture(picture);}
            pickUP = 1;
            break;
        case 'e': case 'E':
            exiterror = 1;
            break;
        default:
            /*goto error;*/
            SysBeep(8);SysBeep(8);
            printf("%c\n",c); /*DrawString("\pDatafile error:Type not found!")*/;
            exiterror = 1;
            break;
        }
    CommandNumber++;
    if (exiterror == 1)
            break;
    }

if(exiterror == 3)
    printf("keyboard hit -- program aborted");


MBarHeight = gOldMBarHeight;
}


/**********************main*************************/

void main(void)
{
    int scriptLines; int NumExecutes;
    int exit; /**/ int xxx;
    exit = 0;

    exit = intro();
    if (exit == 1)
        {printf("hit <return> to exit"); return;}

    ToolBoxInit();

    exit = filename();
    if (exit == 1)
        {printf("hit <return> to exit"); return;}
    if ((fp = fopen(DscriptName, "r")) == NULL)
        {printf("cant open file\nhit <return> to exit"); return;}
    scriptLines = getScriptLength();
    if (scriptLines > MAXCHAR)
        {printf("this file is too long!\nhit <return> to exit"); return;}
```

```
        fp = fopen(DscriptName, "r");
        NumExecutes = prepScript(scriptLines);

        /**for(xxx = 0; xxx <= 6; xxx++)
                {printf("%d\t%d\t%s\n", xxx, when[xxx], optionPtr[xxx]);}**/

        exit = continues();
        if (exit == 1)
                {printf("hit <return> to exit"); return;}

        board();
        scanready();
        showtime();


        MBarHeight = gOldMBarHeight;
        printf("done");

}
```

B.3 Source Code for Program to Test Counter-Timer Board

This section gives a brief overview of the short program used to test the counter-timer aspect of the project. The program sets the parameters of the counter-timer and counts 100 scans by the GE scanner. The resources and library files (and their corresponding segmentation) used in creating the program are listed below.

Segment 2:

- ChkErr.c
- MacTraps
- TestCT.c
- NI_DAQ_MACL1

Segment 3:

- ANSI

The files were too large to be contained in one segment so ANSI (the largest file) was placed in a separate segment. ChkErr.c and NI_DAQ_MACL1 were used to allow the program to access the board and correctly gate off the scanner. MacTraps gives access to the Macintosh ToolBox, while ANSI was useful for displaying text and working with strings. With the exception of TestCT.c, all files were resources provided by Think C version 6.0. TestCT.c, the heart of the program, is listed below (note that it also contains code for the option of concatenating the counters):

```
/*** simply uses counter 1 with timebase=source1
   Trigger connected to source1, pin#1        ***/

#include <stdio.h>
#ifndef TC4
#include <Events.h>
#else
#include <EventMgr.h>
#endif
#include "NI_DAQ_MAC.h"
```

```
#define CTR1                    1               /* IDs counter #1*/
#define CONT                    0               /*  no rollover @65k; "overflow" registered
*/
#define TIMEBASE1               6               /* time base is counter 1 */
#define  DEV                    3               /*ID NB-TIO-10  */
#define OUT                     0               /* on reset, does not change counter output
*/
#define EDGE                    0               /*   count RISING edge*/
#define OUTTYPE                 1               /* TC pulse output (could be 0) */
#define OUTPOLAR                0               /* output pos logic  (could be 1) */
#define GATEMODE                0               /* no gating -- computer controlled */
#define CONTrollover            0               /* rollover @65k; counter 1 */

/* key board hit test routine */
int my_kbhit()
{
        EventRecord     keyEvent;
        return (EventAvail(keyDownMask, &keyEvent));
}

int main()
  {
  int exit = 0;
  int16 errorB;   int16 overflow;
  long count;
  char blah;
  int lastC = 0; int value;



errorB = CTR_Reset (DEV,CTR1,OUT);                      /* RESETS COUNTER 1*/
chkerr("CTR_Reset",errorB);

errorB = CTR_Config (DEV, CTR1, EDGE, GATEMODE, OUTTYPE, OUTPOLAR);

        /* SETS COUNTER 1 CONFIGURATION */
chkerr("CTR_Config",errorB);

errorB = CTR_EvCount (DEV, CTR1, TIMEBASE1, CONTrollover); /* STARTS COUNTING EVENTS
*/
chkerr("CTR_EvCount",errorB);

printf("ready to count events\n ");


while(exit == 0)
 {
 errorB = CTR_EvRead(DEV, CTR1, &overflow, &count);
 chkerr("CTR_EvRead",errorB);                           /* READS OUNTER
W/OUT DISTURBING

          EVENT COUNTING INFO; PUTS VALUE
```

```
            INTO count */
  value = count;
  if (value != lastC)
           {printf("event # %d \n", value);
            lastC = value;}
  if (value == 100)
           exit = 1;
  }

  printf("\n done counting\n");
   scanf("%s",blah);

 errorB = CTR_Stop(DEV, CTR1);

