Fall 1-31-2015

Visualeyes 2020: a software suite to integrate instrumentation to study the near triad of vergence

Stephen J. Lestrange
New Jersey Institute of Technology

ABSTRACT

VISUALEYES 2020: A SOFTWARE SUITE TO INTEGRATE INSTRUMENTATION TO STUDY THE NEAR TRIAD OF VERGENCE

by
Stephen J. Lestrange

A complete instrumentation suite has been constructed for use in a randomized clinical trial to be funded from the National Institute of Health and study the underlying potential mechanism(s) of vision therapy. This suite is designed to track rotation of each eye (vergence) as well as measure the lens power of the eye's lens (accommodation). The system is designed to dissect the near triad which is composed of the pupil constriction, vergence and accommodation. The visual targets are programmed to be shown on two sets of computer screens which allowed vergence, accommodation and proximal vergence cues to be presented in isolation or in combination to study the Maddox components of vergence. This instrument uses FDA approved devices for subject safety. In our custom LabView based software, the Haploscope is able to generate ocular stimuli from programmed scripts, record data, as well as perform other various experimental requirements. Post processing is done in a MATLAB GUI environment. The results support that vergence responses can be studied using one to several visual cues.

VISUALEYES 2020: A SOFTWARE SUITE TO INTEGRATE INSTRUMENTATION TO STUDY THE NEAR TRIAD OF VERGENCE

by
Stephen J. Lestrange

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Biomedical Engineering

Department of Biomedical Engineering

January 2015

APPROVAL PAGE

VISUALEYES 2020: A SOFTWARE SUITE TO INTEGRATE INSTRUMENTATION TO STUDY THE NEAR TRIAD OF VERGENCE

Stephen J. Lestrange

Dr. Tara L. Alvarez, Thesis Advisor Professor of Biomedical Engineering, NJIT	Date
Dr. Mesut Sahin, Committee Member Associate Professor of Biomedical Engineering, NJIT	Date
Dr. Max Roman, Committee Member Assistant Research Professor of Biomedical Engineering, NJIT	Date

BIOGRAPHICAL SKETCH

Author: Stephen J. Lestrange

Degree: Master of Science

Date: January 2015

Undergraduate and Graduate Education:

- Master of Science in Biomedical Engineering,
New Jersey Institute of Technology, Newark, NJ, 2015

- Bachelor of Science in Biomedical Engineering,
New Jersey Institute of Technology, Newark, NJ, 2014

- Bachelor of Arts in History,
Wake Forest University, Winston-Salem, NC, 2009

Major: Biomedical Engineering

To Kelly and ‘Rupert’

ACKNOWLEDGMENT

I would like to acknowledge my Thesis Advisor and Undergraduate Advisor, Dr. Tara Alvarez for providing me with the tools and advisement necessary in order to complete this project. I would also like to acknowledge my other committee members, Dr. Mesut Sahin, and Dr. Max Roman for their advisement in my thesis. I would like to thank my fellow lab mates: Raj Jaswal, Henry Talasan, Chang Yaramothu, and Nils Warfving. Thanks to NJIT for providing me the environment to complete my work. Also, I would like to acknowledge my funding source, NSF grant #MRI CBET1228254.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 The Visual System

Of our five senses, one can make an argument that the visual system is the sense which we rely the heaviest. Our visual system allows us to take in information from distances we are able to turn into direct action. Without consciously thinking, we can divert our gaze from a book in front of us to a clock across the room. As fast as we can think to ourselves, “what time is it?” our two eyes have independently rotated so that their intersection point is at the clock and the clock comes into focus by the minute adjustments in our focusing lens.

With the incredible ability of the eyes comes great complexity. Like one can argue that vision is our most important sense, the eyes are the most complex sensory organ. From the outside of the eye inwards, the sclera, the choroid, and the retina form the major divisions of the eye. With the exception of front of the eye where light enters, the sclera is made of dense opaque connective tissue that gives the eye its structure. The choroid is the vascular layer that nourishes the eye, providing oxygen transportation and waste removal. This vascularization is required by the retina, the nervous sensory tissue of the eye that is responsible for turning incoming light to electrical signals to the brain.

Light enters the eye through the cornea, the protective and clear portion of the eye. Light then passes into the anterior chamber that covers both the iris and pupil. The iris is a muscular body that controls the amount of light that is allowed into the eye. It can constrict like the aperture of a camera, and relax to allow more light in in low light conditions. In the center of the iris is the pupil where light enters the interior chamber of the eye. Once inside the eye, light passes through the vitreous humor, water based gelatin that helps provides structure to the eye. The light will end its trip through the eye at the retinal layer.

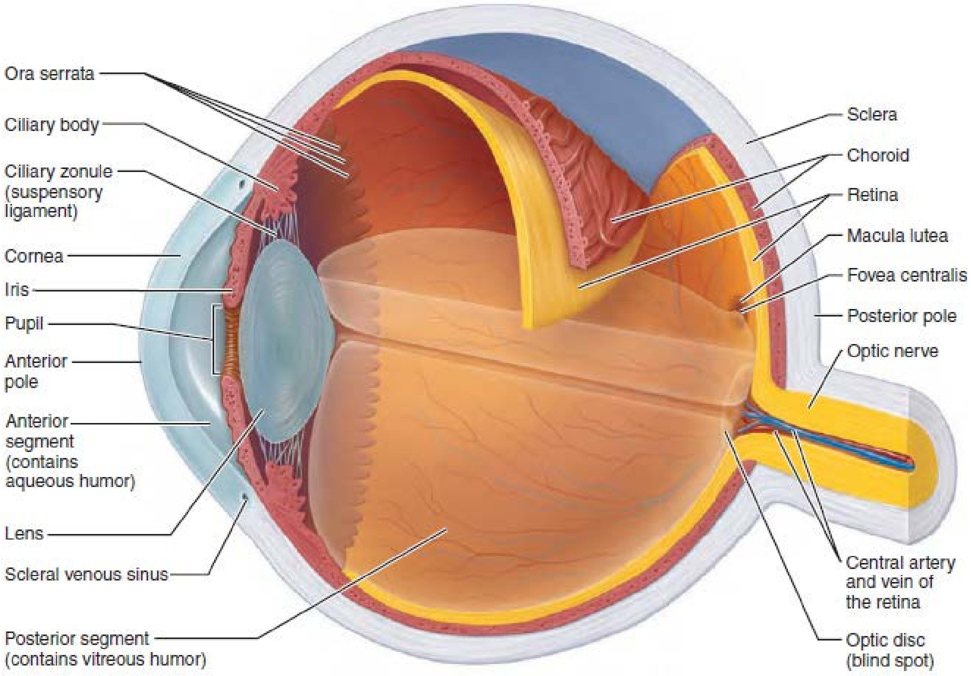


Figure 1.1 Basic structure of the eye. [1]

If the eye is complicated, it is the retina that is the most complicated portion of the eye. The five main cell types of the eye are photoreceptors, bipolar cells, amacrine cells, horizontal cells, and ganglion cells. The photoreceptors are the ones most applicable to this paper. Of the photoreceptors, there are two types: rods and cones. [2] The retina is comprised of different areas with different concentrations of photoreceptors. The area with that highest overall concentration is the macula lutea. This region in the back of the eye has specialized structures for high acuity vision. Located in the center of the macula is the fovea centralis, the area of overall highest acuity. [1] Despite the macula lutea having the highest acuity, there is a section of the retina that is not sensitive to light at all, aptly name the blind spot, where the optic nerve forms from the retina.

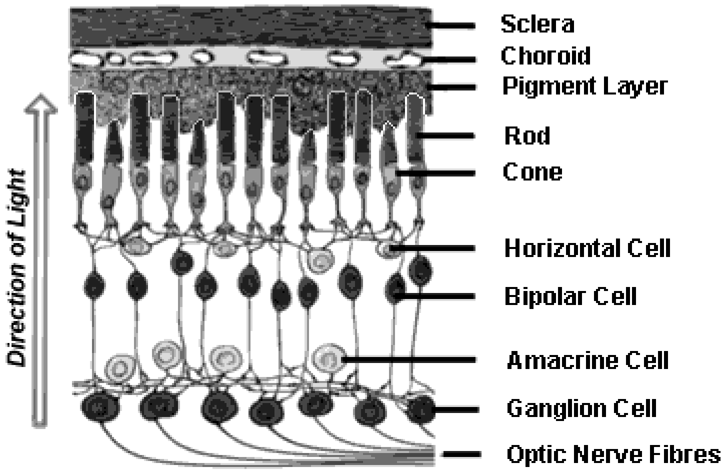


Figure 1.2 Diagram of retinal layers. [3]

As light strikes the retina, the photoreceptors become excited and send electrical signals via the optic nerve to the posterior occipital lobe of the brain for interpretation. The optic nerves from each of the eyes cross at the optic chiasm in the hypothalamus, causing the right side of the primary visual cortex to be responsible for the left half of the visual field and the left side to be responsible for the right visual field.

Our eyes are not fixed inside our orbit, but are able to move around independent of the motion of our head. Because of the superior, inferior, medial, and lateral recti muscles and the superior and inferior obliques, we are able to effortlessly direct our eyes so that the object of interest is project to the fovea on the back of the retinal. The fovea has the greatest concentration of cones and hence allows for the greatest resolution of an object. The superior and inferior oblique muscles are responsible for intorsion and extorsion movements of the eyes, respectively. The medial rectus acts as a principal adductor, the lateral rectus as a principal abductor, the inferior rectus as the principal elevator and superior rectus as the principal depressor. [4] As is the common notation, this paper will measure the degree of rotation of the eyes.

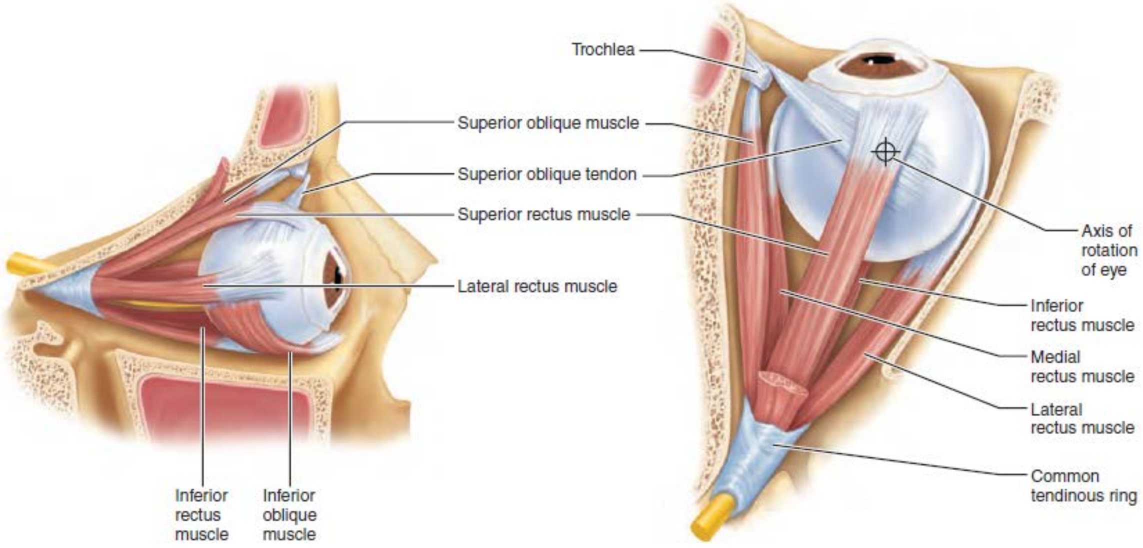


Figure 1.3 Anatomy of extraocular muscles. [1]

1.2 Vergence System

Vergence movements are eye movements that utilize the medial and lateral recti muscles. These extraocular muscles rotate the eyes within the transverse plane. The two major types of vergence movements are convergence and divergence. Convergence is the inward rotation of the eyes toward one another, going ‘cross eyed’ or looking towards one’s nose. A subject will exhibit this movement when a visual target is moving closer to them. Divergence is the outward rotation of the eyes away from each other as a target moves farther away. To ensure that we have clear and single vision, our visual system uses three clues: disparity, retinal blur, and proximity. Disparity is the difference between where a target of interest is projected onto the retina and the current position of the fovea. The fovea contains the highest density of cones which gives the fovea the great visual resolution.   Retinal blur stimulates accommodation that causes the eyes to change their focus. However, accommodation and vergence are linked via the cross over links known as the AC/A (accommodative convergence to accommodation) and CA/C (convergence accommodation to convergence) ratios. Hence, under normal viewing conditions, blur stimulates not only accommodation but also vergence. Proximal cues are those that are developed through experience and prior knowledge of the target. For example, if I know a person is approximately 5-6 ft tall, but appears to me as smaller, then he or she must be farther away.

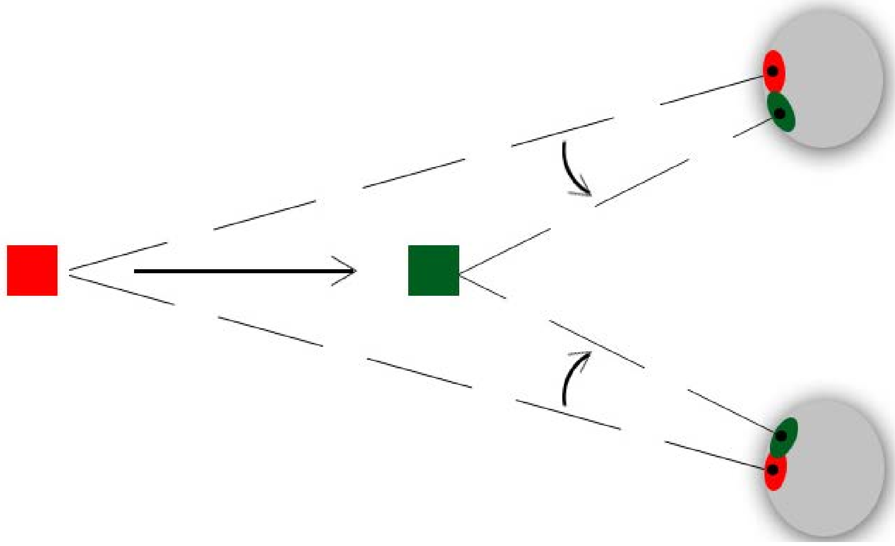


Figure 1.4 Disparity vergence movements. In this case, divergence would be the movement from green to red and convergence would be the movement from red to green.

We seek to quantify vergence as a function of the position of the eyes and the speed at which a subject can fix their eyes upon the target. If we present a target to a subject at a known length, and their eyes converge or diverge to that target at a different distance away from them, we can call this fixation disparity. Fixation disparity is the vergence error between the fixation point and the intersection of the gaze of each eye. [5] A large factor vergence speed and ability is prediction and anticipation. Our instrument will need to include engineering to make the stimuli appear with no apparent pattern. Otherwise, the subject will be able to predict where their next target will be and cause their vergence data to appear with smaller fixation disparity and increase velocity than would have occurred otherwise. [6]

1.3 The Treatment of Binocular Dysfunctions

About 4%-8% [7] of people have an oculomotor dysfunction called convergence insufficiency (CI), a condition were a patient is unable to comfortably point their gaze at a target and create one fused image when objects are located close to them. [8] Another binocular dysfunction is called amblyopia, a weakness in vision in one eye. [9] Common treatments for CI and amblyopia include physical therapy by the way of vision therapy. By enhancing the capabilities of the vision system, the hope is that the subject’s vision system will be able to adapt and do what it was once unable to accomplish.[8]The gold standard for amblyopia is to cover the stronger eye with a patch and force the vision system to rely on the weaker eye.[9] The only vision therapy that has been validated within a randomized clinical trial (RCT), for CI is the office based vergence and accommodative therapy with home reinforcement (OBVAT) where this RCT was conducted as part of the convergence insufficiency treatment trial (CITT).

There are multiple issues with these treatments, not because they do not work, but because people are unwilling to comply with repetitive visual tasks which the vast majority of patients find boring. With the use a brock string, a person can train their vergence system by being presented multiple targets at varying distances away and just practicing diverting their gaze from one to another. Patching an eye removes binocular vision and can cause anxiety, especially in children. [9]

It has been shown that office based vision training is more effective than training in the home, but compliance suffers and subjects may not have the time or financial resources to participate in vision therapy conducted within an optometrist's office which last for several weeks.[8]

In order to study the underlying mechanisms that lead to the high efficacy of vision training, a traditional vision therapy scheme must be devised and used as the gold standard by which the experimental trials can be compared against. One way to assess vision therapy is to record disparity and accommodative vergence responses through the use of a modified Haploscope. The Haploscope is a device able to present a target to each one of the eyes independently which trains disparity vergence while keeping accommodative vergence constant. Clinicians hypothesize that training disparity may be one major mechanism by which vision therapy results in a sustained reduction of symptoms.[1] By changing where the stimulus is presented as a function of how much the eye must rotate from optical infinity (straight ahead) to inward rotation (convergence), we can simulate a target presented to the subject at various distances by adjust disparity. Disparity is the angular difference of where an object is interest falls in the back of the eye compared to where the fovea is currently located.

1.4 Haploscopes

A Haploscope is a device that can present an image to each eye independently. By doing so, a Haploscope can stimulate a greater range of disparity vergence in a small physical space. More importantly, because a Haploscope only stimulates changes in disparity, the vergence system can be dissected to study disparity vergence in isolation without accommodative or proximal cues. There are many visual experimental set-ups with varying schemes of presenting stimuli and recording ocular movements. In the past, lasers have been projected onto a screen, light bulbs have been lit in sequence, or stimuli have been physically moved using motors, but the advent of the computer monitor has made most other forms of stimuli presentation obsolete. While our device will allow for the use of legacy devices such as an LED stick. VNEL's former Haploscope is shown schematically in Figure 1 below. However, to study the Maddox components of vergence our Haploscope will be expanded to create visual stimuli through the use of its five computer monitors. The schematic of the new Haploscope using five computer monitors will be discussed within the Methods and Materials section of this thesis.

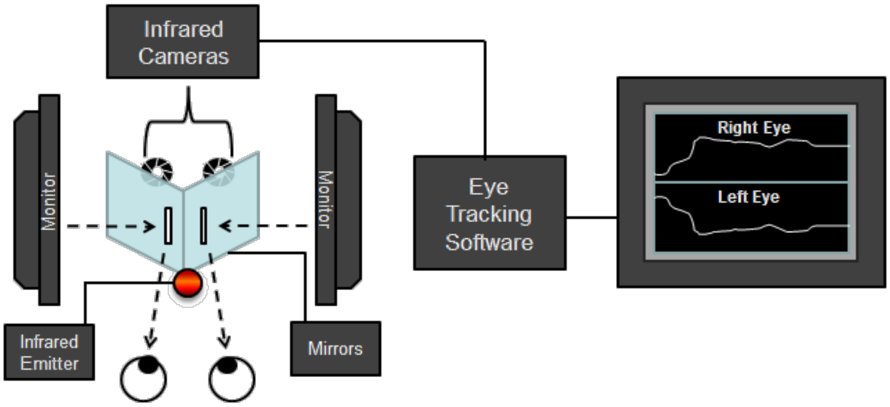


Figure 1.5 Basic Haploscope layout. This layout is designed to stimulate disparity vergence only while keeping accommodative and proximal vergence constant.

CHAPTER 2

METHODS AND MATERIALS

2.1 Construction of Haploscope Table

A table was needed to mount all of our instrumentation for this Haploscope construction. The table needed to be stable where the components would not move, which could lead to artifacts within the data. It was important to consider human ergonomic considerations and total overall stability when designing the table. It was also necessary to allow the tables to collapse and come apart for installation into their final laboratory.

Construction of the tables was done primarily with premium grade 2”x4” nominal lumber and ¾” particle board. These materials were chosen for their ease of use and relatively low cost. For fasteners, common drywall screws were used in various sizes, as well as the Kreg Joinery system. This system allows easy joinery of both 2”x4” and ¾” material. Also used for fasteners were low grade steel 3/8” bolts, 3/8” washers, and 3/8” nuts. These were chosen to be used for the parts that are to be removable. The idea being to transport and place the table in the laboratory, the Haploscope would simply be transported in a truck in pieces and bolted together on site. Each Haploscope was made up of six major parts: four leg assemblies, a lower instrument table and an upper deck where the monitors would be mounted.

The leg assemblies were designed to be sturdy and quick to make. An ‘H’ pattern was chosen for each leg for its simplicity and the fact that each pair of legs would provide a stable 4-point base for instrumentation. This meant that the larger legs for the upper deck and the smaller legs for the lower table would be able to stand independent of each other. This was not an unimportant consideration when considering the net worth of the instrumentation that was to be mounted to each assembly. One last consideration for the legs was that because sheer size of the completed assembly and especially the end most distant from the lower table, there was a lot of space available for storage under the table. By using two legs at the extremes of each side of the table, available storage space was maximized.