  }


 /*
 If we concantonate the counters,

 Low order or sensitive counter has:
 #define CTR1                        1               IDs counter #1
 #define CONT1                       1               rollover @65k; count for #1 now = 0
 #define TIMEBASE1                   2               10 us resolution
 #define  DEV1                       3               ID NB-TIO-10
 #define OUT1                        1               on reset, sets counter output to 0
 #define EDGE1                       0               count RISING edge
 #define OUTTYPE1                    1               output TC pulse
 #define OUTPOLAR1                   0               output pos logic
 #define GATEMODE1                   0    no gating -- computer controlled


 The high order counter (which is incremented every 65K counts:
 #define CTR2                        2               IDs counter #1
 #define CONT2                       0               no rollover @65k; signals overflow
 #define TIMEBASE2                   0               counts low order counter output
 #define  DEV2                       3               ID NB-TIO-10
 #define OUT2                        1               on reset, sets counter output to 0
 #define EDGE2                       0               count RISING edge
 #define OUTTYPE2                    0               toggle output
 #define OUTPOLAR2                   0               output pos logic
 #define GATEMODE2                   0    no gating -- computer controlled


 also,
 errorB = CTR_Reset (DEV,3,OUT);
 errorB = CTR_Config (DEV, 3, EDGE, GATEMODE, OUTTYPE, OUTPOLAR);
 errorB = CTR_EvCount (DEV, 3, 5,1);
 */
```

B.4 Source Code for Program to Test Presentation of Visual Stimuli

This section gives an overview of a short program used to view a single picture through the Radius video board, which feeds motion video to the Resonance Technologies Projector for display on a screen in the scanning room.

A resource file was created with ResEdit. The resource file contains a plain box window, 640x480 pixels, with its right border matched up with the left border of the viewing screen. This is the location of the viewing area presented by the Radius video board that feeds motion video to the Resonance Technologies projector.

The resources and library files (and their corresponding segmentation) used in creating the program are listed below.

Segment 2

- MacTraps
- PictDisplay.c

Segment 3:

- ANSI

PictDisplay.c, the heart of the program, was created with Think C version 6.0 and is listed below:

```
#include <GestaltEqu.h>

#define kBaseResID      128
#define kMoveToFront    (WindowPtr)-1L
#define kEmptyString    "\p"
#define kNilFilterProc  nil
#define kErrorAlertID   kBaseResID
#define kPICTHeaderSize 512
#define kEmptyTitle                     kEmptyString
#define kVisible                true
#define kNoGoAway                       false
```

```
#define kNilRefCon                    (long)nil

short     gOldMBarHeight;

void ToolBoxInit(void);
void GetFileName(StandardFileReply *replyPtr);
PicHandle LoadPICTFile(StandardFileReply *replyPtr);
void WindowInit(void);
void WindowInit2(void);
void DrawMyPicture(PicHandle picture);
void CenterPict(PicHandle picture, Rect *destRectPtr);

/***********************main*********************/

void main(void)
{
PicHandle picture;
StandardFileReply reply;
ToolBoxInit();
GetFileName(&reply);

if (reply.sfGood)
 {
        picture = LoadPICTFile(&reply);
        if( picture != nil)
                { WindowInit2();
                 DrawMyPicture(picture);
                 while(! Button());
                 }
        }
                MBarHeight = gOldMBarHeight;
}

/***********************ToolBoxInit()***********************/

void ToolBoxInit(void)
{
InitGraf( &thePort);
InitFonts();
InitWindows();
InitMenus();
TEInit();
InitDialogs(nil);
InitCursor();
}

/***********************GetFileName***********************/

void GetFileName(StandardFileReply *replyPtr)
{
SFTypeList typeList;
short     numTypes;
long      feature;
/*OSErr    err;*/
```

```
/*
err = Gestalt(gestaltStandardFileAttr, &feature);
if (err!=noErr)
        SysBeep(10);
if (feature & (1 << gestaltStandardFile58))
        { */
         typeList[0] = 'PICT';
         numTypes = 1;
         StandardGetFile(kNilFilterProc,numTypes,typeList,replyPtr);
        /* }
else
        {SysBeep(10); SysBeep(10);}*/
}

/***********************************************/

PicHandle  LoadPICTFile(StandardFileReply *replyPtr)
{
short           srcFile;
PicHandle       picture;
char            pictHeader[ kPICTHeaderSize];
long            pictSize, headerSize;
long            feature;
OSErr     err;

/*err = Gestalt( gestaltFSAttr, &feature);*/

if ( feature & (1<< gestaltStandardFile58))
        {
         err = FSpOpenDF(&(replyPtr->sfFile),fsRdPerm,&srcFile);

         err = GetEOF(srcFile,&pictSize);

         headerSize = kPICTHeaderSize;

         err = FSRead(srcFile,&headerSize, pictHeader);

         pictSize -= kPICTHeaderSize;

         picture=(PicHandle)NewHandle(pictSize);

         HLock((Handle)picture);

         err = FSRead(srcFile,&pictSize, *picture);

         HUnlock((Handle)picture);
         FSClose(srcFile);
         return(picture);
         }
}

/*****************WindowInit*****************/
```

```
void WindowInit( void )
{
WindowPtr   window;

window = GetNewWindow(kBaseResID, nil, kMoveToFront);

ShowWindow(window);
SetPort(window);

}

/*****************WindowInit2*****************/

void WindowInit2( void )
{
WindowPtr   window;
Rect            totalRect, mBarRect;
RgnHandle       mBarRgn;

gOldMBarHeight = MBarHeight;   /**!!!**/
MBarHeight = 0;

window = NewWindow(nil, &(screenBits.bounds), kEmptyTitle,
                                        kVisible, plainDBox,kMoveToFront,
                                        kNoGoAway, kNilRefCon);

SetRect( &mBarRect, screenBits.bounds.left,
                                        screenBits.bounds.top,
                                        screenBits.bounds.right,
                                        screenBits.bounds.top+gOldMBarHeight );

mBarRgn = NewRgn();
RectRgn(mBarRgn, &mBarRect);
UnionRgn(window->visRgn,mBarRgn, window->visRgn);
DisposeRgn(mBarRgn);
SetPort(window);
FillRect( &(window->portRect), white);
PenMode(patXor);

}



/**********************DrawMyPicture**********************/

void DrawMyPicture(PicHandle picture)
{
Rect   pictureRect;
WindowPtr window;

window = FrontWindow();
pictureRect = window->portRect;

CenterPict(picture, &pictureRect);
```

```
DrawPicture(picture, &pictureRect);
}

/************************CenterPict**********************/

void  CenterPict(PicHandle picture, Rect *destRectPtr)
{
Rect   windRect, pictRect;

windRect = *destRectPtr;
pictRect = (**(picture)).picFrame;
OffsetRect(&pictRect, windRect.left - pictRect.left,
                                    windRect.top - pictRect.top);
OffsetRect(&pictRect, (windRect.right - pictRect.right)/2,
                                    (windRect.bottom - pictRect.bottom)/2);
*destRectPtr = pictRect;
}
```

Appendix C

Script Files

C.1 Sample Script File and Explanation

This section contains a sample script file for a dummy experiment and an explanation of what it does. The file is written with a simple text editor and saved as a text file:

```
this is a sample script file;
BEGIN;
4=0=p=face.pict;
9=500=p=bear.pict;
23=0=b=blank;
31=0=t=tone;
47=0=p=orange.pict;
47=0=t=tone;
50=0=e=end;
```

The file presents seven stimuli over a period in which 50 scans occur (note this could be 50 scans of one slice of the brain, or two slices with 25 scans each). The begin line contains the word "begin" in all caps, followed by a semicolon. No information is processed before the begin line; every line thereafter represents a stimulus or a change of conditions. Each line lists four conditions, with each condition separated by an equals sign, "=", and ending with a semicolon. The conditions are (in the following order): the scan number, after which a stimulus occurs; the delay in presenting the stimulus (in milliseconds) after the initiation of the scan; the type of stimulus; and the name of the stimulus (if relevant).

In the sample file above, the first stimulus is a picture file (denoted with the letter "p") "face.pict," and it occurs immediately after the fourth scan. The picture remains in the subject's view until 500 msec after the ninth scan, when the picture file "bear.pict" is presented. This picture remains until 0 ms after scan number 23, when the screen turns blank; the blank screen is denoted by the letter "b" (note that anything can be listed for type of stimulus). On the thirty-first scan, a tone is presented denoted by "b" (the tone is the one specified in the Macintosh control panel). After the 47th scan, two stimuli are presented: a tone followed immediately by the picture "couch.pict." Finally, after the fiftieth scan, the stimulus presentation ends, denoted by the letter "e."

Some features of the program are of special note. All spaces in the script file are ignored. The only stimuli that require stimuli names are picture files. For all other stimuli (tones[t], blank screens [b], and end of program[e]), the stimulus name is irrelevant. Finally, more than one stimulus can occur in a scan interval. Such stimuli are presented in the order that they are listed.

C.2 Experimental Script Files

Listed on the following pages are the scripts used for presenting the experimental paradigms during fMRI. For all files, each line represents a change in stimuli. An explanation of script files is contained in Appendix E. Note that with each experimental run, data is acquired on two coronal slices of the brain. Each posterior scan is followed by an anterior scan 1.5 seconds later; both posterior and anterior scans are taken at a rate of one scan every three seconds. The program counts both posterior and anterior scans.

Face