The lower instrumentation table is simply a table top constructed of a 2”x4” frame and particle board top, but it has two important features. First, the table has cut outs for the legs of the subject. Material was removed from the subject facing 2”x4” so that the chair they are sitting in could move up by an additional 2”. This would partially allow for the greatest possible comfort while sitting, unmoving, for up to an hour. The second major feature is the standoffs mounted to the lower instrument table that would allow for articulation with the larger upper deck. Only one point of contact was used each table can stand by itself, so the connection is really only there to line up the two major assemblies and prevent movement of one part with respect to the other.

The upper deck is the most complex subassembly in the entire build. This was because it had to feature winglets, to allow for the addition of extra monitor tracks that would otherwise extend beyond the width of the table, and have an area where instrumentation from the lower table can look through to the subject, nicknamed the ‘trench’. The lower skeleton was built exclusively of 2”x4” lumber. These pieces of wood would allow for stability and rigidity of the rather larger upper deck.

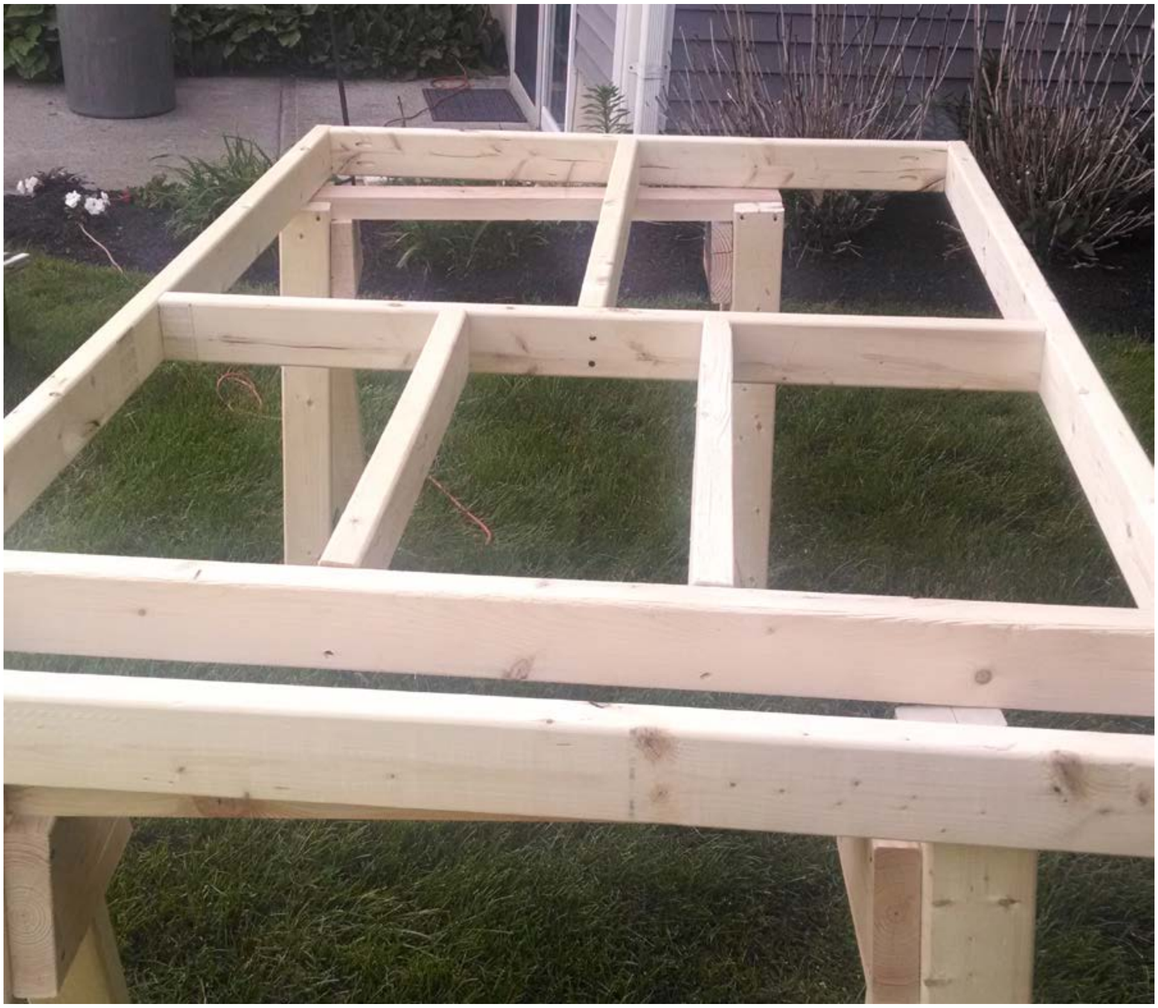


Figure 2.1 Upper deck of Haploscope. Shown here without particle board covering. Saw horses were used only for assembly.

To finish each upper deck, a sheet of particle board was used to cover the entire top, except the winglets. To remove material the covering material from the ‘trench,’ a flush cutting router bit was used. The design of the skeleton, with 2”x4” around the outside of the trench, allowed for a deeper trench and less light that would be able to enter the Haploscope through the bottom. Added for later convenience and instrumentation installation was a 1/8” deep channel along the center of the table that would serve as the axis of symmetry of the table and midline.

To finish, all edges of the top subject facing surfaces were routed with a round over bit so that no sharp edges were presented. The Haploscope was painted in matte black paint to minimize light reflection and make the Haploscope environment darker.

2.2 Monitor Layout and Design

With the tables built and the computer hardware set up, we had to develop a scheme for presenting small stimuli into meaningful visual stimulus. To do that, we must configure our stimuli screens so that there is a pair of screens that act together to form one complete visual stimuli. We can present stimuli on any part of the screen by software, but we seek to maximize the amount of each screen visible through the mirrors.

In the previous iteration of the Haploscope, there was only one pair of screens. Haploscopes work by projecting one image into each eye. The brain is able to fuse these two images into one simulated stimuli target. While the original Haploscope was able to stimulate stimuli at different distances away from the subject, the one pair of screens meant that there was only ever one focal length.

In our Haploscope, this length was the combined distance of the eye away from the reflective mirror and the reflective mirrors to the screen.

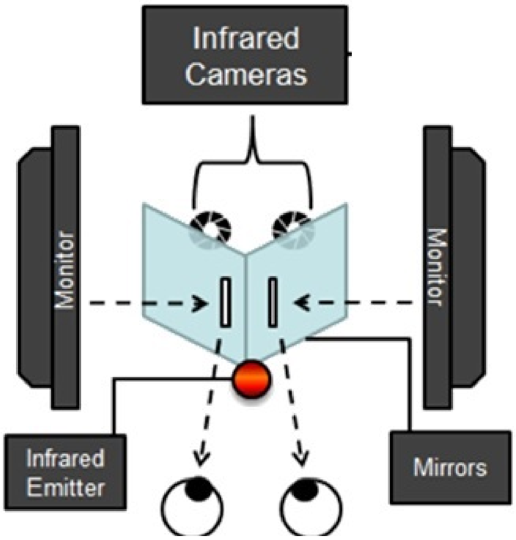


Figure 2.2 Basic Haploscope workings.

To stimulate accommodation, a second pair of screens would have to be added to the system in order to change the focal length, forcing the lens in the eye to accommodate. The original design was based on this principle. A pair of screens was added as shown in the figure below. There was room for another pair of partially reflective mirrors to stimulate disparity and accommodative vergence independently or simultaneously.

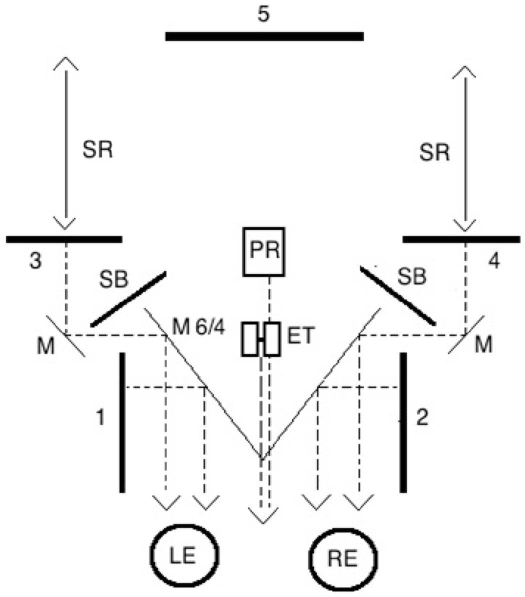


Figure 2.3 First tested design of Haploscope. It was designed as a proof of concept. It worked, but was had only limited simulated visual range.

This design worked well, but we found that by moving the secondary mirrors closer to the subject we gained a greater stimuli range. Moving the mirrors toward the subject also meant that the monitors would have to move closer to the middle of the Haploscope. With this configuration, we could use a larger area of the computer screens to present stimuli, and therefore, present stimuli targets at greater simulated distances.

.

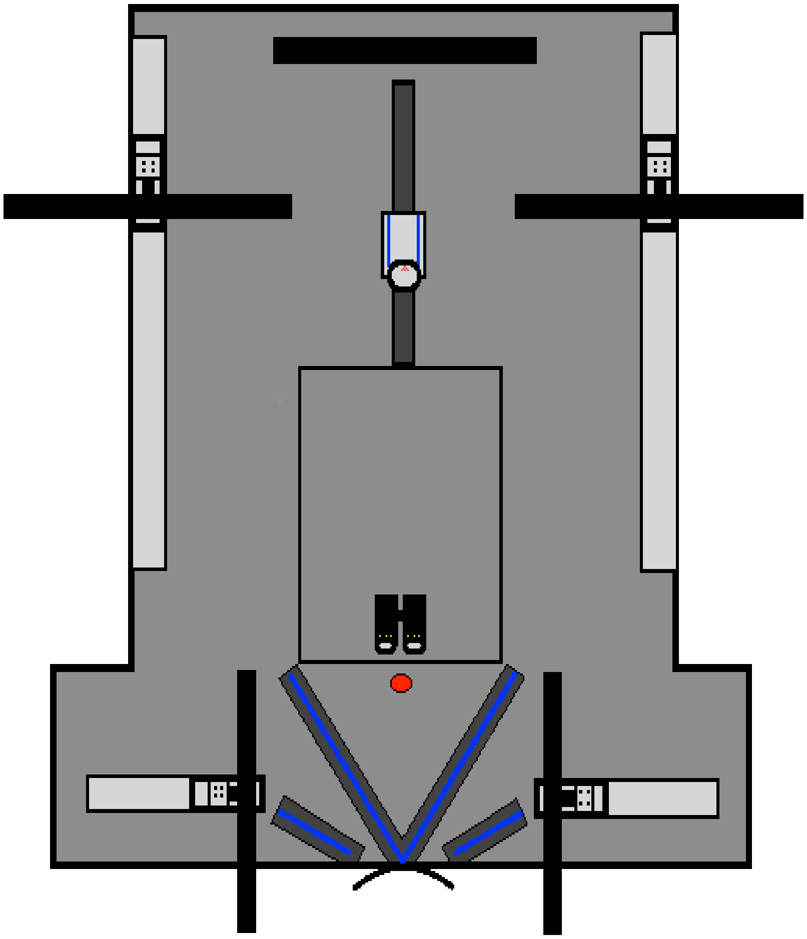


Figure 2.4 Final layout of Haploscope.

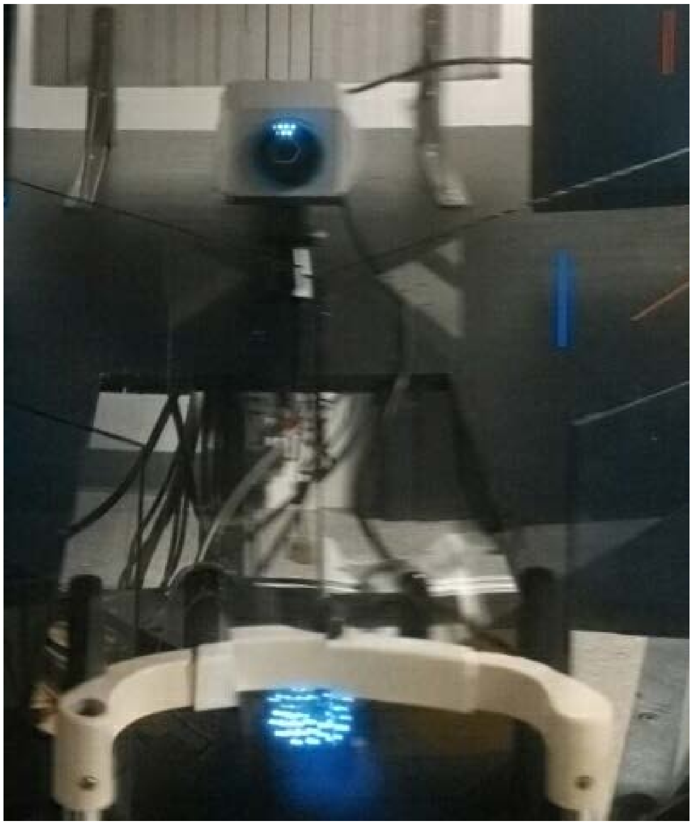


Figure 2.5 Instrumentation installed.

To summarize the workings of the newly designed Haploscope, the stimuli from the two sets of screens, front and back, are reflected to the eye by the primary and secondary mirrors. The back screens reflect first onto the secondary mirrors, which then reflect to the primary mirrors and to the eyes. The front screens do not reflect off of the secondary mirrors, but through them and on to the primary mirrors. Because both the primary and secondary mirrors are half silvered, we experience 50% loss at each reflection. This does not present a problem for us, as we have a lot of signal from the screens. Later, we actually had to turn the brightness down on the screens. The added benefit of this design is that both sets of screens experience approximately the same optical resistance and have their power attenuated the same. This way, the subject is not able to tell which set of screens are active by the brightness of the stimuli.

Another alteration that we made to improve our signal was to move the ISCAN vergence cameras closer to the subject. They are now 15cm away. We had to modify the existing Haploscope table by making the ‘trench’ larger, but in the final design, this modification was built into the design. Instrumentation sits in this trench so to be out of the way and aimed at the eyes in a way that it does not have to go through the mirrors. The PlusOptix accommodation camera cannot be moved closer or father away, as it has a fixed calibration distance of one meter. This is an FDA approved device. As part of the safety feature, if the PlusOptix is moved closer than 1m away the instrument will shut off.

At the end of the build, the fifth calibration screen was added. This was done because the calibration and validation of the instrument required a manual calibration board be placed there. Without this manual board, the lines that appear on the mirrors and are presented to the subject are totally worthless. Without a way to gauge where these lines appear in space, it would be impossible to get any meaningful data out of the Haploscope. Later, this fifth screen would be controlled so that it can display the calibration screen dynamically. Future use of this monitor includes experiments that stimulate version only movements such as saccades or smooth pursuit. VNEL plans to also conduct reading experiments where this fifth screen will be used.

In order to change the focal length of the near or far screens, we must physically move the screens themselves. To ensure that the screens move only in their intended direction, a track system was implemented. The monitor mounts had to be machined and installed onto the slide rails so that an experimenter can change the focal length of a stimulus that they are presenting. This gives our system more experimental possibilities. The mounts that we used are designed to hold a monitor, but are designed to be used vertically, not horizontally as we used them. Because of this, a mount had to be designed and made that would hold the screens. This was problematic in the old Haploscope, as the screens were not fixed to the Haploscope table and their orientation was controlled by their own OEM stands. They would get bumped and moved, throwing off the calibration. Currently, they are not allowed to move in any axis about the rear of the monitor where the screen meets its post and the only movement that they can undergo with respect to the table is the front and back motion that the track allows. Even the motion along the track is controlled by a set screw as to stop unwanted movement.

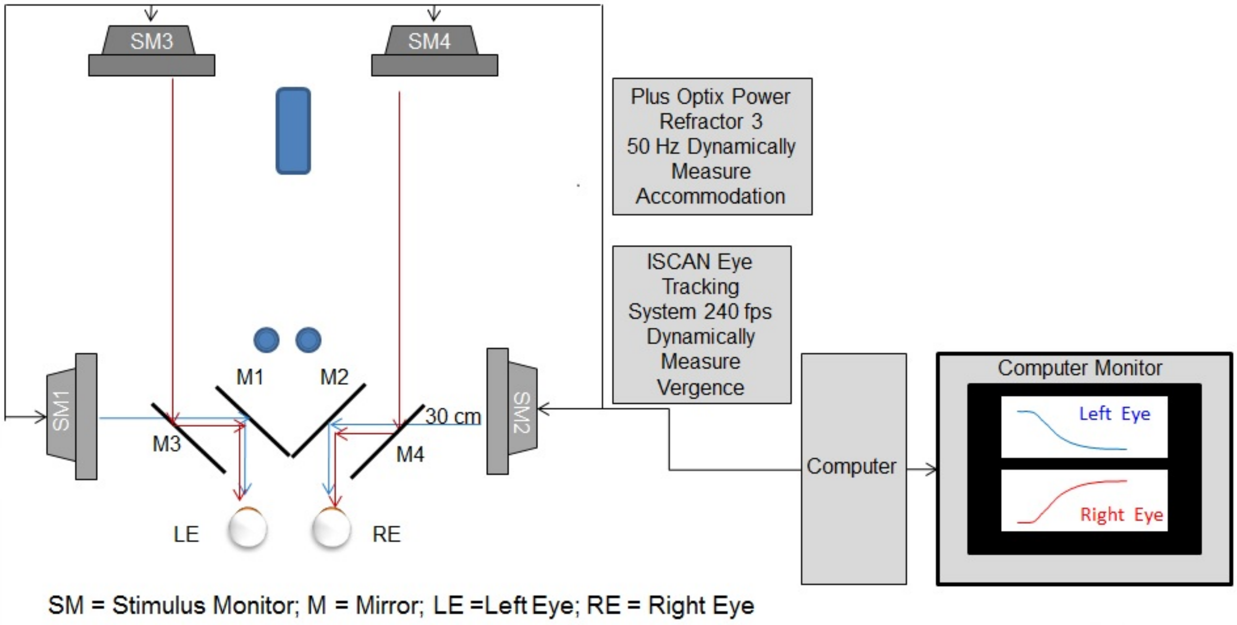

SM3
SM4
Plus Optix Power Refractor 3 50 Hz Dynamically Measure Accommodation
ISCAN Eye Tracking System 240 fps Dynamically Measure Vergence
M1
M2
SM1
SM2
M3
M4
30 cm
LE
RE
Computer
Computer Monitor
Left Eye
Right Eye
SM = Stimulus Monitor; M = Mirror; LE =Left Eye; RE = Right Eye

Figure 2.6 Haploscope layout summary.

2.3 ISCAN Vergence Tracker

To track the movement of eye movements from our subjects, we require the use of an eye tracker. As a visual target moves closer or farther way, the eyes need to rotate toward, if the target is coming closer, or rotate outward, if the target is going away, so that both eyes remain pointed directly at the target. There are a multiple of companies that make devices that measure this ocular rotation dynamically using various methods. ISCAN is a company that uses an infrared (IR) camera mounted externally, in goggles, or into a self-contained unit. [10] SensoMotoric Instruments makes eye trackers that can be mounted onto the head, goggles, and a USB model that can be placed into the corner of a laptop to monitor what a subject looks at. [11] SR Research makes devices that can be mounted on the head or externally, but they also make an MRI safe device. [12] Tobii makes devices that can be worn as glasses or mounted, but are meant for research and advertising firms to see what parts of images people's eye look towards. [13] A final eye tracking firm includes Cambridge Research Systems. [14]

Ultimately, it was decided to choose the ISCAN eye tracker. This system is camera based, so the instrumentation can be easily mounted to our instrumentation deck on the Haploscope. It has several axes of motion, enabling it to adjust to the position of comfort for the subject. It is also able to be mounted to the lower component deck, protrude only slightly above the surface of the instrument deck, and record signals from the eyes underneath the reflective mirror system. By placing the eye tracker in this way, we can get the best possible signal as we would lose at least 50% of our signal through the partially reflective mirrors. Also important was that the ISCAN system is able to output data as analog waveforms, allowing us to record this data into our LabView software, VisualEyes.