```
BEGIN;
1=0=p=BRICK1.pict;
5=0=p=rock2.pict;
9=0=p=neut1.pict;
13=0=p=rock14.pict;
17=0=p=mad1.pict;
21=0=p=sad1.pict;
25=0=p=happy1.pict;
29=0=p=fear1.pict;
33=0=p=rock10.pict;
37=0=p=brick2.pict;
41=0=p=rock13.pict;
45=0=p=NEUT3.pict;
49=0=p=fear8.pict;
53=0=p=mad2.pict;
57=0=p=fear2.pict;
61=0=p=happy2.pict;
65=0=p=rock2.pict;
69=0=p=BRICK8.pict;
73=0=p=brick6.pict;
77=0=p=BRICK2.pict;
81=0=p=happy3.pict;
85=0=p=fear3.pict;
89=0=p=sad3.pict;
93=0=p=mad3.pict;
97=0=p=NEUT1.pict;
101=0=p=rock11.pict;
105=0=p=BRICK1.pict;
109=0=p=NEUT6.pict;
113=0=p=sad4.pict;
117=0=p=happy4.pict;
121=0=p=mad4.pict;
125=0=p=fear4.pict;
129=0=p=BRICK8.pict;
133=0=p=rock13.pict;
137=0=p=brick4.pict;
141=0=p=rock12.pict;
145=0=p=happy8.pict;
149=0=p=mad5.pict;
153=0=p=sad5.pict;
157=0=p=sad2.pict;
161=0=p=rock10.pict;
165=0=p=brick5.pict;
169=0=p=BRICK1.pict;
173=0=p=BRICK7.pict;
177=0=p=fear6.pict;
181=0=p=sad6.pict;
185=0=p=mad6.pict;
189=0=p=fear3.pict;
193=0=p=rock2.pict;
197=0=p=BRICK2.pict;
201=0=p=neut3.pict;
205=0=p=BRICK6.pict;
209=0=p=happy7.pict;
213=0=p=happy6.pict;
217=0=p=sad7.pict;
221=0=p=mad7.pict;
225=0=p=rock11.pict;
229=0=p=BRICK5.pict;
233=0=p=rock14.Pict;
237=0=p=rock12.pict;
241=0=p=sad2.pict;
245=0=p=happy5.pict;
249=0=p=fear8.pict;
253=0=p=mad9.pict;
257=0=p=NEUT6.pict;
261=0=p=BRICK4.pict;
265=0=p=rock14.pict;
269=0=p=neut1.pict;
273=0=p=mad8.pict;
277=0=p=happy9.pict;
281=0=p=mad10.pict;
285=0=p=fear9.pict;
289=0=p=NEUT3.pict;
293=0=p=rock13.pict;
297=0=p=BRICK7.pict;
301=0=p=rock12.pict;
305=0=p=happy10.pict;
309=0=p=mad12.pict;
313=0=p=fear10.pict;
317=0=p=sad1.pict;
321=0=p=BRICK4.pict;
325=0=p=rock10.pict;
329=0=p=rock11.pict;
333=0=p=brick5.pict;
337=0=b=blank;
337=O=E=END;
```

Nice

```
BEGIN;
1=0=p=BRICK1.pict;
5=0=p=rock2.pict;
9=0=p=neut1.pict;
13=0=p=rock14.pict;
17=0=p=nice59.pict;
21=0=p=nice21.pict;
25=0=p=nice22.pict;
29=0=p=nice108.pict;
33=0=p=rock11.pict;
37=0=p=BRICK8.pict;
41=0=p=neut3.pict;
45=0=p=BRICK2.pict;
49=0=p=nice3.pict;
53=0=p=nice23.pict;
57=0=p=nice101.pict;
61=0=p=nice4.pict;
65=0=p=NEUT1.pict;
69=0=p=rock13.pict;
73=0=p=BRICK7.pict;
77=0=p=rock12.pict;
81=0=p=nice69.pict;
85=0=p=nice102.pict;
89=0=p=nice19.pict;
93=0=p=nice6.pict;
97=0=p=NEUT6.pict;
101=0=p=BRICK4.pict;
105=0=p=rock10.pict;
109=0=p=brick5.pict;
113=0=p=nice20.pict;
117=0=p=nice71.pict;
121=0=p=nice104.pict;
125=0=p=nice8.pict;
129=0=p=rock11.pict;
133=0=p=BRICK5.pict;
137=0=p=rock2.Pict;
141=0=p=rock12.pict;
145=0=p=nice29.pict;
149=0=p=nice9.pict;
153=0=p=nice10.pict;
157=0=p=nice30.pict;
161=0=p=rock13.pict;
165=0=p=BRICK2.pict;
169=0=p=brick6.pict;
173=0=p=neut6.pict;
177=0=p=nice31.pict;
181=0=p=nice105.pict;
185=0=p=nice41.pict;
189=0=p=nice107.pict;
193=0=p=rock10.pict;
197=0=p=rock14.pict;
201=0=p=BRICK1.pict;
205=0=p=BRICK7.pict;
209=0=p=nice13.pict;
213=0=p=nice50.pict;
217=0=p=nice51.pict;
221=0=p=nice66.pict;
225=0=p=BRICK1.pict;
229=0=p=rock13.pict;
233=0=p=neut3.pict;
237=0=p=rock12.pict;
241=0=p=nice52.pict;
245=0=p=nice15.pict;
249=0=p=nice16.pict;
253=0=p=nice53.pict;
257=0=p=NEUT1.pict;
261=0=p=brick4.pict;
265=0=p=BRICK8.pict;
269=0=p=brick6.pict;
273=0=p=nice54.pict;
277=0=p=nice17.pict;
281=0=p=nice72.pict;
285=0=p=nice18.pict;
289=0=p=rock10.pict;
293=0=p=neut6.pict;
297=0=p=rock2.pict;
301=0=p=NEUT3.pict;
337=0=b=blank;
337=0=e=end;
```

Nice - Variant