The ISCAN system works by constantly bathing the eyes in very low power IR light. OSHA's standard on near infra-red light onto the lens of the eye shall not exceed 10 mW/cm^2. The ISCAN system is very safe in that even if the entire power of the IR source was to be somehow directed into the eye, only 1.2 mW/cm^2 would enter the eye. [15] This safe low intensity IR light is reflected off the face in different intensities, but this IR light is not reflected from pupil of the eye. This absence of reflected light can be detected by two IR cameras, one for each eye. By taking the center of mass of area with IR light return below threshold, the center of the pupil can be found. The ISCAN IR tracker outputs the X and Y center of each pupil as a voltage value that can be easily read and recorded by our LabView data acquisition software.

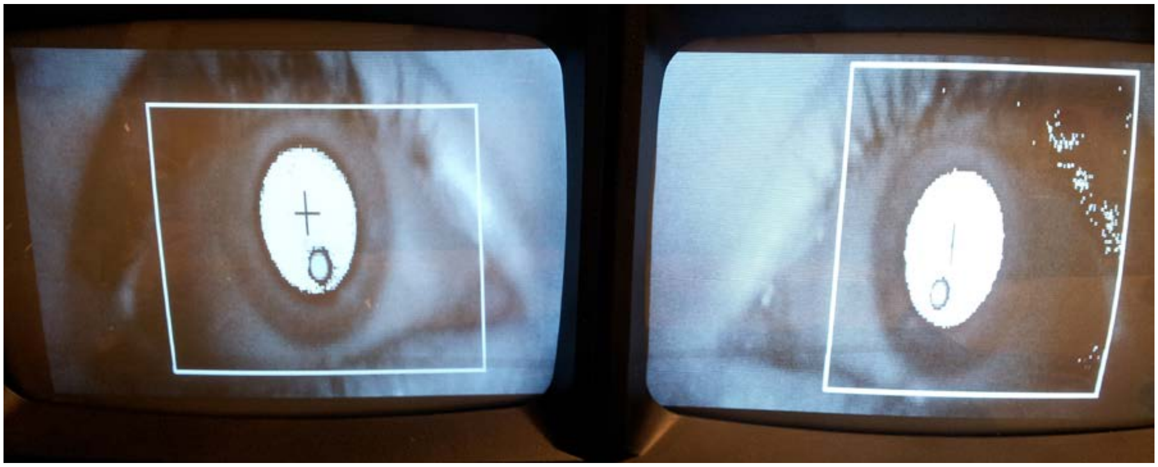


Figure 2.7 ISCAN camera view. Cross hairs are visible in the center of the highlighted pupil.

2.4 Plus Optix Power Refractor III Accommodometer

An accommodometer is for tracking the lens power of the eye's lens. While autorefractors which measure the static accommodation level of the eye are very common and most eye care professional use an autorefraction within their clinical exam, very few devices measure accommodation dynamically as a function of time. To the best of our knowledge, the Plus Optix PowerRefractor III is the only commercially available FDA approved device that can measure accommodation dynamically. Therefore, it became a design requirement that we measure our accommodation using this device.

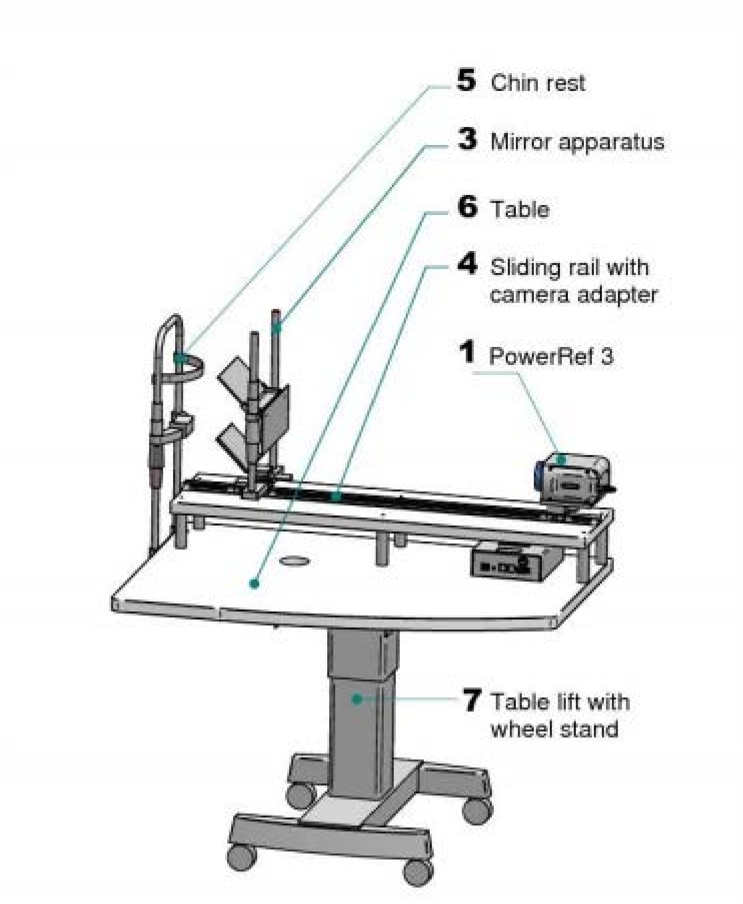

5 Chin rest
3 Mirror apparatus
6 Table
4 Sliding rail with camera adapter
1 PowerRef 3
7 Table lift with wheel stand

Figure 2.86 Power Refractor III.

The Power Refractor III works by eccentric photorefraction, which means infrared light is symmetrically presented to the visual system in pairs that are equal distance or eccentricity of the line of site. Using this technique, it is possible to measure the lens power of both eyes from a distance and a relatively high sampling rate. This is done by the analysis of light reflected from the retina of the eye back through the lens. The lens power is calculated by the returning distribution of light. It is also possible to calculate the gaze of the subject by their purkinje images. While the Power Refactor does not sample vergence at high enough rates to capture vergence movements with the resolution we desire, we will later use the vergence gaze data to synchronize the vergence data from the ISCAN system. [16] This is important because now we can measure vergence and accommodation simultaneously.

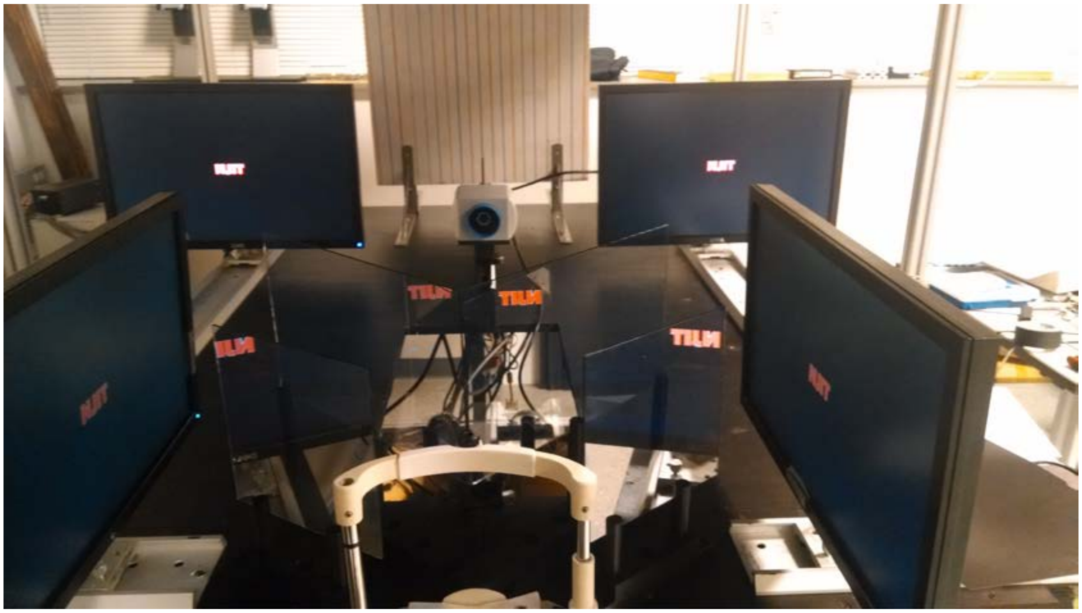


Figure 7 Power Refractor III as installed in Haploscope. Based upon the manufacturer's recommendation the system is automatically calibrated so long as it is 1m away from the subject.

2.5 Computer Hardware and Selection

Before the start of this project, the VNEL was using a custom LabView program which VNEL called VisualEyes2. It was capable of displaying images on only two stimuli monitors at a time and had high end computer specifications for when it was purchased in 2010. The legacy system had 2.00GHz, 3GB RAM, and four processor cores. Using this VE2 platform, VisualEyes3 was created so that it could do the exact same thing that VE2 could do, except on an indefinite amount of screens. As part of the proof of concept for our final device, we used this upgraded version of the legacy VisualEyes software to test our hypothesis. It quickly became apparent that the computer was not powerful enough to process the images on up to five display screens at a time. We found that as we increased demands on the system by adding additional stimuli, adding additional screens, and ramping the stimuli (smoothly moving the target), that our performance suffered greatly.

It was our requirement to get smooth movement of stimuli across the screen. We found that the upgraded legacy system was able to go from one static image to another static image across all five screens with no problem. Ramping would allow for the experiment designer to move a target away from the subject as a function of time and test the subject's ability to track the image. However, ramping of stimuli placed much harder demands on the computer as it would have to generate a new image every 5ms across many different screens. We found that the computer was not able to keep up with such demands and it would slow down, causing the ramps to jitter across the screen whenever the computer was able to catch up. This was a proof of concept failure for us.

It was clear to us that we had exposed a weakness of LabView. It is simply not designed for the generation of images that fast and our image processing demands would slow, if not freeze, the computer system. Therefore, we needed a new approach. We determined that if we used a gaming style computer and C++ based image processing software, we could create smooth ramps across all of our screens.

First, our computer hardware had to be upgraded. We started our hardware selection process at the motherboard. We chose a motherboard with multiple PCI x16 slots; we would need these computer slots for our two graphics cards, and NI DAQ card. In our previous computers, we would run out of physical room on our motherboard before all of the additional hardware was inserted. We knew that we would need one NI DAQ card for recording data, and two full sized graphics cards. These graphics cards are generally twice the size of a regular card, taking up two ports on the rear of the computer instead of the usual one port. Therefore, any motherboard that was designed to fit on a in a smaller chassis computer was immediately disqualified.

We needed to select a graphics card for inclusion into our new system. In the legacy system, we used two legacy Matrox cards with two DVD-D ports each. Four ports would not be enough for our new system, so for our proof of concept, we tried two Matrox M9148 cards with four ports each. These cards had the monitor ports that we needed, but were designed to run simple applications of low graphic intensity. Most importantly, they were not compatible with DirectX 11 (DX11). DX11 is a system development kit with computer gaming modules for windows based computers. Therefore, our selection of a graphics card would need 4 computer monitor ports and be DX11 capable. We selected the EVGA GeForce 970 because it was the most economical graphics card to meet our requirements.

We would also need to select a NI data acquisition (DAQ) card. The legacy system used NI DAQ PCIe-6351 that was capable of 1.25 MS/s across 16 channels. There are five main types of eye movements where saccadic (side to side eye movements) are the fastest. Frequency analysis has shown that these movements can be sampled at 200 Hz or greater to avoid aliasing where most researchers sample saccades at about 1000 Hz. Vergence is about an order of magnitude slower, hence we decided to purchase with more economical NI DAQ PCIe-6320 card. This card is still capable of 250kS/s, which is still much more than our required sample rate. We required only 7 analog ports (six for digitalization of analog data and one to determine when the subject has pressed the trigger button. Most DAQ cards come with capabilities for 16. A BNC 2090A breakout box was used to connect our BNC cable inputs to the DAQ card.

Beyond specific requirements of the DAQ card, graphics cards, and motherboard, we chose specifications that would round out our high end computer. Including 16GB RAM, and a Quad Core 3.50GHZ microprocessor.

2.6 VisualEyes LabView System

We needed a software platform that allowed for easy prototyping and data acquisition, but versatile enough to do all of the required tasks. For this, we chose National Instrument’s LabView. LabView is a graphical programming language where symbolic blocks and wires replace lines of scripted text code. Our DAQ cards were also made my National Instruments, and synergy between hardware and software made the data acquisition portion of our code relatively straightforward. Later, this ease of use and LabView’s status as a non-compiled code would cost us dearly in terms of the speed that we needed.

The LabView program, named VisualEyes 2020 (VE), has several modules that report to one main controlling module. VE is designed to utilize a simple scripting language and file types that allows for maximum flexibility for a wide variety of experiments. A user can define a VisualEyes (a .vei file) (VEI) that is a tab delimited text matrix of what image a stimulus should use, where it is in terms of the X and Y pixel values of the screen, how many degrees of rotation the image will spin, and its size relative to its own native resolution all as function of time. These individual scripts can be indexed and called in VE’s draw configuration indexes. A VisualEyes Script (VES) can call these stimuli to preform trials on the subject and much more.

Table 2.1 Draw VEI Format

VisualEyes 2020 Draw VEI Layout Template

Variable	time	X Location	Y Location	Rotation	Stretch X	Stretch Y
Unit	s	pix	pix	degree	%	%

Note: If Stretch Y is blank, it will be the same value of Stretch X

VisualEyes is also able to control a light emitting diode (LED) array through the use of a serial port. This can also be controlled through VE similarly to how it can control graphical stimuli. The VNEL LED sticks, one MRI safe and the other not, can be placed in the center of the Haploscope in lieu of screen base stimuli. By controlling which LED light is on at a given time, the target can be moved farther or closer away from the subject. However, the limitation of this LED stick is that the target must always appear on midline. This will not be a limitation of our video monitor based system.

Table 2.2 Array VEI Format

VisualEyes 2020 Array VEI Layout Template		
Variable	time	Boolean Location
unit	second	integer 0-512

VisualEyes is a script bases language where the user can create their experiment by selecting an operation from a list of commands. This simple design allows for rapid training of new lab members and allows for the creation of experiments while requiring almost no LabView ability. VE primary ability is to conduct experiments by showing predetermined visual stimuli from the VEI and recording data. To support this function, the program has the ability to wait on user input and play sounds to cue the user at the start and end of a trial. Data can be written to a hard drive on command by another command. Entire sections of code can be easily repeated with another function. Another function creates and displays a calibration board that is a function of the subject's inter-pupillary distance. To add randomness to an experiment, the system can pause for a random amount of time within a range of values supplied.

2.7 The VisualEyes Experimentation Kit

Upon joining the lab, a new member is given a copy of VisualEyes from the master copy. The master copy is version controlled so that any changes or updates can filter out into the lab from the central source. The hope is that a new lab member is given the majority of the things that they would need to test their hypothesis. By creating a simple complete package to use, our lab members can spend more time experimenting and less time duplicating efforts. This kit comes with a bank of premade visual stimuli including lines of different colors, crosses of different colors, and blurry stimuli that follow a distribution of Gaussian. Simple visual stimuli are also created for the user. Premade calibration stimuli, steps, and ramps are all found within the VEI directory. Demonstration scripts with comments on how to build their own experiment come in the VES directory. Of course, a manual and quick guide sheet is included for easy training and reference.

Included within the VisualEyes2020 kit are the legacy versions of VisualEyes that do not require advanced graphics handling. Finally, a development system is included that allow a person to place a stimulus on the screen, nicknamed ManualEyes.

2.8 Calibration of Instrument

Calibration of our instrument is critical to the overall success or failure of the experiments that will be performed using it. If our instrument is out of calibration, we can easily invalidate any experimental trials preformed on it; a costly mistake that can undermine the credibility of the lab. Our instrumentation is only able to record voltage values generated from our vision tracking software. Without a way to change these voltages into a meaningful ocular measurement, the data becomes worthless. To calibrate the vision tracker, the ISCAN infrared video based system targets are presented to the subject on the computer screen that are adjusted until the image on the Haploscope is superimposed on real world target that are located at measured known distances from the eye. By placing the target over the known target, we are able to the X and Y coordinates of the target on the screen.

There are two calibration board types; the first is a static calibration board that is set up for an average interpupillary distance (IPD) of 6cm. The colored lines represent rotation of the eyes. The lines are equally spaced outward from the central black line that represents a continuation of the midline of the Haploscope. The location of each pair of lines represents the continuation of gaze from each eye past the point in intersection along the mid-sagittal plane of the visual system if a person is correctly centered within the instrument. It is important to remember that the right eye’s lines are on the left side of the calibration board and vice versa. When the distances from the subject to the calibration board are known and that simulated target distance is given as a variable, it is possible to calculate how far a given line is from the midline to represent a target at the given stimulus.

$$\mathrm{D}_{CalibrationLineFromMidline} = [D_{CalBoard} - D_{T\arg et}] * \arctan(\theta)$$

Figure 2.10 Calibration board line spacing equation.

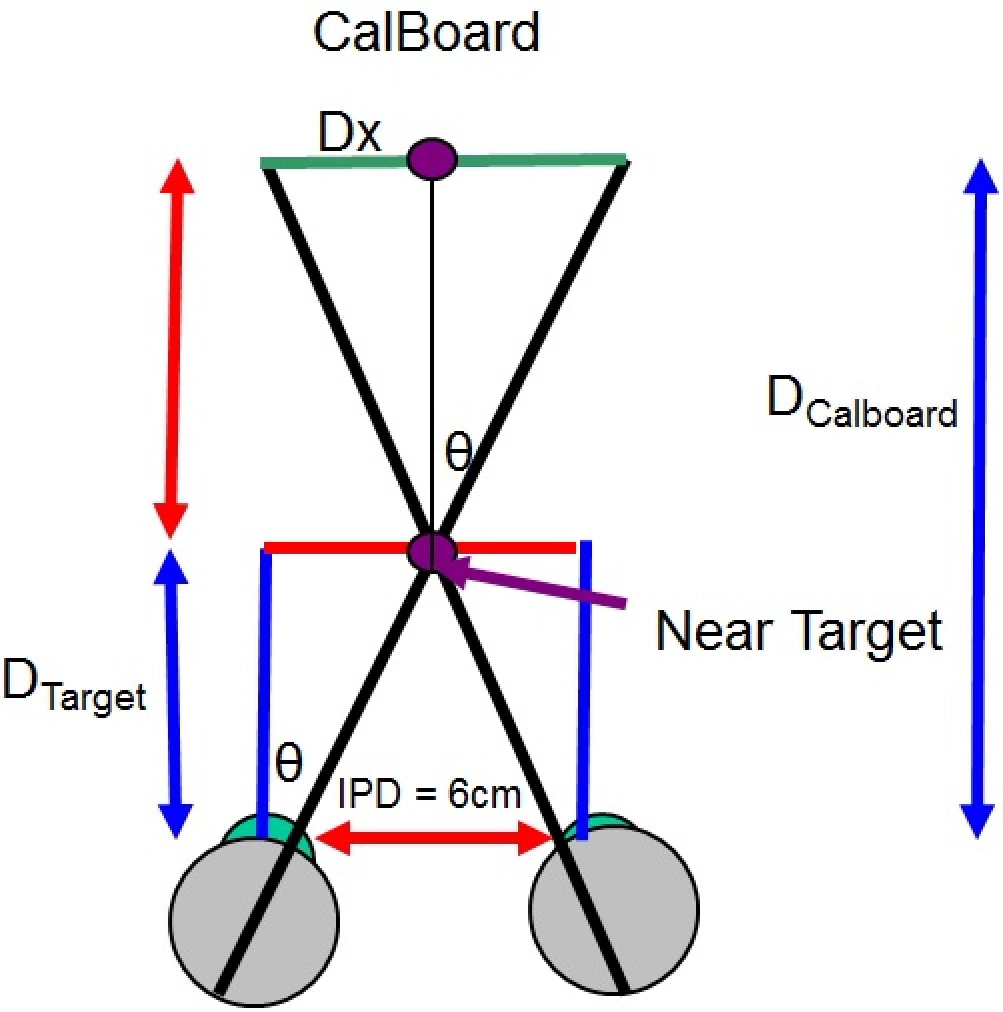


Figure 2.11 Haploscope calibration diagram (assuming IPD of 6 cm).

This method has been used for many decades in the Oculomotor Laboratory that Dr. Alvarez's trained in under the direction of Dr. John Semmlow. The procedure assumes an average IPD. Recently, reviewers have begun to question whether an average IPD is valid. Hence, an electronic calibration board was developed as a function of each person's IPD using the same LabView software that controls the rest of VisualEyes 2020. A person's IPD can be measured quantitatively with the Plus Optix instrumentation. As the program initializes, it will create this calibration board using the same mathematical calculations as shown in the static calibration board. The calibration board can be hidden or shown either programmatically or by a button on the controller screen.

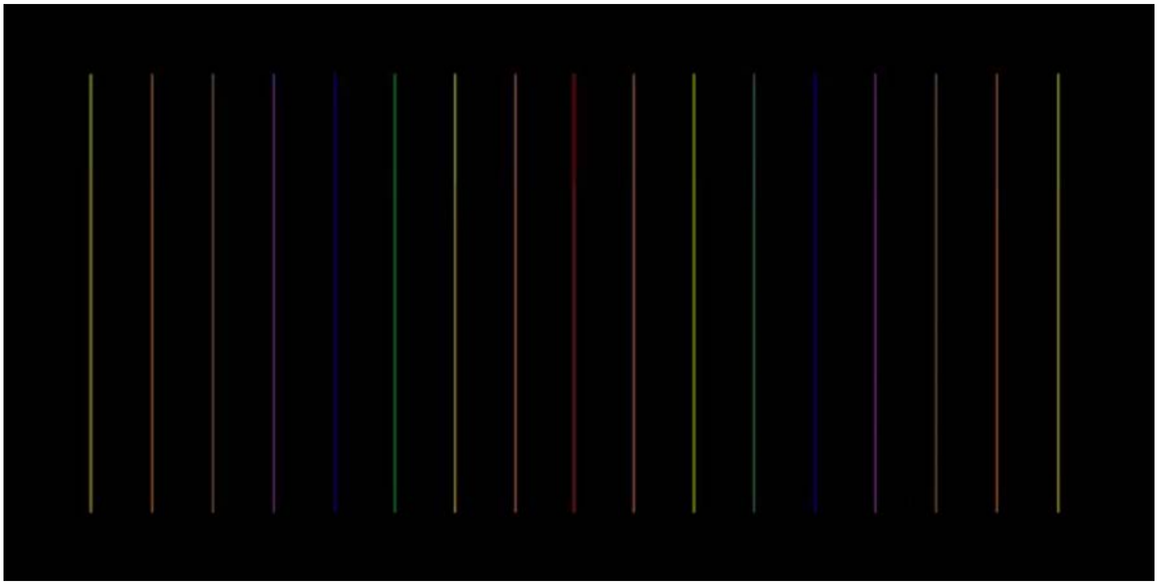


Figure 2.12 Electronic calibration board. This board is shown on a monitor as has the ability to change between subjects depending on an individual's IPD.

Using either board (static analog or electronic which changes as a function of IPD), calibration must be conducted before starting an experimental trial. The goal of this calibration is to derive a linear relationship between the voltage values recorded from VisualEyes with the meaningful unit of ocular rotation. To do this, we record data while the subject is looking at a calibration line that causes a known position of ocular rotation (θ). By recording three of these angular rotations as a function of voltage measurements, we can derive a linear transformation. VNEL has adopted a procedure that requires the linear regression to have a correlation coefficient greater that R^2=.98 to be considered a valid calibration. Typical causes of calibration error are blinking during the measurement or head movement.

To do this easily, a calibration program was developed that is separate from the rest of VisualEyes, but comes included in the VisualEyes kit. This program will allow the user, via a wireless mouse, to place a stimulus over the known ocular rotation line and display the pixel values of that line. By getting at least three data points per screen, the linear relationship can be developed to show any angular degree of rotation by showing a target at a given location on the screen. This same relationship is used to transform experimental voltage values into degrees of ocular rotation.

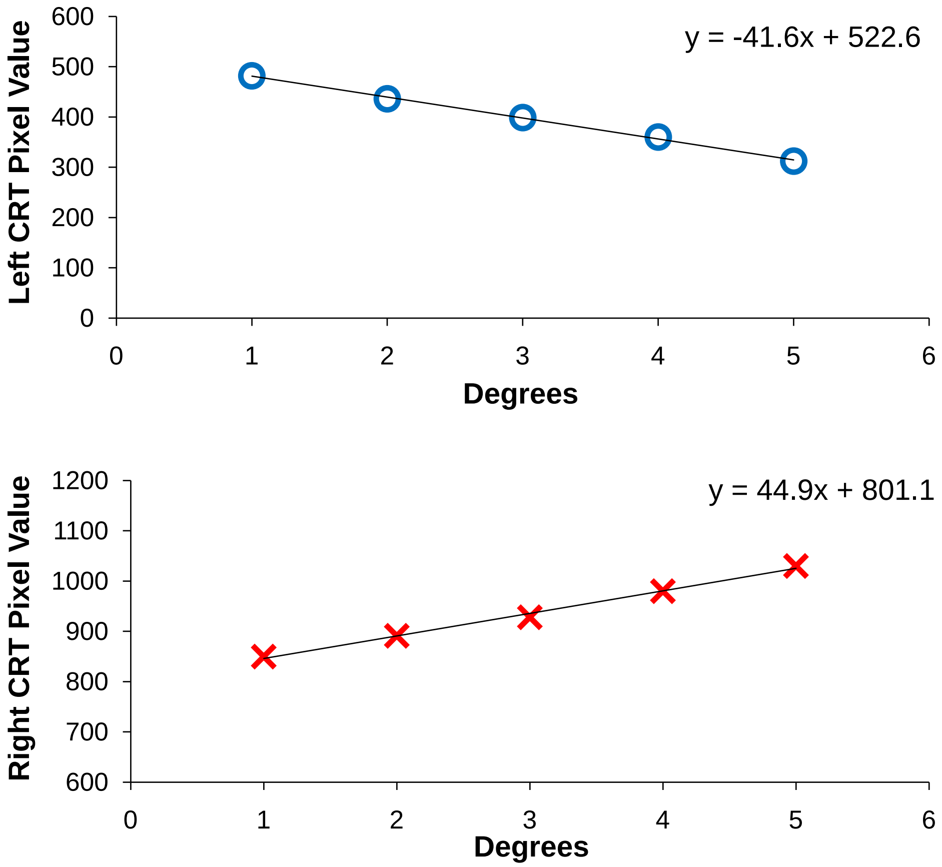


Figure 2.13 Pixels to degrees linear relationship. Experimenter chose to use 5 values to assess for linearity.

The above represents that calibration for the ISCAN eye tracker. The accommodometer is self-calibrating so long as it is kept 1 meter away from the subject.

While the manufacturer states that as long as the accommodometer is 1 m away then calibration is not necessary, one published paper by Blade and Candy [16] shows that calibration is necessary. Within the results section, we will compare our calibration with that published within the literature.

Blade and Candy [16] calibrated the accommodometer by adding lenses from in front of the human eye and measuring the amount of refraction from the lens. They found the slope could vary from 0.5 to 1.15 compared to the manufacturer's claim of a slope equal to 1. Hence, it is advised to calibrate per person. For each subject, this calibration curve was calculated by placing a known lens in front of one eye while recording data.

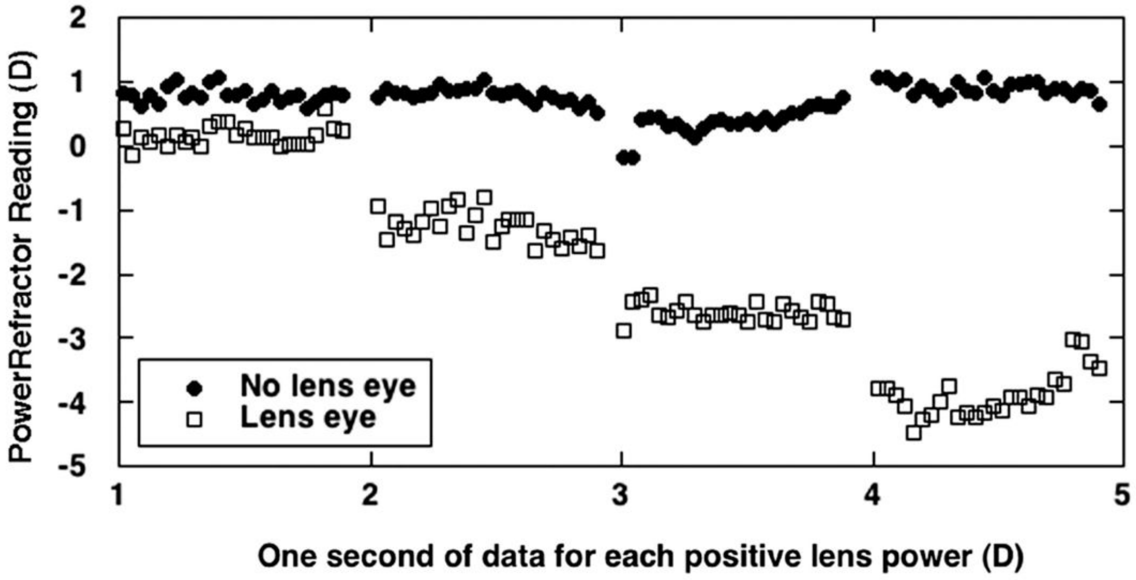


Figure 2.14 Blade and Candy's plot of their four lens powers. [16]

Taking the average of each of the four lens power groups allows us to create a plot of lens power vs lens power. From here, we can easily derive a linear transformation that allows us to go from the Plus Optix Power Refractor III's native curve to our customized subject dependent curves.

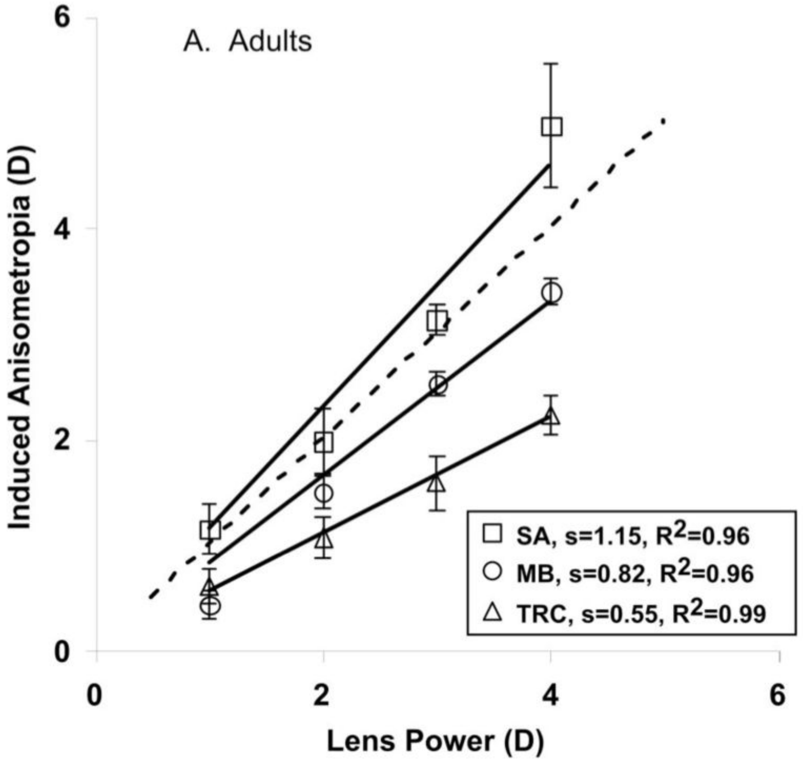


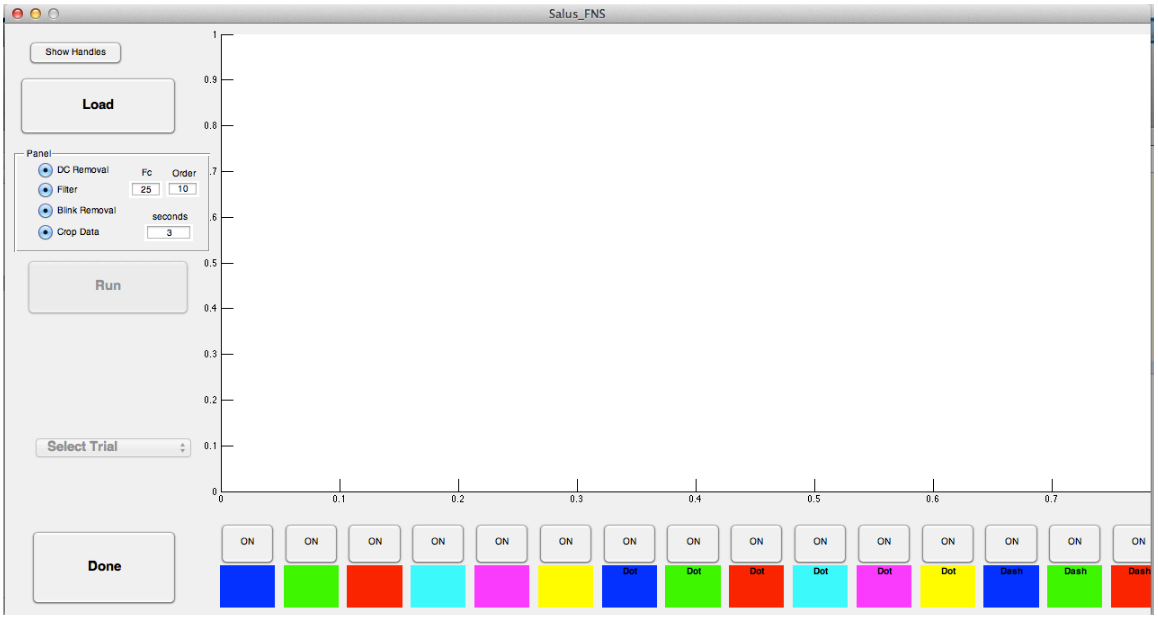
Figure 8 Accommodation calibration curves. Blade and Candy's Personalized calibration curves vs the assumed 1-to-1 calibration curve. [16]

The results section of this thesis will show that when use the same protocol described by Blade and Candy, our system reports analogous results.

2.9 Post-processing MATLAB GUI

Despite our best efforts in making the subject as comfortable as possible, there are still going to be trials where the subject fails to do the ocular movement for reasons other than their visual inability. Reasons for poor ocular movements include blinking during the trial, especially the transient portion of the movement, head motion during the visual task, or a subject just not attempting to follow the visual task. There exists a need to filter the good experimental trials from the bad. During this process, we can do other simple signal processing tasks.

MATLAB was chosen as the programming language to sort eye movement data. A MATLAB script requires a substantial investment in time and a basic level of programming ability that the labs clinical partners may or may not have. A MATLAB graphical user interface (GUI) for data processing was developed.



Fc 25

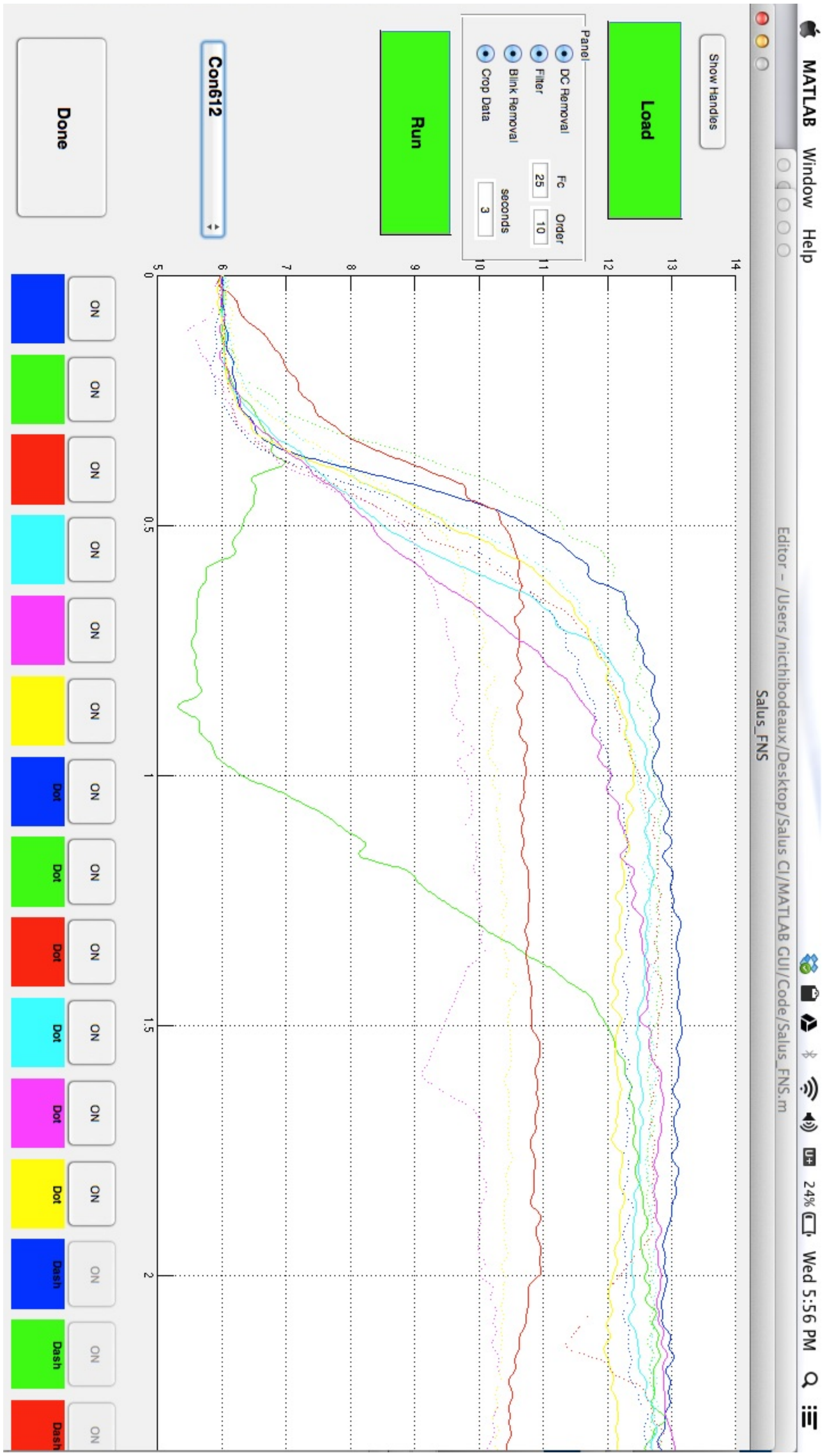
Order 10

seconds 3

Figure 2.16 MATLAB GUI interface upon initialization.

The user has limited options by design; more options leads to greater complexity. After processing, additional MATLAB code can be written for case specific analysis. The radio buttons on the GUI allow for DC filtering, Butterworth filtering, blink removal, and the option to crop the data at a given length. Calibration of the data by linear transformation of the voltage values into meaning degrees of ocular rotation is done by default. The DC filter eliminates any baseline drift that may occur during the experiment. Butterworth filtering removes any high frequency noise that may be introduced. Blink removal interpolates between areas where the signal saturates because ISCAN has lost the pupil signal and hence the DAQ card is digitized to +/- 5V. A blink within an ocular movement is grounds for the trial to be excluded from further analysis because the subject did not track the visual target.

After data is loaded by selecting the raw text file from VisualEyes and configuring the options given, run will display the data and enable the multi-color switches. The user can change their option settings and hit run again if they are unhappy with the original settings. The switches will control which of the similar trials are kept and which are discarded. By displaying them in this fashion, an operator can easily see a pattern and discard those that are bad. This gives the operator feedback as to whether the data collection experiment was a success. It also allows the operator to begin to define data inclusion and exclusion criteria. Once this process is completed for each ocular movement, the user can click done. A cleaned data packet will be exported that can be used for plotting or further data analysis.



Editor – /Users/nicthibodeaux/Desktop/Salus CI/MATLAB GUI/Code/Salus_FNS.m

Salus_FNS

Panel

● DC Removal

● Filter

● Blink Removal

● Crop Data

Fc 25

Order 10

seconds 3

Con612

Figure 2.17 MATLAB GUI in operation.

CHAPTER 3

RESULTS

Test experiments were designed and executed to assess the validity of the instrument. Before larger validation experimentation, a simple proof of concept was conducted. In regular day-to-day life, there is only one real distance that a target is presented at. When we look at the clock across the room, our gaze must intersect at that and we must focus our lens for the new length. In a Haploscope, we are not presenting actual targets, but simulated ones. Therefore, this relationship becomes uncoupled and we can present targets at different simulated distances to stimulate disparity vergence, but not change the actual distance away from the eye to keep accommodation. First, we tested each system independent of the other. We did this to ensure that any errors we found later would be due to simultaneous vergence and accommodation recording and not an individual system.

To prove our individual systems, we first placed a lens of known lens power over the eye and had the Plus Optix read through that lens. We can see that as we change the lens, a linear relationship develops showing that the change in lens power (x-axis) in the right experimental eye shows through on the Plus Optix output (y-axis). Also, the control left eye does not change from trial to trial as it does not get a test lens. This process was covered in greater detail in the calibration section.