```
BEGIN;
1=0=p=BRICK1.pict;
5=0=p=rock2.pict;
9=0=p=neut1.pict;
13=0=p=rock14.pict;
17=0=p=nice59.pict;
21=0=p=nice3.pict;
25=0=p=nice23.pict;
29=0=p=nice4.pict;
33=0=p=rock11.pict;
37=0=p=BRICK8.pict;
41=0=p=neut3.pict;
45=0=p=BRICK2.pict;
49=0=p=brick4.pict;
53=0=p=NEUT1.pict;
57=0=p=rock13.pict;
61=0=p=BRICK7.pict;
65=0=p=rock12.pict;
69=0=p=nice13.pict;
73=0=p=nice6.pict;
77=0=p=nice8.pict;
81=0=p=nice29.pict;
85=0=p=NEUT6.pict;
89=0=p=BRICK4.pict;
93=0=p=rock10.pict;
97=0=p=rock11.pict;
101=0=p=BRICK5.pict;
105=0=p=rock2.Pict;
109=0=p=rock12.pict;
113=0=p=nice9.pict;
117=0=p=nice10.pict;
121=0=p=nice31.pict;
125=0=p=nice105.pict;
129=0=p=rock13.pict;
133=0=p=BRICK2.pict;
137=0=p=brick6.pict;
141=0=p=neut6.pict;
145=0=p=rock3.pict;
149=0=p=brick5.pict;
153=0=p=brick1.pict;
157=0=p=brick4.pict;
161=0=p=rock14.pict;
165=0=p=neut6.pict;
169=0=p=BRICK7.pict;
173=0=p=nice41.pict;
177=0=p=nice15.pict;
181=0=p=nice52.pict;
185=0=p=nice53.pict;
189=0=p=BRICK1.pict;
193=0=p=rock13.pict;
197=0=p=neut3.pict;
201=0=p=neut6.pict;
205=0=p=rock10.pict;
209=0=p=rock13.pict;
213=0=p=BRICK2.pict;
217=0=p=brick6.pict;
221=0=p=neut6.pict;
225=0=p=nice102.pict;
229=0=p=nice20.pict;
233=0=p=nice13.pict;
237=0=p=nice66.pict;
241=0=p=BRICK1.pict;
245=0=p=rock12.pict;
249=0=p=neut6.pict;
253=0=p=neut3.pict;
257=0=p=rock10.pict;
261=0=p=rock14.pict;
265=0=p=BRICK4.pict;
269=0=p=brick6.pict;
273=0=p=neut1.pict;
277=0=b=blank;
281=0=e=end;
```

Horror

```
BEGIN;
1=0=p=BRICK1.pict;
5=0=p=rock2.pict;
9=0=p=neut1.pict;
13=0=p=rock14.pict;
17=0=p=HORROR21.pict;
21=0=p=HORROR20.pict;
25=0=p=HORROR1.pict;
29=0=p=HORROR39.pict;
33=0=p=rock11.pict;
37=0=p=BRICK8.pict;
41=0=p=neut3.pict;
45=0=p=BRICK2.pict;
49=0=p=HORROR28.pict;
53=0=p=HORROR15.pict;
57=0=p=HORROR23.pict;
61=0=p=HORROR32.pict;
65=0=p=NEUT1.pict;
69=0=p=rock13.pict;
73=0=p=BRICK7.pict;
77=0=p=rock12.pict;
81=0=p=HORROR35.pict;
85=0=p=HORROR16.pict;
89=0=p=HORROR3.pict;
93=0=p=HORROR22.pict;
97=0=p=NEUT6.pict;
101=0=p=BRICK4.pict;
105=0=p=rock10.pict;
109=0=p=brick5.pict;
113=0=p=HORROR17.pict;
117=0=p=HORROR50.pict;
121=0=p=HORROR51.pict;
125=0=p=HORROR4.pict;
129=0=p=rock11.pict;
133=0=p=BRICK5.pict;
137=0=p=rock2.Pict;
141=0=p=rock12.pict;
145=0=p=HORROR33.pict;
149=0=p=HORROR14.pict;
153=0=p=HORROR40.pict;
157=0=p=HORROR12.pict;
161=0=p=rock13.pict;
165=0=p=BRICK2.pict;
169=0=p=brick6.pict;
173=0=p=neut6.pict;
177=0=p=HORROR29.pict;
181=0=p=HORROR7.pict;
185=0=p=HORROR8.pict;
189=0=p=HORROR53.pict;
193=0=p=rock10.pict;
197=0=p=rock14.pict;
201=0=p=BRICK1.pict;
205=0=p=BRICK7.pict;
209=0=p=HORROR5.pict;
213=0=p=HORROR13.pict;
217=0=p=HORROR30.pict;
221=0=p=HORROR11.pict;
225=0=p=BRICK1.pict;
229=0=p=rock13.pict;
233=0=p=neut3.pict;
237=0=p=rock12.pict;
241=0=p=HORROR19.pict;
245=0=p=HORROR26.pict;
249=0=p=HORROR18.pict;
253=0=p=HORROR24.pict;
257=0=p=NEUT1.pict;
261=0=p=brick4.pict;
265=0=p=BRICK8.pict;
269=0=p=brick6.pict;
273=0=p=HORROR6.pict;
277=0=p=HORROR2.pict;
281=0=p=HORROR54.pict;
285=0=p=HORROR10.pict;
289=0=p=rock10.pict;
293=0=p=neut6.pict;
297=0=p=rock2.pict;
301=0=p=NEUT3.pict;
337=0=b=blank;
337=0=e=end;
```

Horror - Variant