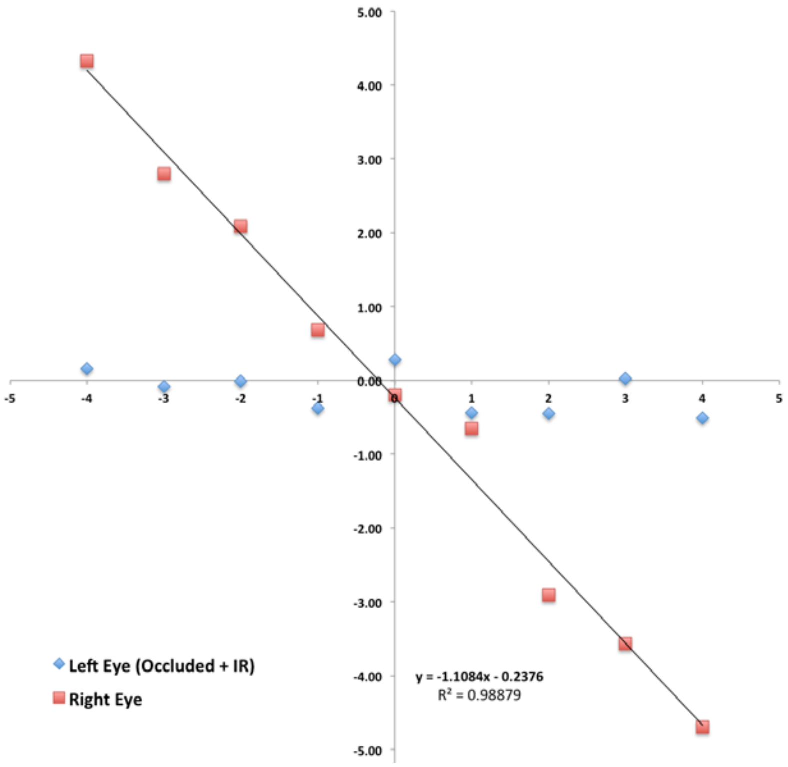


Figure 3.1 Accommodation validation.

Our vergence test was also done using simple steps. After processing, the movements and velocities are easily seen.

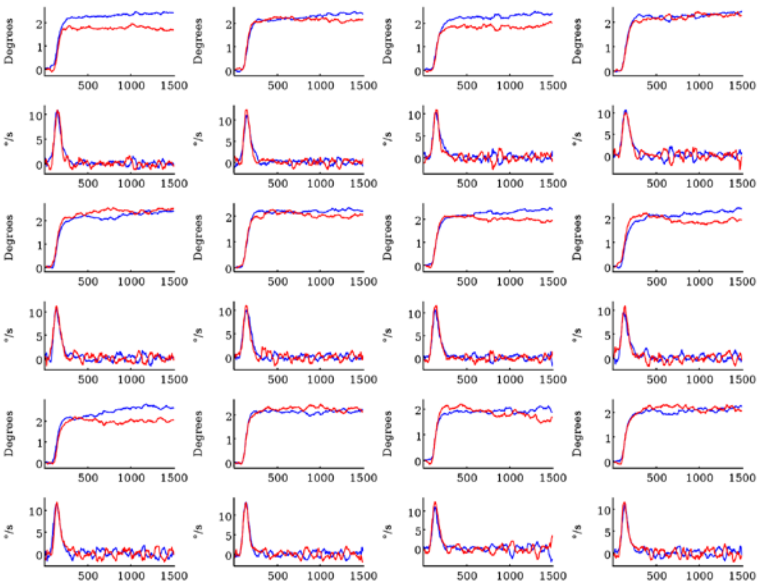


Figure 3.2 Vergence validation.

Second, we designed a movement where a vergence step movement that would cause the eyes to rotate, but not stimulate accommodation which was held constant. This was done by keeping the stimuli on the near pair of screens, to keep accommodation constant, and changing their ocular vergence angle.

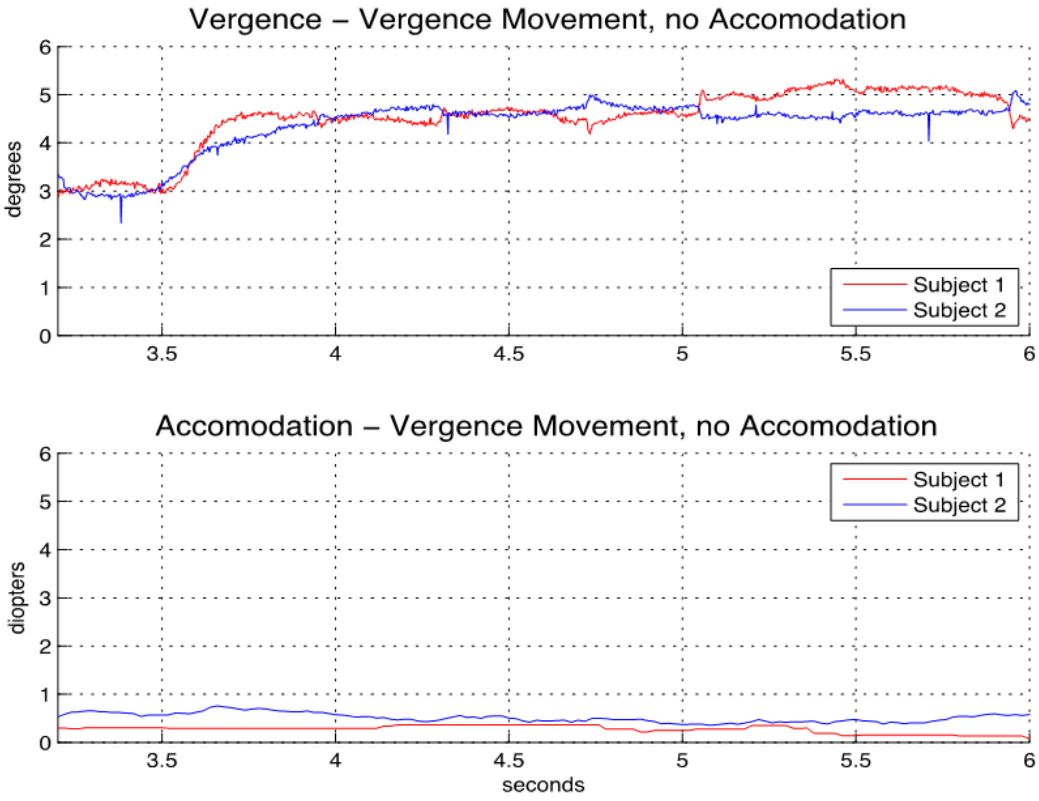


Figure 3.3 Vergence movement at one fixed distance. This is done by changing the simulated target distance, but not the physical distance. We can see a clear vergence movement, but without any substantial accommodation activity.

Similarly, we can show that the opposite is true. By switching screen sets, we can make a target at the same degree of ocular rotation, but only farther away.

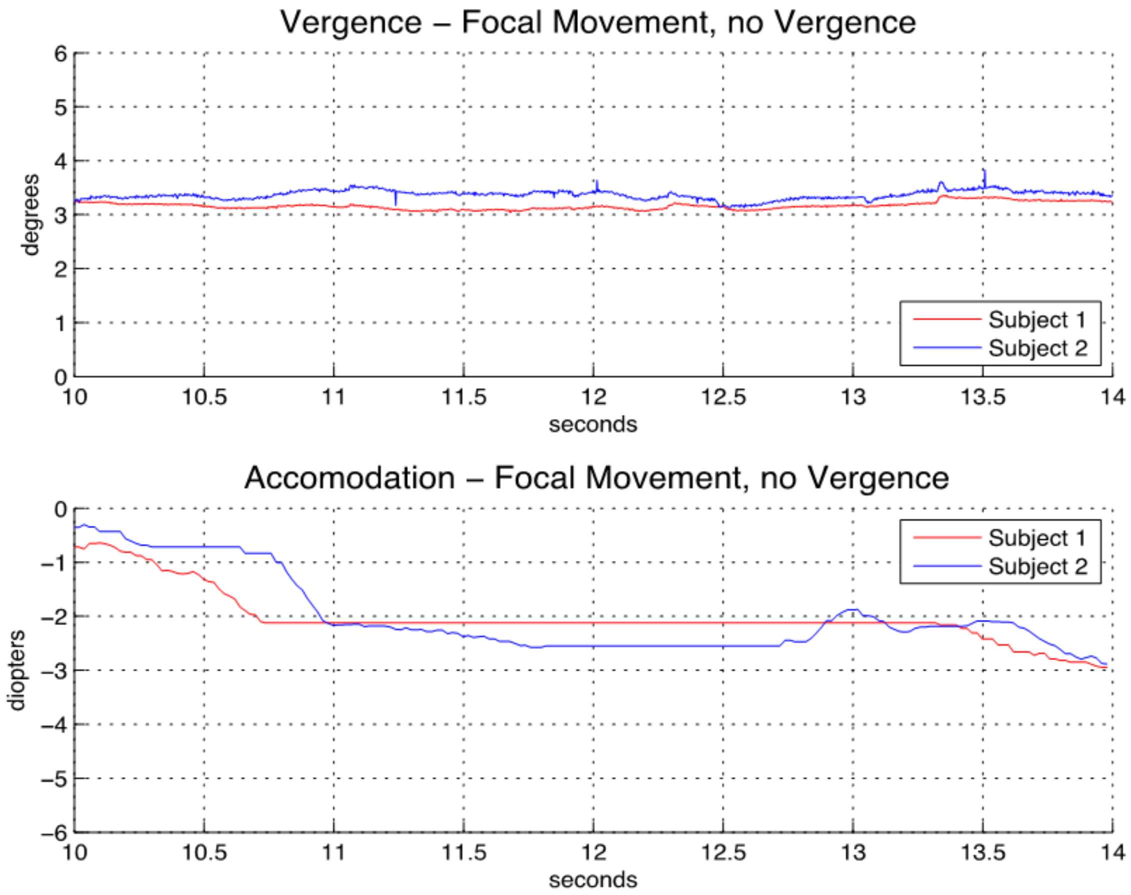


Figure 3.4 Accommodation movement due to changing distances. Changing physical distance, but keeping the target at the same ocular angle.

Since we have shown that each system works independently and together, we can now create a sample experiment and for a more rigorous test. The discussion section of this thesis, describes several experiments that are using the new experimentation. Each of these studies has subjects participate in multiple sessions to assess the precision of the system.

CHAPTER 4

DISCUSSION

This instrumentation was primarily designed to fit into the larger aims of a National Institute of Health study on Convergence Insufficiency (CI). The instrumentation is also being used within every active research study within VNEL. The discussion will discuss the new Haploscope's primary purpose for VNEL's NIH study followed by the active studies in the lab. Those studies are the following: vergence asymmetry potential due to color differences, influence of distracters on symmetrical vergence, investigation of symmetrical compared to asymmetrical phoria adaptation, monocular versus binocular control in vergence, visual stimuli to study the neural correlates of vergence, and investigation of vergence and saccades in children with CI before compared to after vision therapy.

Primary application, underlying mechanism of vision therapy

The etiology of CI is unknown. VNEL will use the instrumentation which was the focus of this engineering MS thesis to investigate two potential dysfunctions that may in part be the cause of CI. The two hypothesized dysfunctions in CIs compared to binocularly normal controls BNCs are the following: (1) reduced ability to adapt phoria in near and far space and (2) reduced ability to quickly diminish disparity error. Such dysfunctions may be remediated by vision therapy. Understanding (1) how CIs differ from BNCs and

(2) how changes in the brain-behavior correlate to changes in visual symptoms are significant because such knowledge can lead to targeted therapeutic interventions. Future targeted therapies could remediate symptoms sooner, further improve vision function, and achieve higher success rates ultimately reducing health care cost.

Schor describes two components of vergence. The fast component responds rapidly (within 1 second for a BNC) to reduce retinal disparity. The slow component maintains net fusional vergence over extended periods of time. As described below, the literature supports the fast and slow components may be dysfunctional in those with CI. This instrumentation will allow VNEL to take its first step towards understanding how vision therapy adapts the underlying neurophysiology to have a sustained effect on reducing CI patients' visual symptoms.

AC/A is the ratio of the accommodative convergence AC (in prism diopters, Δ) to the stimulus of accommodation A (in diopters, D). The accommodation (primarily stimulated through blur) and vergence (primarily stimulated through disparity) systems interact through cross links where the AC/A is the associated input of accommodation to the vergence system. CI patients have a low AC/A ratio compared to BNCs. Schor and Saladin hypothesize that CI patients have reduced ability to adapt vergence in visual space compared to BNCs. They hypothesize that the imbalance in adaptation between the accommodative and convergence systems (via the AC/A ratio) may be a major cause of CI, which is a dysfunction in the 'slow' adaptive component of vergence.

North studied 7 CIs reporting all had a decrease in the magnitude of phoria adaptation at near and far using 6 base-out (BO) and 6 base-in (BI) prisms, respectively, and a reduced rate of phoria adaptation at near using a 6BO prism. Adaptation improved with therapy in the 5 CI patients who experienced a reduction in symptoms. Rate of phoria adaptation is reported to be considerably slower in near space with a 6BO prism compared to other distances and 6BI prisms. Brautaset showed CIs improve their rate of phoria adaptation at near using a 6BO prism after vergence therapy studying 10 CIs. While prior results are encouraging, these studies had small sample sizes and was not masked. Hence, a detailed study of phoria adaptation in near and far space using convergent and divergent stimuli is warranted. Such knowledge is significant because if the slow vergence system is dysfunctional then future therapies should concentrate on improving a patient's ability to adapt vergence to different visual spaces (i.e., near and far space).

CIs tend to have a larger exophoria at near (40cm) compared to far (6m). Exophoria is the outward deviation of the eye when binocular fusion is disrupted (i.e., one eye is occluded while the other eye is fixating on a target). Scheiman and others hypothesize that CIs are symptomatic because of the excessive convergence needed to compensate for a larger exophoria at near compared to far. The true mechanism by which vision therapy leads to a reduction in symptoms is unknown. Clinicians hypothesize that vision therapy leads to a reduction of symptoms by increasing positive vergence amplitudes. Positive vergence amplitudes are typically reduced in CIs compared to BNCs and improve after vision therapy. Alvarez and others report the peak velocity (and hence the ability to reduce disparity error) of fast fusional disparity convergence is significantly reduced in CIs compared to BNCs and improves after vision therapy. Studying whether the fast vergence system is dysfunctional will allow us to determine whether a critical element in therapies should be to target improving the fast vergence system's ability to quickly reduce disparity error.

It is also unknown whether CI may be the result of one or several dysfunction(s). Schor hypothesizes that phoria (prism) adaptation reduces the load/stress on the fast vergence system. VNEL has published that phoria adaptation influences fast vergence peak velocity and hence the ability to quickly reduce disparity error in the fast vergence system. Thus, results from both independent labs suggest phoria offloads work (i.e., reduces stress) of the fast vergence system. If phoria adaptation is dysfunctional, then additional stress is placed on the fast vergence system. Perhaps some CIs have a reduced ability to adapt their phoria in near or far space, while others have a reduced ability to quickly decrease disparity error at all distances. Perhaps the reduced ability to adapt phoria adds stress to the fast vergence system and then both become dysfunctional. Knowledge about which parameters vergence therapy should concentrate on improving, can lead to more successful and targeted therapeutic interventions - ultimately reducing health care cost.

Symptoms from CI Patients and their Negative Impact on Activities of Daily Living: Symptoms for CIs include: blurred vision, double vision (diplopia), eye strain, loss of concentration, frequent loss of place causing a rereading of text, reading slowly, print moving on the page, difficulty remembering what was read, sleepiness and headaches. These symptoms negatively impact an individual's daily activities such as schoolwork and employment. 89% of CI patients successfully treated via vision therapy were asymptomatic one year later. CITT designed/validated the CI Symptom Survey (CISS) that has a sensitivity of 98% and specificity of 87% in young adults.

CI has a reported prevalence of 4.2% to 7.7% in the general population. [17] [18] [19] Hence, using the 2010 Census, 13.8 to 24.1 million people suffer from CI in the US. Although it is unknown whether CI patients with and without brain injury share the same etiology, the therapeutic interventions are similar. The prevalence rates of CI after traumatic brain injury (TBI) range from 23% to 43% in civilian and 28% to 46% in veteran populations. 35% of stroke patients are reported to have CI. Studying acquired brain injury is beyond the scope of our current proposal. Our retrospective analysis of 557 TBI patients shows that, while 23% of the TBI population will have CI, only 9% will have CI without other visual or vestibular dysfunction(s). Thus, many clinical sites would be needed. However, the techniques we develop in this project have broader impact because they could be adapted to study future brain injury populations.

Vision therapy developed from the CITT will be used by VNEL because it is the only validated vergence therapy via a RCT. Vision therapy has a 73% success rate in children with a large Cohen effect size (>0.8) and was significantly more effective than CITT-placebo therapy which had a small Cohen effect size (<0.2).[20] However, the mechanisms by which vision therapy translates into a reduction of symptoms was not part of their study. [21] The results obtained from our project can explain which mechanism (improvement in ability to adapt vergence or reduce retinal disparity) is more correlated to the reduction of visual symptoms. This knowledge can lead to targeted therapies while further improving visual function in potentially less time. Expected higher success rates will lead to a reduction in health care cost.

Vision therapy is a type of physical therapy. By enhancing the capabilities of the vision system, the hope is that the subject's vision system will be able to adapt and do what it was once unable to accomplish. [8] A similar principle works for amblyopia. The gold standard for amblyopia is to cover the stronger eye with a patch and force the vision system to rely on the weaker eye. [9]

There are multiple issues with these treatments, not because they do not work, but because people are unwilling to comply with the intrusive tasks. Patching an eye removes binocular vision and can cause anxiety, especially in children. [9] It has been shown that office based vision training is more effective than training in the home, but compliance suffers and subjects are unwilling to consistently go the optometrist's office for several weeks. Office based training is also more expensive.

The instrument that was constructed is far too expensive to be used in common clinical practices; however, its true potential is in validating other CI treatment procedures within a research setting. By comparing the results from our Haploscope at the onset of a treatment routine and at the end to show final results, we can quantify the level of improvement from our subjects.

Vergence asymmetry:

VisualEyes 2020 has been upgraded where it can now display numerous images. This was critical for this study because VNEL needed multiple images which were identical except for a difference in color. Clinically, many practitioners use red and green visual stimuli. The different colors of light are presumed to penetrate different focal lengths to the back of the retinal. However, it is unknown whether such a difference in accommodative vergence would have an impact on vergence peak velocity.

Distracters interaction with symmetrical vergence:

The prior version data acquisition code was not capable is displaying more than two images per screen. Now many images can be displayed simultaneously. Another active study in VNEL uses multiple images where subjects are asked to track a ‘x’ target with and within the presence of a distracter ‘o’. While the influence of distracters in saccades is well studied, very little literature exists on the study of distracters in vergence. In everyday conditions distracters are present in the distance. This is especially true while driving where you may have a car going at a similar speed with your car or a car in the opposing lane moving in the opposite direction of your car. Your vergence system is constantly adjusting in these scenarios where you have distracters close or away from you. The new VisualEyes 2020 can systematically study the influence of distracters in vergence.

Symmetrical compared to asymmetrical phoria adaptation:

Clinically, researchers use a 6 base out or a 6 base in prism in front of typically the right eye to assess vergence phoria adaptation. When a prism is placed in front of only one eye the adaptation will be asymmetrical and hence also stimulate the version system. By comparison, a set of flipper prisms which are used binocularly with both eyes could be used. VNEL is proposing to use a binocular 3 base out flipper and a 3 base in flipper which would symmetrically adapt the near dissociated phoria. VNEL prefers to assess phoria adaptation using an eye tracker which will quantify eye movement rather than relying on the subject to give the correct answer. With VisualEyes 2020, this code was easily programmed within a day.

Monocular versus binocular control in vergence:

There is a debate in the literature which dates back to the 1800s between two German physiologists specifically Helmholtz and Hering. Hering believed the eyes acted as a single organ similar to how one guides a horse with reins. While Helmholtz believed the eyes were independent and binocular coordination was learned. Science favored Hering’s theories commonly referred to as Hering’s final common pathway. However, King and Zhou rekindled this binocular versus binocular debate with their publication in Nature suggesting that the saccade burst cells were monocular. VNEL has devised a set of unnatural stimuli using VisualEyes 2020 which may add more data to this debate. The visual stimuli take advantage of the system’s ability to send independent images to the left and right eye are not only independent in terms of the image but also can be moved at different times.