```
BEGIN;
1=0=p=rock14.pict;
5=0=p=rock2.pict;
9=0=p=neut1.pict;
13=0=p=brick1.pict;
17=0=p=HORROR21.pict;
21=0=p=HORROR15.pict;
25=0=p=HORROR32.pict;
29=0=p=HORROR7.pict;
33=0=p=rock11.pict;
37=0=p=BRICK8.pict;
41=0=p=neut3.pict;
45=0=p=rock10.pict;
49=0=p=brick5.pict;
53=0=p=brick6.pict;
57=0=p=rock13.pict;
61=0=p=BRICK7.pict;
65=0=p=brick1.pict;
69=0=p=HORROR51.pict;
73=0=p=HORROR8.pict;
77=0=p=HORROR3.pict;
81=0=p=HORROR11.pict;
85=0=p=neut3.pict;
89=0=p=brick2.pict;
93=0=p=neut6.pict;
97=0=p=rock10.pict;
101=0=p=rock13.pict;
105=0=p=brick1.pict;
109=0=p=HORROR23.pict;
113=0=p=HORROR14.pict;
117=0=p=HORROR27.pict;
121=0=p=HORROR1.pict;
125=0=p=NEUT6.pict;
129=0=p=BRICK4.pict;
133=0=p=rock10.pict;
137=0=p=rock12.pict;
141=0=p=neut6.pict;
145=0=p=brick2.pict;
149=0=p=rock14.pict;
153=0=p=neut3.pict;
157=0=p=rock11.pict;
163=0=p=rock2.Pict;
167=0=p=brick1.pict;
171=0=p=HORROR18.pict;
175=0=p=HORROR19.pict;
179=0=p=HORROR31.pict;
183=0=p=HORROR2.pict;
187=0=p=rock13.pict;
191=0=p=neut6.pict;
195=0=p=rock3.pict;
199=0=p=rock2.pict;
203=0=p=NEUT3.pict;
207=0=p=brick5.pict;
211=0=p=rock10.pict;
215=0=p=rock14.pict;
219=0=p=BRICK7.pict;
223=0=p=BRICK1.pict;
227=0=p=HORROR5.pict;
231=0=p=HORROR13.pict;
235=0=p=HORROR30.pict;
239=0=p=HORROR6.pict;
243=0=p=rock12.pict;
247=0=p=rock13.pict;
251=0=p=brick6.pict;
255=0=p=rock11.pict;
259=0=p=BRICK8.pict;
263=0=p=brick6.pict;
267=0=p=BRICK1.pict;
271=0=p=rock10.pict;
275=0=p=brick2.pict;
279=0=p=neut3.pict;
283=0=p=brick5.pict;
287=0=p=brick6.pict;
291=0=p=rock13.pict;
295=0=p=BRICK7.pict;
299=0=p=brick1.pict;
303=0=p=rock13.pict;
307=0=p=rock2.pict;
311=0=p=neut1.pict;
315=0=p=brick3.pict;
319=0=b=blank;
323=0=e=end;
```

Bibliography

[1] Belliveau J W, Kennedy D N, McKinstry R C, Buchbinder B R, Weisskoff R M, Cohen M S, Vevea J M, Brady T J, Rosen B R (1991). Functional Mapping of the human visual cortex by magnetic resonance imaging. *Science*, **54:** 716-719.

[2] Kwong K, Belliveau J W, Chesler D A, Goldberg I E, Weisskoff R M, Poncelet B P, Kennedy D N, Hoppel B E, Cohen M S, Turner R, Cheng H, Brady T J, and Rosen B R (1992). Dynamic magnetic resonance imaging of human brain activity during primary sensory stimulation. *Proc Natl Acad Sci, USA*, **89:** 5675-5679.

[3] . Lukas S, Dobrosielski M, Chiu T, et al. (1993). A nonferrous instrumental joystick device for recording behavioral responses during magnetic resonance imaging and spectroscopy. *Pharm Biochem & Behavior,* **46:** 781-785.

[4] Ives J R, Warach S, Schmitt F, Edelman R R, and Schomer D L (1993). Monitoring the patients EEG during echo planar MRI. *Electroenceph & Clin Neurophys*, **87:** 417-420.

[5] Sergent J, Ohta S, and MacDonald B (1992). Functional neuroanatomy of face and object processing. A positron emission tomography study. *Brain*, **115:** 15-36.

[6] Grodd W, Schneider T F, Klose U, and Nagele T (1995). Functional magnetic resonance imaging of psychological functions with experimentally induced emotion. *Radiologe*, **35:** 283-289.

[7] Sergent J, Ohta S, and MacDonald B (1992). Functional imaging of face and object processing. A Positron Emission Tomography Study. *Brain*, **155:** 15-36.

[8] Cohen R, Semple W, Gross M, King A, and Nordahl T (1992). Metabolic brain pattern of sustained auditory discrimination. *Exp Brain Res,* **92:** 165-172.

[9] Corbetta M, Miezin P, Dobmeyer S, Shulman G, and Peterson S (1991). Selective and divided attention during visual discrimination between shape, color, and speed: Functional Anatomy by positron emission tomography. *J Neurosci*, **11:** 2383-2402.