Visual stimuli to study the neural correlates of vergence

The NIH grant to VNEL also includes functional MRI (fMRI) studies where VisualEyes 2020 has been loaded onto a gaming laptop. Rather than multiple monitors, the fMRI applications use a single monitor that can be projected to an fMRI compatible screen. The VisualEyes 2020 code has been tested and fMRI experiments will begin within the next few months.

Vergence and saccades in children with CI before compared to after vision therapy:

Another project within VNEL is a collaboration with Salus University, where an exact copy of the instrumentation at VNEL is located at the Eye Institute in Philadelphia. The primary clinical collaborator for VNEL is Mitchell Scheiman, O.D. This MS thesis included making a third set-up which is completed and has been tested. Currently, Dr. Scheiman is collecting vergence and saccade responses from children ages 8 to 17 years of age who have CI. The children will have their eye movements recorded again after vision therapy to assess how vergence and saccade change after therapy.

The strength of the new Haploscope design with the VisualEyes 2020 software suite is its versatility for numerous experiments. This thesis serves as documentation for future researchers in VNEL to access to learn how it was constructed. In addition, while all features requested have been implemented and tested, it is unknown what features VNEL may need in the future. This thesis serves to document the VisualEyes 2020 code so that future researchers can modify the software if needed.

APPENDIX

VISUALEYES LABVIEW PROGRAMMING CODE, MATLAB GUI PROGRAMMING CODE, AND HAPLOSCOPE COMPONENT DIMENSIONS

The appendix contains three parts. Part A will document the VisualEyes2020 Labview program. Part B will document the MATLAB GUI program. Finally, Part C will document the physical dimensions of constructed items for the Haploscope Table.

A.1 LabView VisualEyes Code

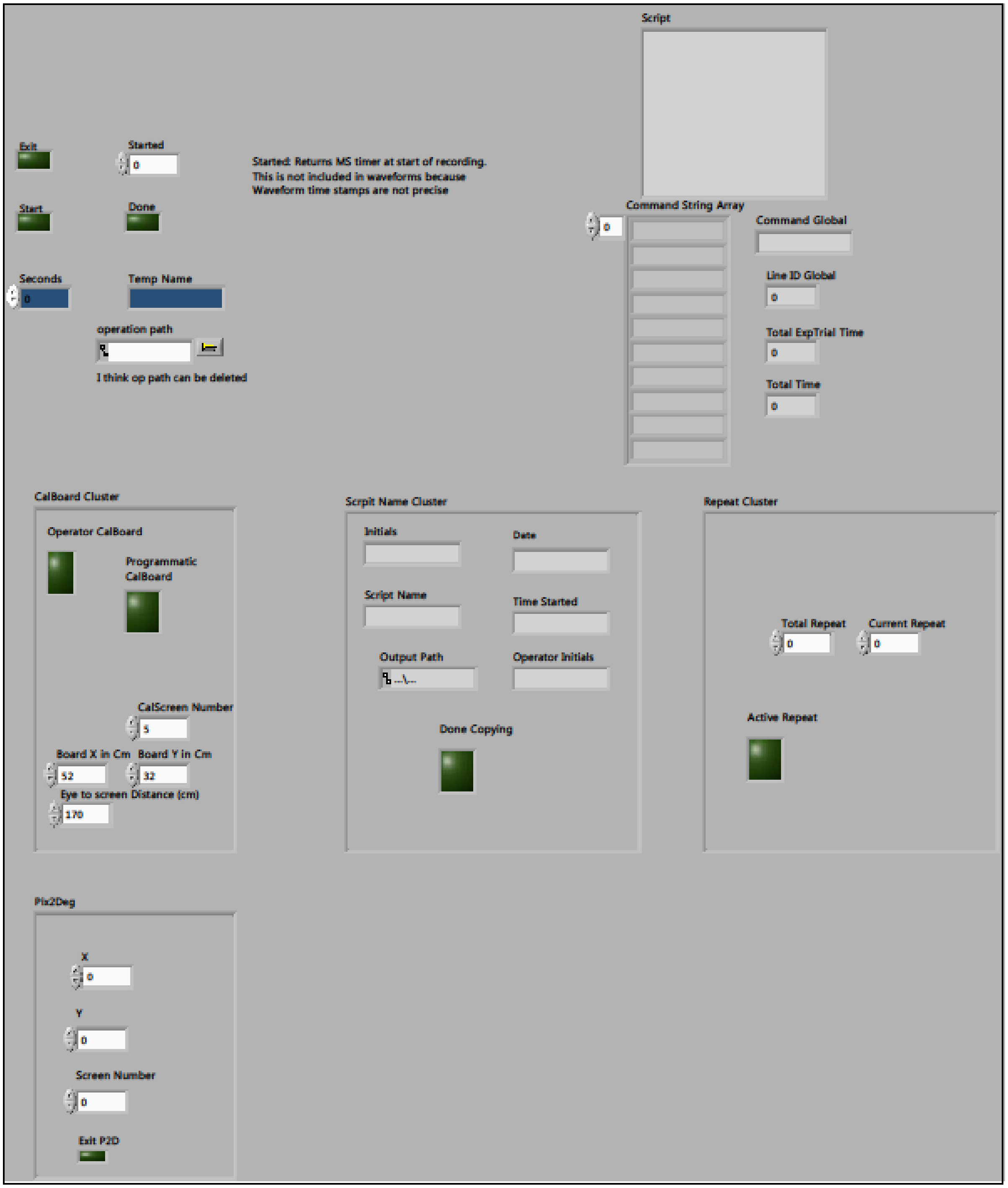
In Part A of the appendix, LabView front panels and block diagrams are shown.

aquire status.vi
C:\Users\Steve\Desktop\VisualEyes 2020\VI\aquire status.vi
Last modified on 8/20/2014 at 2:53 PM
Printed on 11/15/2014 at 3:49 PM

aquire status.vi

Page 2

aquire status.vi
C:\Users\Steve\Desktop\VisualEyes 2020\VI\aquire status.vi
Last modified on 8/20/2014 at 2:53 PM
Printed on 11/15/2014 at 3:49 PM



Started: 0

Started: Returns MS timer at start of recording.
This is not included in waveforms because
Waveform time stamps are not precise

Seconds: 0

I think op path can be deleted

Command String Array: 0

Line ID Global: 0

Total ExpTrial Time: 0

Total Time: 0

CalScreen Number: 5

Board X in Cm: 52

Board Y in Cm: 32

Eye to screen Distance (cm): 170

Output Path: ...\...

Total Repeat: 0

Current Repeat: 0

X: 0

Y: 0

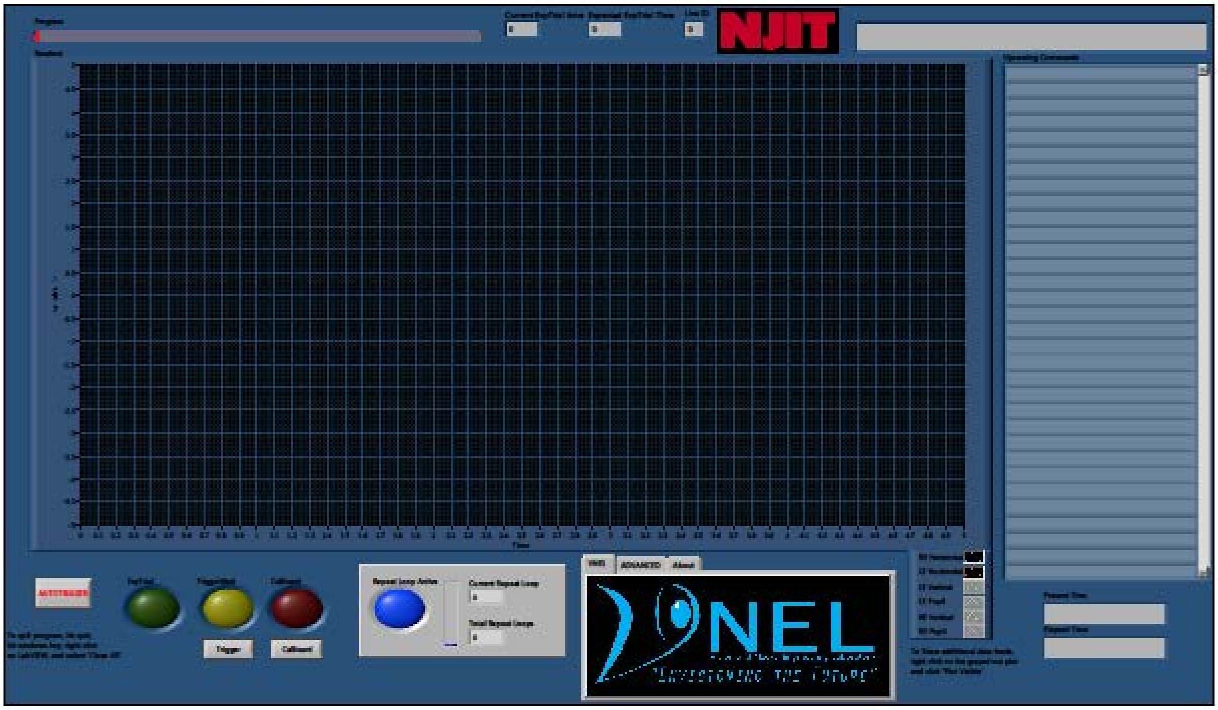
Screen Number: 0

Page 1

Aquire.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Aquire.vi
Last modified on 8/22/2014 at 12:25 PM
Printed on 11/15/2014 at 3:49 PM

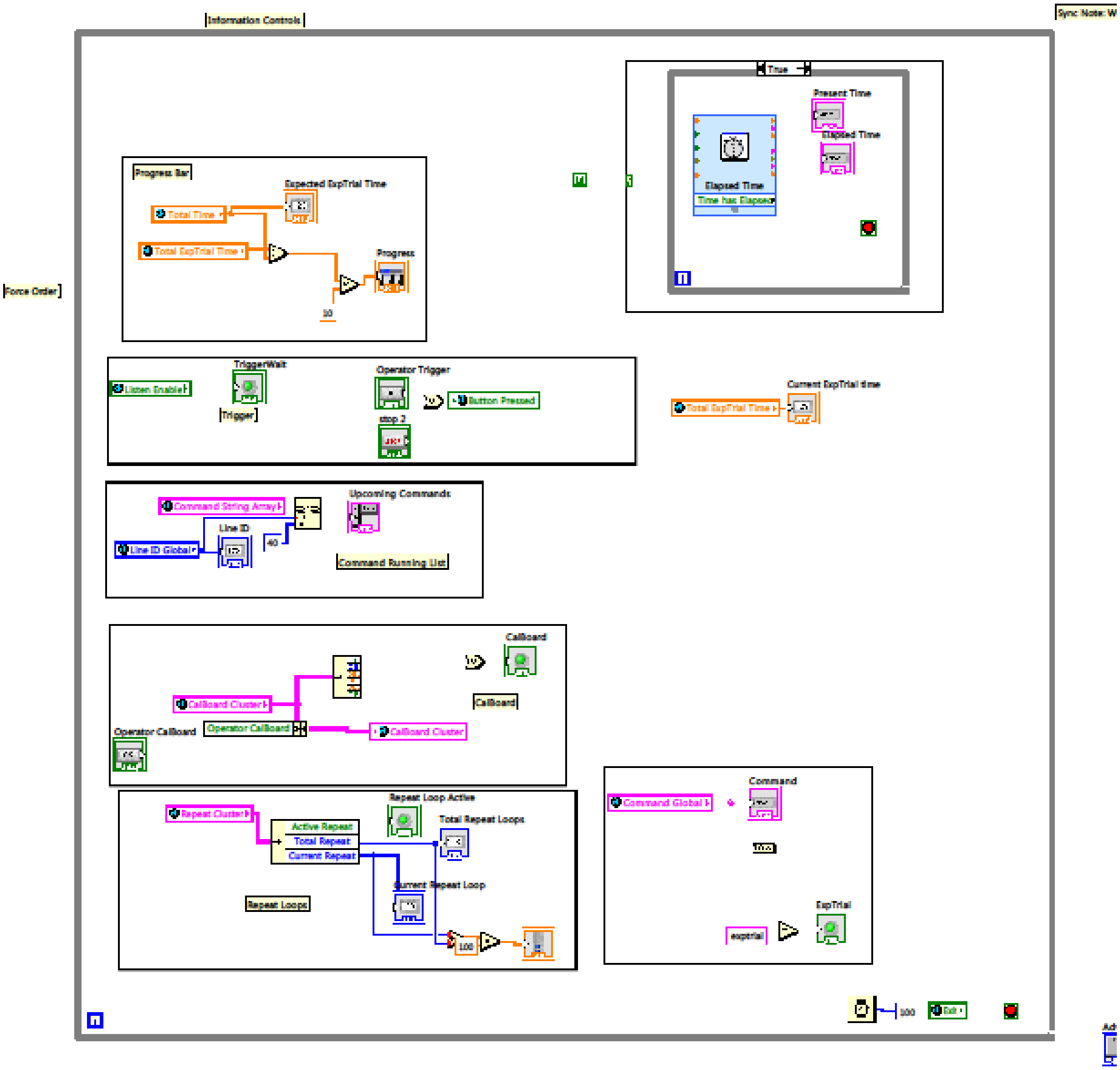
Aquire.vi

Config file

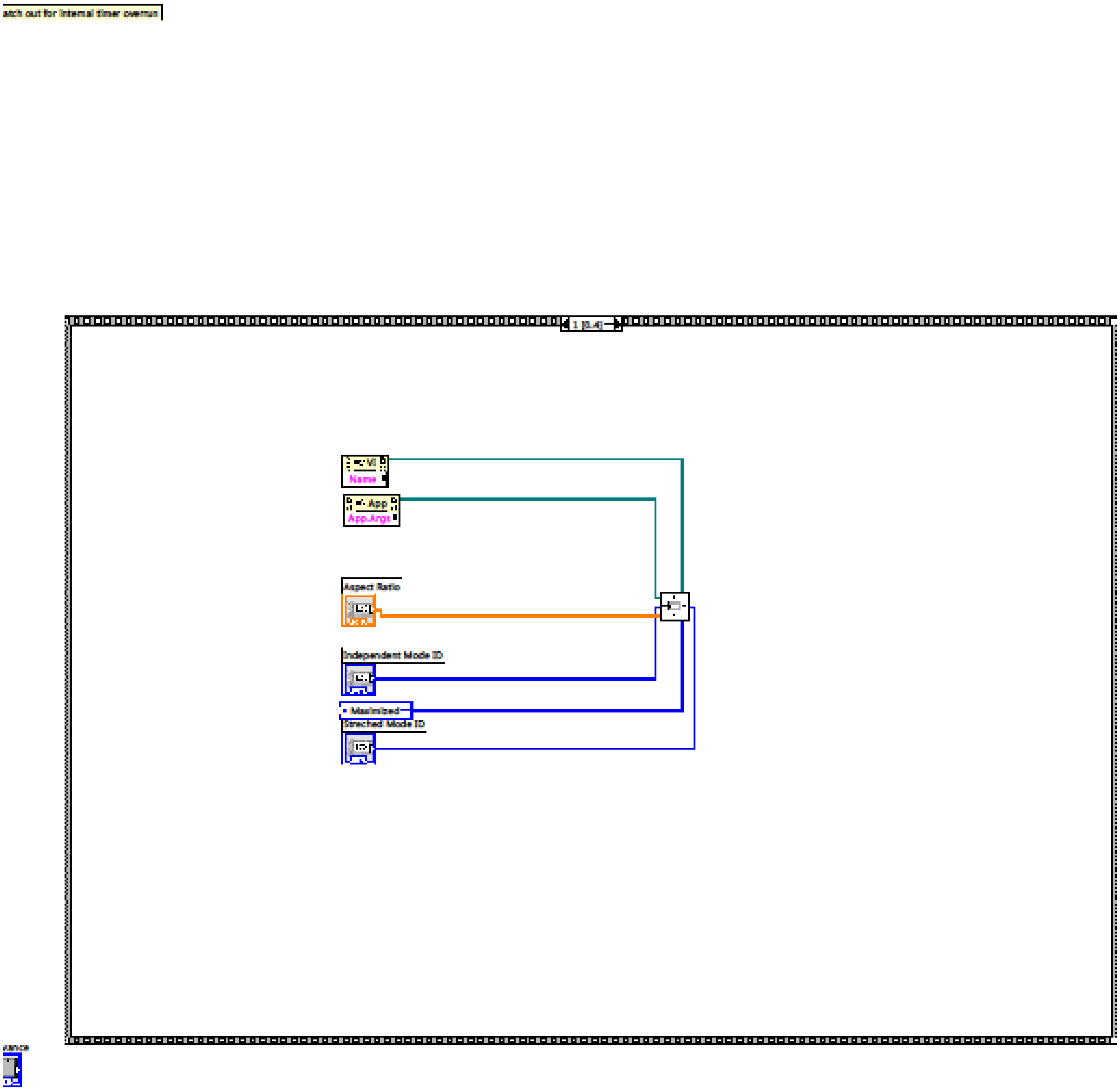


Aquire.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Aquire.vi
Last modified on 8/22/2014 at 12:25 PM
Printed on 11/15/2014 at 3:49 PM

Page 2

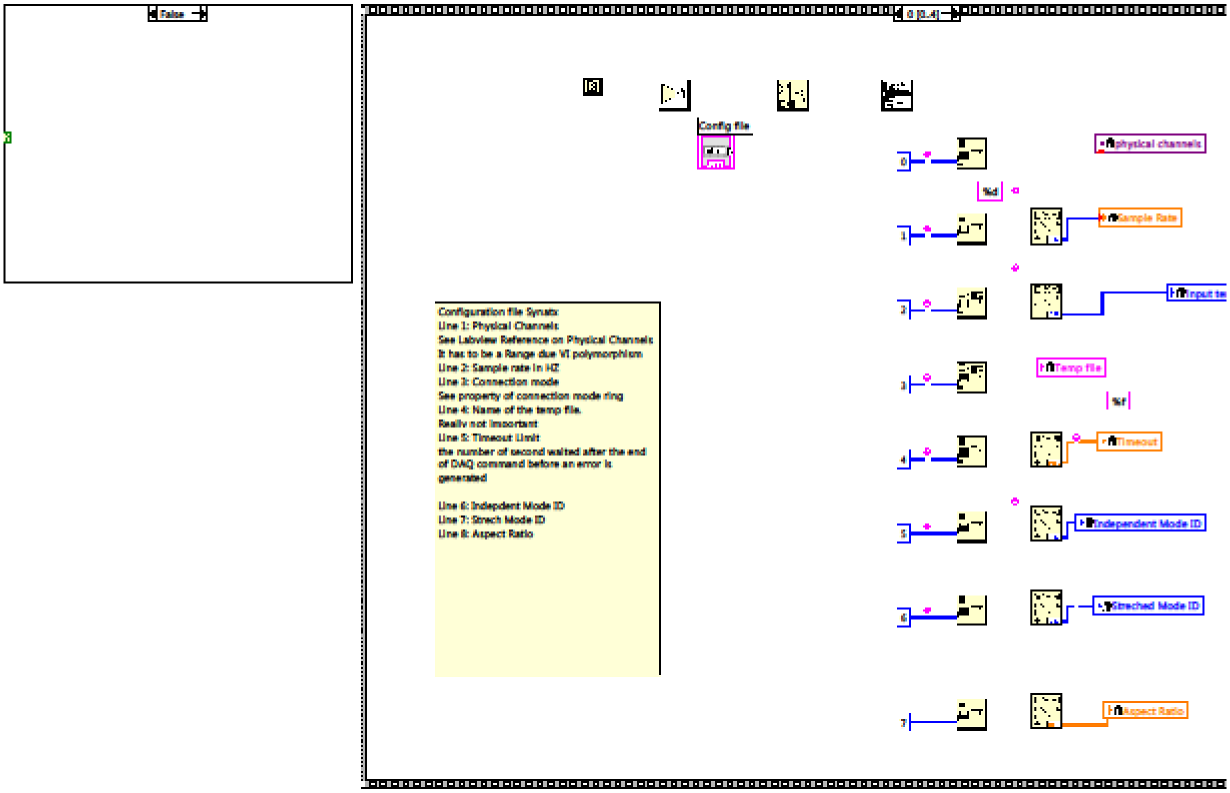


Aquire.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Aquire.vi
Last modified on 8/22/2014 at 12:25 PM
Printed on 11/15/2014 at 3:49 PM