[10] George M, Ketter T, Gil D (1993). Brain regions involved in recognizing facial emotion or identity: An oxygen-15 PET study. *J Neuropsych & Clin Neurosci*, **5:** 384-394.

[11] Papez J W (1937). A proposed mechanism for emotion. *Arch Neurol Psychiatry*, **38:** 725-743.

[12] Diamond M C, Scheibel A B, and Elson L M. *The Human Brain Coloring Book.* Harper Collins Publishers, Inc.: NewYork, NY, 1985.

[13] Aggleton J. *The Amygdala.* Wiley-Liss: New York, NY, 1992.

[14] Nolte J. *The Human Brain: An Introduction to Functional Anatomy.* Mosby Year Book Publishing, Inc.: St. Louis, 1988.

[15] Kandell E R, Schwartz J H, and Jessel T M. *Principles of Neural Science.* Third Edition. Simon & Schuster: Norwalk, CT, 1991.

[16] Gleitman H. *Psychology.* Second Edtion. W W Norton Co.: New York, NY, 1986.

[17] Price J L and Amaral D G (1981). An Audiographic study of the central nucleus of the monkey amygdala. *J Neurosci*, **1:** 1242-1259.

[18] Amaral D, Price J L, Pitkanan A, and Carmichael S T (1922). Anatomical Organization of the Primate Amygdaloid Complex. *The Amygdala.* Wiley-Liss: New York, NY, p. 1-66.

[19] McDonald A J (1992). Cell types and intrinsic connections of the amygdala. *The Amygdala.* Wiley-Liss: New York, NY, p. 1-66.

[20] Rolls E T (1992). Neurology and Functions of the Primate Amygdala. *The Amygdala.* Wiley-Liss: New York, NY, p. 143-165.

[21] Ledoux J E (1992). Emotion and the Amygdala. *The Amygdala.* Wiley-Liss: New York, NY, p. 339-351.

[22] Rolls E T (1986). A theory of emotion and its application to understanding the neural basis of emotion. *Emotions: Neural and Chemical Control.* Japan Scientific Press: Tokyo, p. 325-344.

[23] Jacobs B L and McGintry D J (1972). Participation of the amygdala in complex stimulus recognition and behavioral inhibition: Evidence from unit studies. *Brain Res,* **36:** 431-436.

[24] Davis M (1992). The Role of the Amygdala in conditioned fear. *The Amygdala*. Wiley-Liss: New York, NY, p. 255-305.

[25] Weiskrantz (1956). Behavioral changes accociated with ablation of the amygdaloid complex in monkeys. *J Comp Physiol Psychol*, **49:** 381-391.

[26] Jones B and Mishkin M (1972). Limbic lesions and the problem of stimulus reinforcement associations. *Exp Neurol*, **36:** 362-377.

[27] Mishkin M and Aggleton J (1981). Multiple functional contributions of the amygdala in the monkey. *The Amygdaloid Complex*. Elsevier: Amsterdam, p. 409-420.

[28] Downer J D C (1961). Changes in visual gnostic function and emotional behavior following unilateral temporal lobe damagein the split brain monkey. *Nature,* **191:** 50-51.

[29] William T and Yuh C. *MRI of Head and Neck Anatomy*. Churchill Livingstone Inc: New York, NY, 1994.

[30] Cahill D R, Orland M J, and Miller G M. *Atlas of Human Cross-sectional Anatomy: with CT and MR Images*. Third Edition. Wiley-Liss Inc: New York, NY, 1995.

[31] Mansfield P (1977). Multiplanar image formation using spin echoes. *J Solid State Physics*, **10:** L55-58.

[32] Lauterbur P C (1973). Image formation by induced local interactions: examples employed by nuclear magnetic resonance. *Nature,* **242:** 190-191.

[33] Hanstock C, Rothman C, Prichard J W, and Jue T (1988). Spatially localized 1H NMR spectra of metabolites in the human brain. *Proc Natl Acad Sci, USA,* **85:** 1821-1825.

[34] Brown T R, Kincaid B M, and Ugurbil K (1982). NMR chemical shift imaging in three dimensions. *Proc Natl Acad Sci, USA*, **79:** 3523-3526.

[35] Haacke E M, Masaryk T J, Wielopolski P A, Zypman J A, Tkach J A, Amartur S, Mitchell J, Clampitt M, and Paschal C (1990). Optimizing blood vessel contrast in fast three-dimensional MRI. *Mag Res Med*, **14:** 202-221.