Page 4

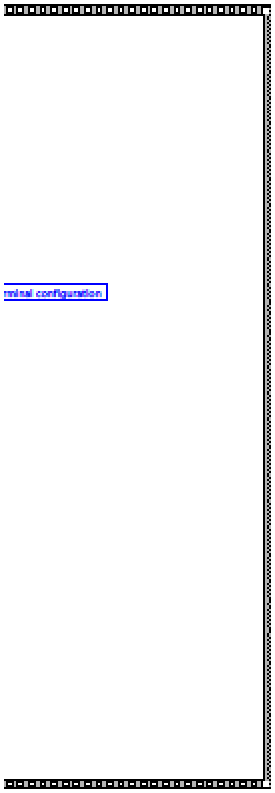
Aquire.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Aquire.vi
Last modified on 8/22/2014 at 12:25 PM
Printed on 11/15/2014 at 3:49 PM



Configuration file Synatx
Line 1: Physical Channels
See Labview Reference on Physical Channels
It has to be a Range due VI polymorphism
Line 2: Sample rate in HZ
Line 3: Connection mode
See property of connection mode ring
Line 4: Name of the temp file.
Really not important
Line 5: Timeout Limit
the number of second waited after the end of DAQ command before an error is generated

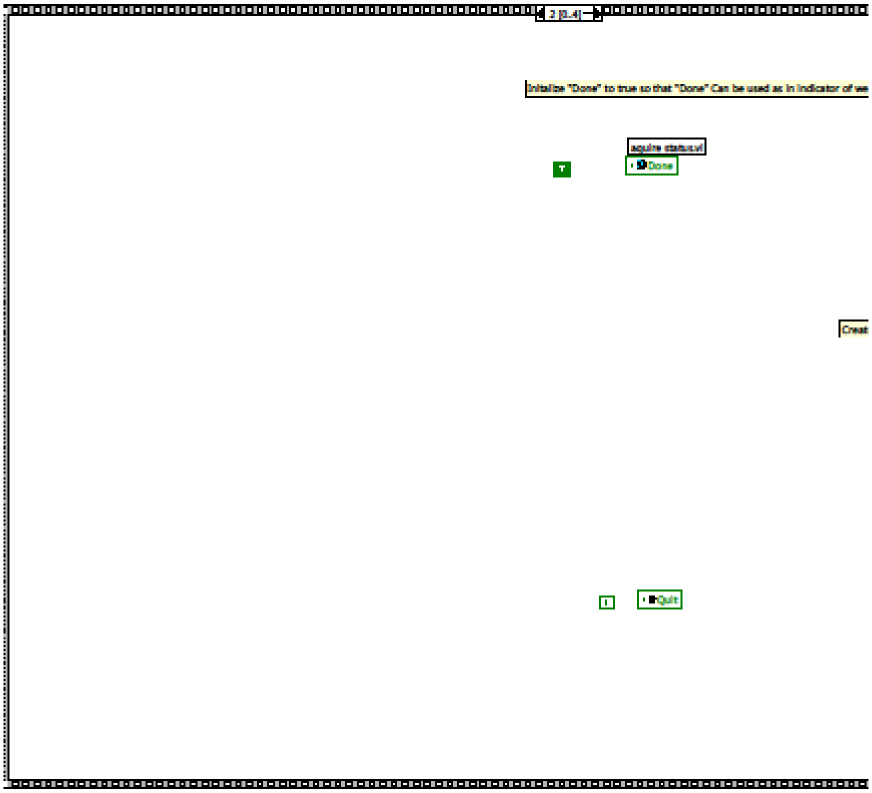
Line 6: Indepdent Mode ID
Line 7: Strech Mode ID
Line 8: Aspect Ratio

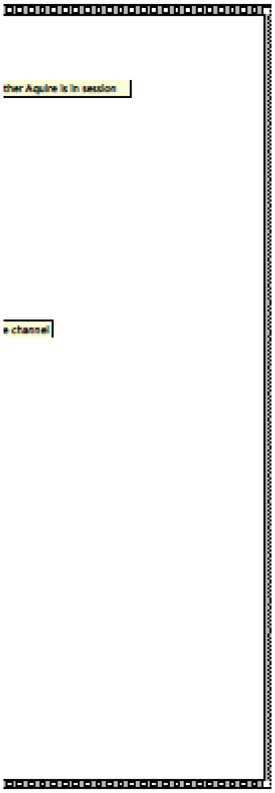
Aquire.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Aquire.vi
Last modified on 8/22/2014 at 12:25 PM
Printed on 11/15/2014 at 3:49 PM



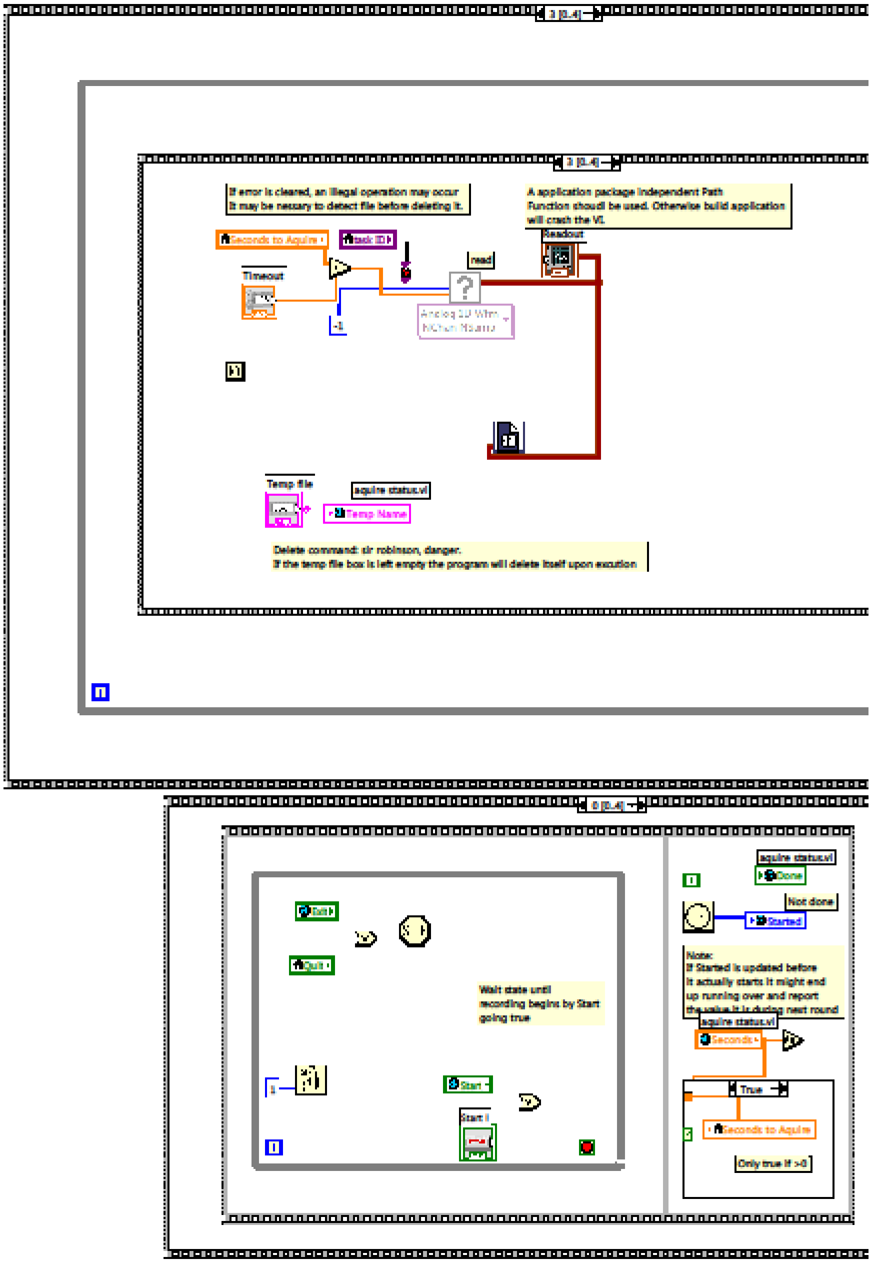
Page 6

Aquire.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Aquire.vi
Last modified on 8/22/2014 at 12:25 PM
Printed on 11/15/2014 at 3:49 PM


2 [0..4]
Initialize "Done" to true so that "Done" Can be used as In Indicator of we
T
aquire status.vi
Done
Creat
Quit


ther Aquire is in session
e channel

Aquire.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Aquire.vi
Last modified on 8/22/2014 at 12:25 PM
Printed on 11/15/2014 at 3:49 PM



If error is cleared, an illegal operation may occur
It may be nessary to detect file before deleting it.

A application package Independent Path
Function shoudl be used. Otherwise build application
will crash the VI.

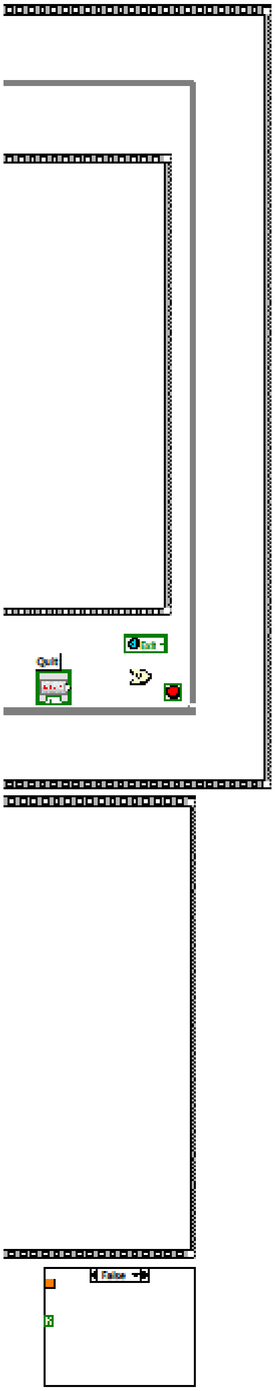
Delete command: dr robinson, danger.
If the temp file box is left empty the program will delete itself upon excution

Wait state until
recording begins by Start
going true

Note:
If Started is updated before
it actually starts it might end
up running over and report
the value it is during next round

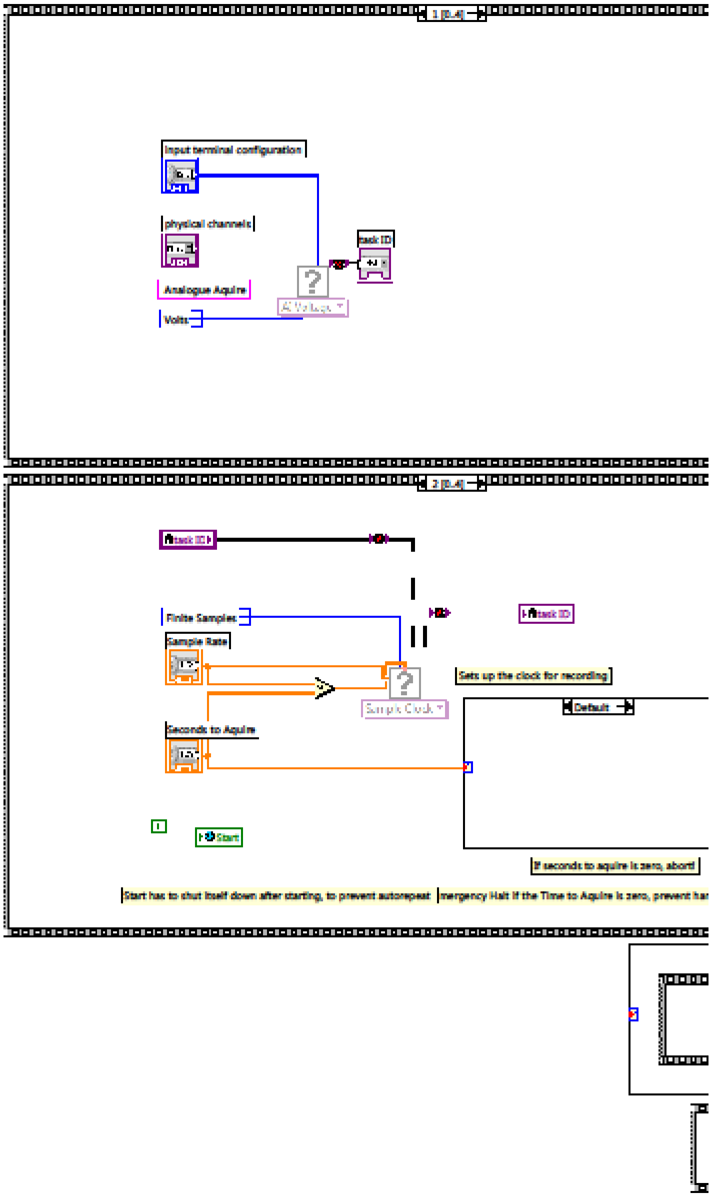
Page 9

Aquire.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Aquire.vi
Last modified on 8/22/2014 at 12:25 PM
Printed on 11/15/2014 at 3:49 PM



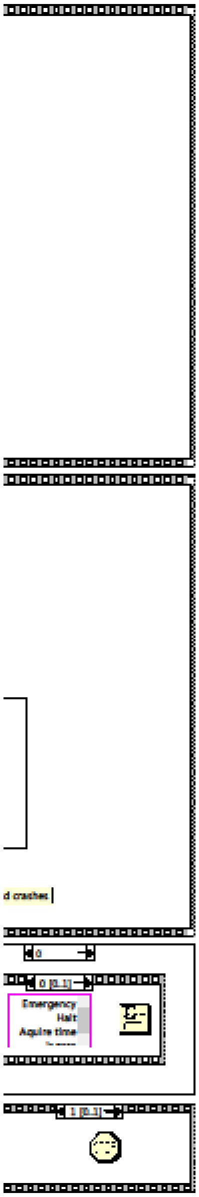
Page 10

Aquire.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Aquire.vi
Last modified on 8/22/2014 at 12:25 PM
Printed on 11/15/2014 at 3:49 PM


1 [0..4]
Input terminal configuration
physical channels
task ID
Analogue Aquire
AI Voltage
Volts
2 [0..4]
task ID
Finite Samples
task ID
Sample Rate
Sample Clock
Sets up the clock for recording
Default
Seconds to Aquire
Start
If seconds to aquire is zero, abort!
Start has to shut itself down after starting, to prevent autorepeat
mergency Halt If the Time to Aquire is zero, prevent ha

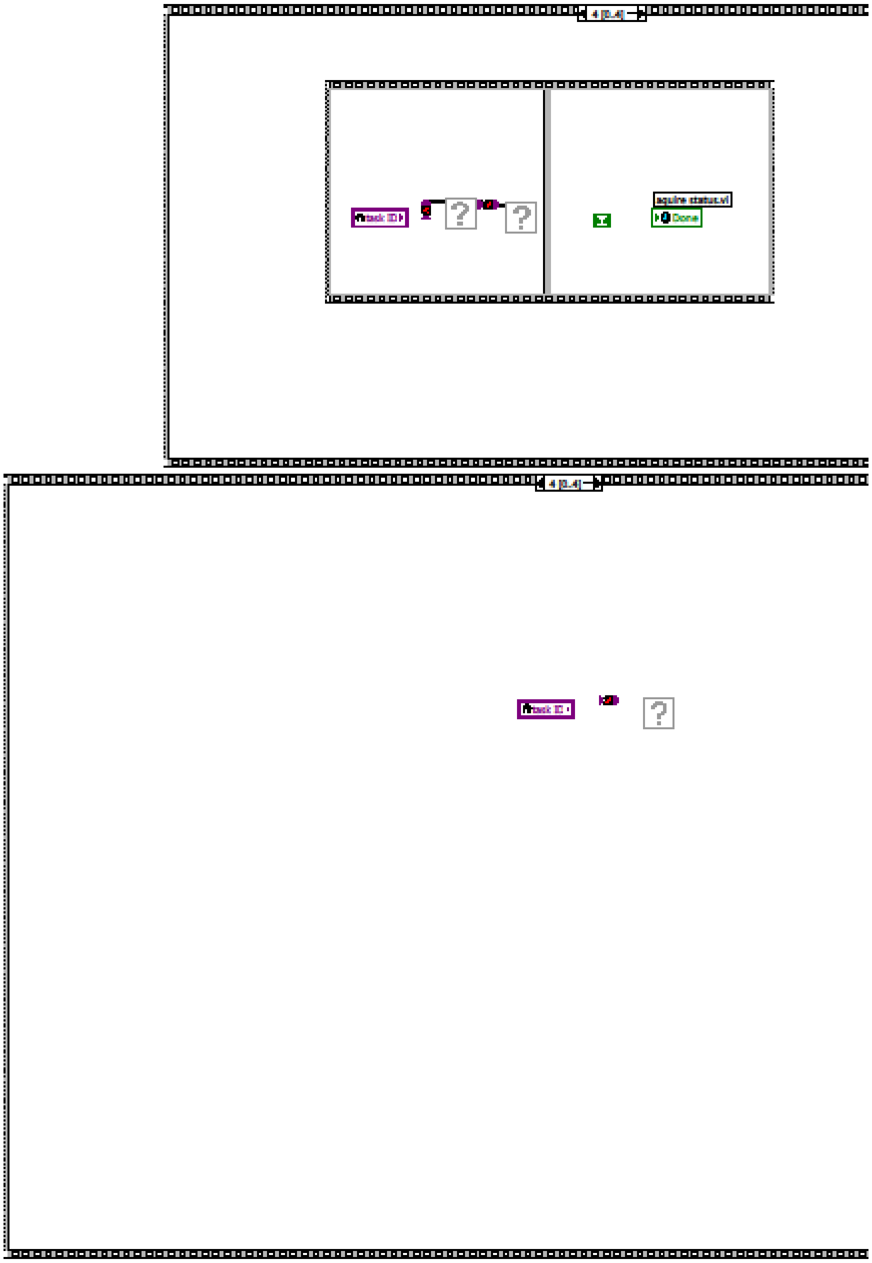
Page 11

Aquire.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Aquire.vi
Last modified on 8/22/2014 at 12:25 PM
Printed on 11/15/2014 at 3:49 PM

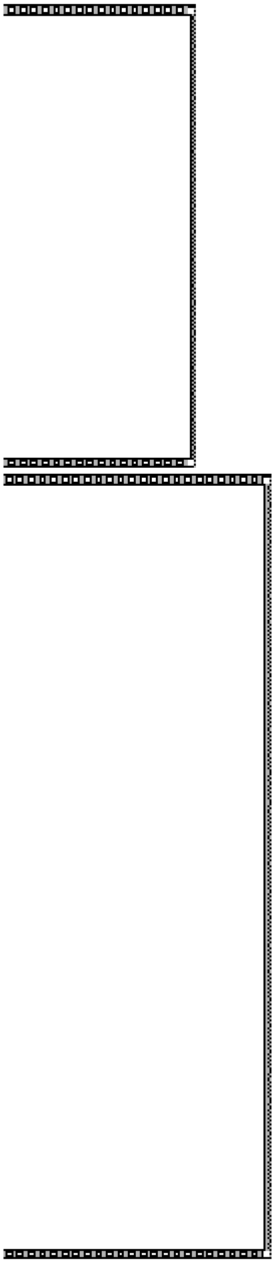


Page 12

Aquire.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Aquire.vi
Last modified on 8/22/2014 at 12:25 PM
Printed on 11/15/2014 at 3:49 PM



Aquire.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Aquire.vi
Last modified on 8/22/2014 at 12:25 PM
Printed on 11/15/2014 at 3:49 PM

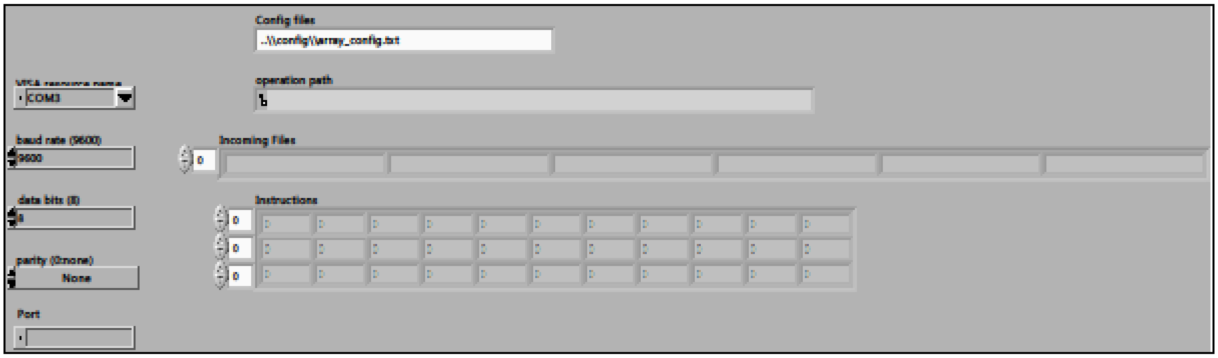


Page 1

Array IO.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Array IO.vi
Last modified on 8/14/2014 at 4:57 PM
Printed on 11/15/2014 at 3:50 PM

Array IO.vi

Config files



Config files: ..\\config\\array_config.txt

VISA resource name: COM3

operation path: %

baud rate (9600): 9600

Incoming Files: 0

data bits (8): 8

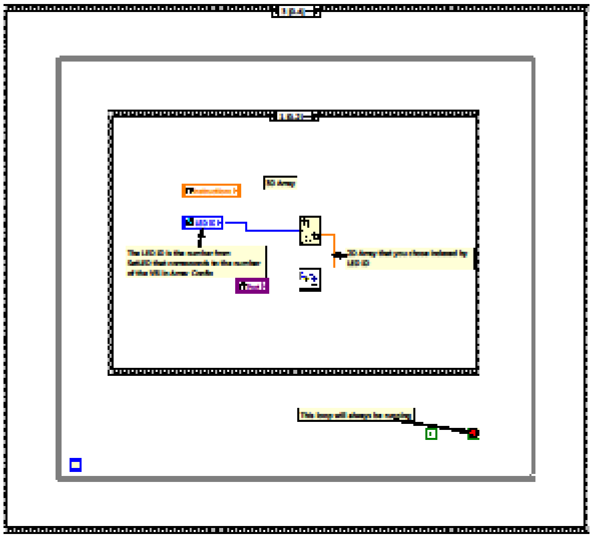
Instructions: 0, 0, 0

parity (0:none): None

Port

Exit as exception
Profile ID is 1.
TimeBased is 1.
add time @ start if relevant

MAKE SURE YOU HAVE THE VISA DRIVER FROM NI!