[36] Kim S-G and Ackerman J J H (1990). Quantification of regional blood flow by monitoring of exogenous tracer via nuclear magnetic resonance spectroscopy. *Magn Res Med*, **14:** 266-282.

[37] Rosen B R, Belliveau J W, Vevea J M, and Brady T J (1990). Perfusion imaging with NMR contrast agents. *Magn Res Med,* **14:** 249-265.

[38] Roy C S and Sherington C S (1896). On the Regulation of the Blood Supply of the Brain. *J Physiol* (London), **11:** 85-108.

[39] Fox P T, Minutmen M A, Raichle M A, Miezin F M, Allman J M, Essen D C (1986). Mapping of the Human Visual Cortex with Positron Emission Tomography. *Nature,* **323:** 806-809.

[40] Fox P T and Raichle M E (1986). Focal physiological uncoupling of cerebral blood flow and oxidative metabolism during somatosensory stimulation in human subjects. *Proc Natl Acad Sci, USA,* **83:** 1140-1144.

[41] Fox P T, Raichle M E, Minutmen M A, and Dence C (1988). Nonoxidative glucose consumption during focal physiologic neural activity. *Science,* **241:** 462-464.

[42] Phelps M E, Kuhl D E, and Mazziotta J C (1981). Metabolic mapping of the brain's response to visual stimulation: studies in humans. *Science*, **211:** 1445-1448.

[43] Prichard J, Rothman D, Novotny E, Petroff O, Kuwabara T, Avison M, Howseman A, Hanstock C, and Schulman R (1991). Lactate rise detected by 1H NMR in human visual cortex during physiologic stimulation. *Proc Natl Acad Sci, USA,* **88:** 5829-5831.

[44] Sandman C A, O'Halloran J P, and Isenhart R (1984). Is there an Evoked Vascular Response? *Science*, **224:** 1355.

[45] Fox P T, Miezen F M, Allman J M, Van Essen D C, and Raichle M E (1987). Retinotropic Organization of Human Visual Cortex Activity Mapped With Positron Emission Tomography. *J Neurosci,* **7:** 913.

[46] Fox P T and Raichle M E. Stimulus Rate Dependance of Regional Cerebral Blood Flow in Human Striate Cortex, Demonstrated by Positron Emission Tomography. *J Nerophysiol,* **51:** 1109.

[47] Roland P E, Larsen B, Lassen N A, and Shinkoj E (1980). Supplementary motor areas and other cortical areas in the orginization of voluntary movements in man. *J Neurophysiol,* **43:** 118-136.

[48] Fox P (1993). Human brain mapping, **1:** 1-2.

[49] Pauling L and Coryell C (1936). The magnetic properties of and structure of hemoglobin, oxyhemoglobin, and carbonmonoxyhemoglobin. *Proc Natl Acad Sci, USA,* **22:** 210-216.

[50] Ogawa S, Lee T M, Nayak A, and Glynn P (1990). Oxygenation-sensitive contrast in magnetic resonance image of rodent brain at high magnetic fields. *Magn Res Med,* **14:** 68-78.

[51] Ogawa S and Lee T M (1990). Magnetic resonance imaging of blood vessels at high fields: in vivo and in vitro measurements and image simulation. *Magn Res Med,* **16:** 9-18.

[52] Ogawa S, Lee T M, Kay A R, and Tank D W (1990). Brain magnetic resonance imaging with contrast dependent blood oxygenation. *Proc Natl Acad Sci, USA,* **87:** 9868-9872.

[53] Bullmore E, Brammer M, Mellers J, Howard R, Sham P, and David A. Deciding which pixels are activated in functional MR images. Abstract. Institute of Psychiatry, London SE5 8AF, United Kingdom.

[54] Scarth G, Somorjai R, Alexander M, Wowk B, Wennerberg A B A, and McIntyre M C. Cluster Analysis of Functional Brain Images. Abstract.

Rosner B. Fundamentals of Biostatistics. Duxbury Press: Belmont, CA. 1990.

[56] *The NB-TIO-10 User Manual.* National Instruments. Austin, TX. 1994.

[57] *NI-DAQ Software Reference Manual for the Macintosh.* National Instruments. Austin, TX. 1993.

[58] Haus H A and Melcher J R. *Electromagnetic Fields and Energy.* Prentice-Hall: Englewood Cliffs, NJ. 1989.