The LED ID is the number from GetLED that corresponds to the number of the VII in Array Config

2D Array that you chose indexed by LED ID

This loop will always be running

Ramps are fine as long as the delay between commands is greater than time it takes to execute the ramp or it will not stack.

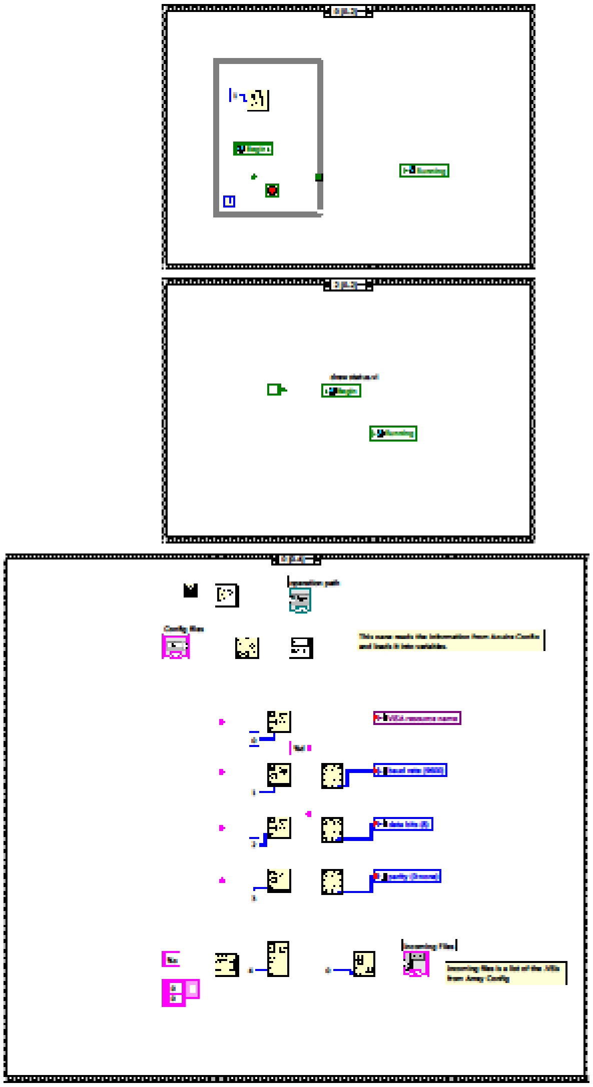
Piping VID to array
Conversions:
Seconds to milliseconds.
LED number to LED address.

Ramp Time is not the ramp time of the NEXT train but the PREVIOUS one.
Each step encodes the new target location.

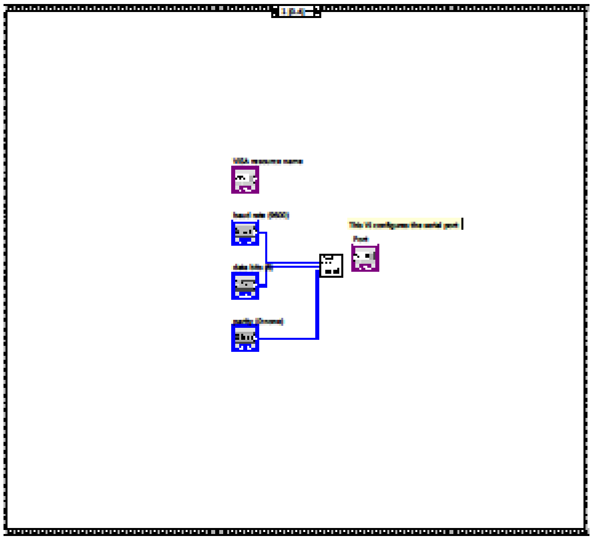
There is a fixed delay + amount of time it takes for the ramp to complete.

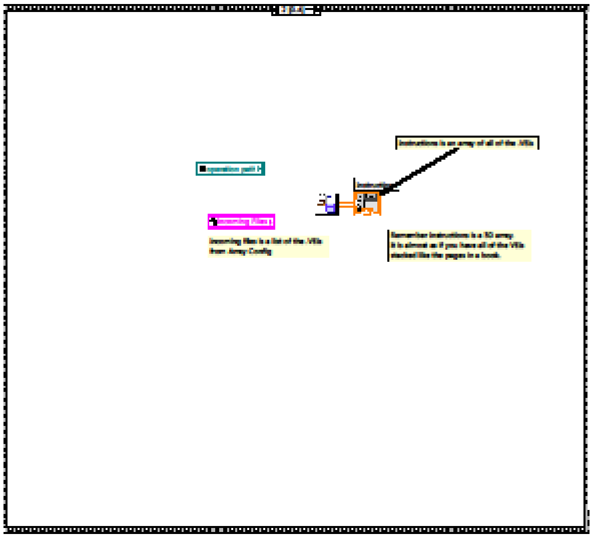
No internal buffer, must wait for each command to complete before sending the next.

Array IO.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Array IO.vi
Last modified on 8/14/2014 at 4:57 PM
Printed on 11/15/2014 at 3:50 PM

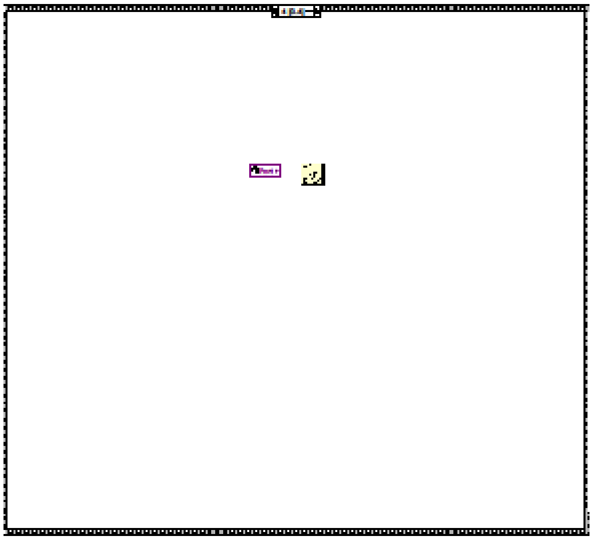


Array IO.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Array IO.vi
Last modified on 8/14/2014 at 4:57 PM
Printed on 11/15/2014 at 3:50 PM




2 [0..4]
Instructions is an array of all of the .VIs
operation poll
Instructions
Incoming Files
Incoming files is a list of the .VIs from Array Config
Remember Instructions is a 3D array.
It is almost as if you have all of the VIs stacked like the pages in a book.

Array IO.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Array IO.vi
Last modified on 8/14/2014 at 4:57 PM
Printed on 11/15/2014 at 3:50 PM

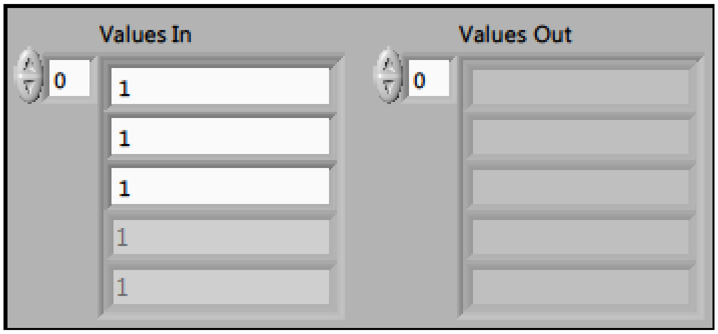


Page 1

Assign.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Assign.vi
Last modified on 1/30/2014 at 3:16 PM
Printed on 11/15/2014 at 3:50 PM

Assign.vi

Values In Values Out

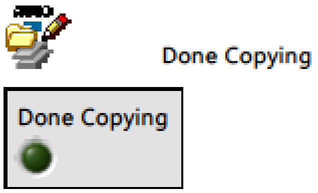


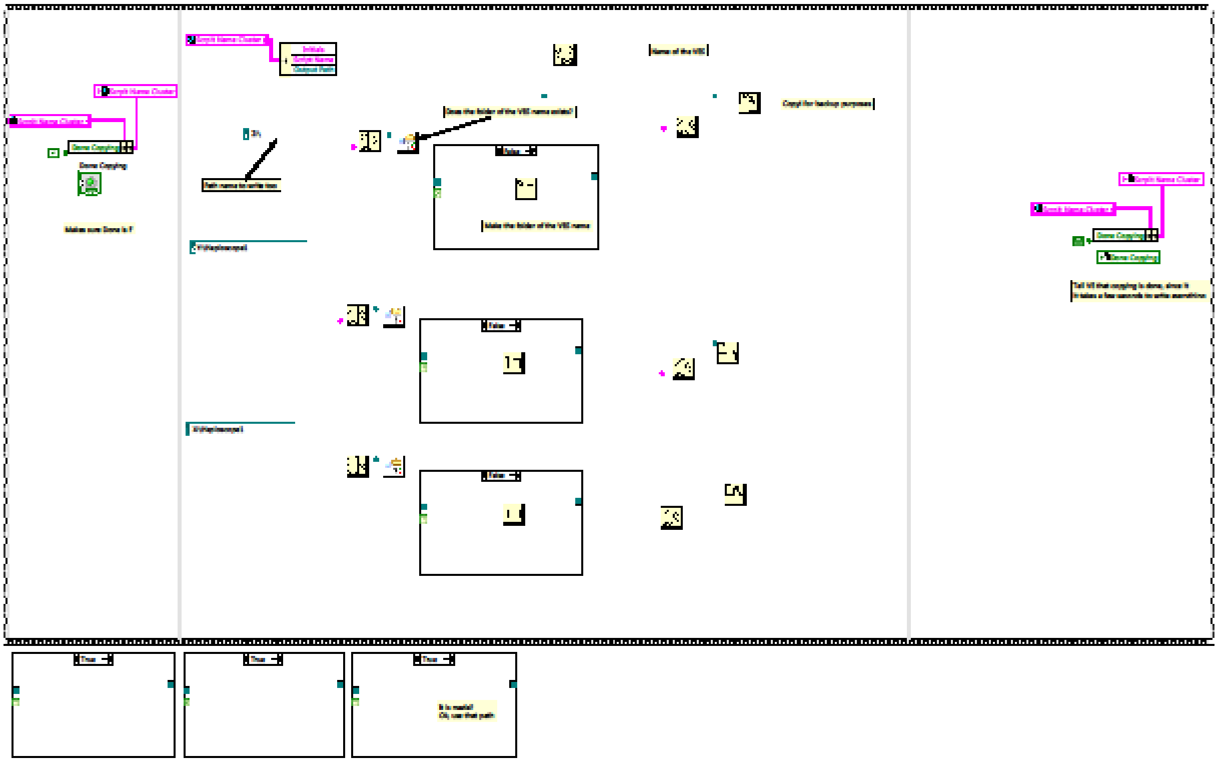
Values In
123
abc
Values Out
123
abc

Page 1

AutoBackup.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\AutoBackup.vi
Last modified on 8/22/2014 at 12:23 PM
Printed on 11/15/2014 at 3:50 PM

AutoBackup.vi

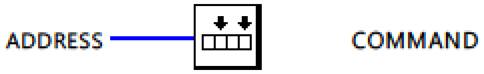


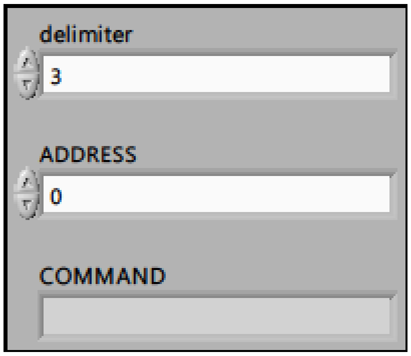


Page 1

Build Command.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Build Command.vi
Last modified on 8/12/2014 at 1:32 PM
Printed on 11/15/2014 at 3:50 PM

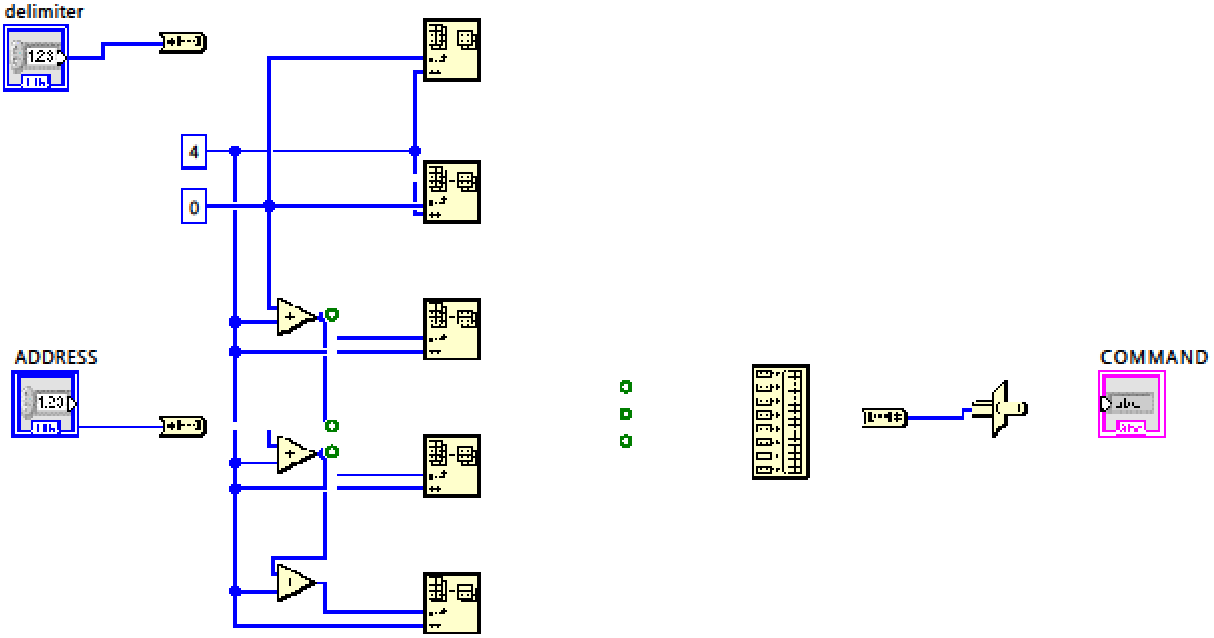
Build Command.vi




delimiter
3

ADDRESS
0

COMMAND



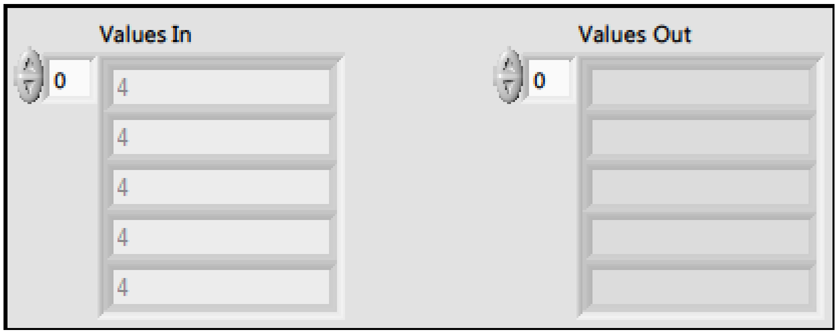
Page 1

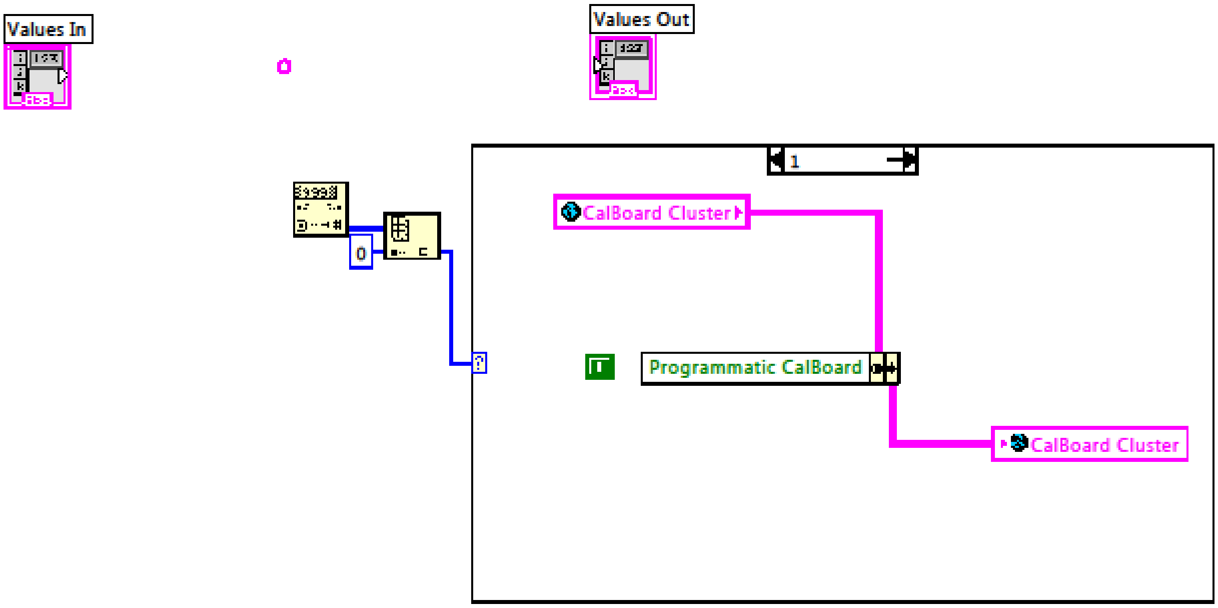
CalBoard.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\CalBoard.vi
Last modified on 8/14/2014 at 5:08 PM
Printed on 11/15/2014 at 3:50 PM

CalBoard.vi

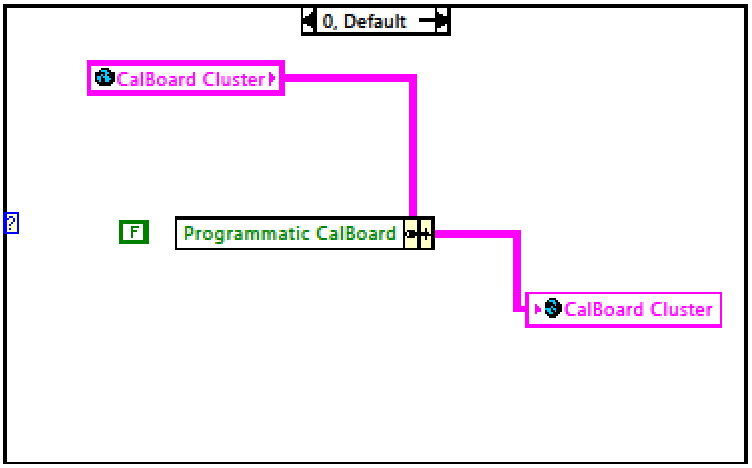
Values Out

Causes the directX renderer to exit.
must be called before end of program.





CalBoard.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\CalBoard.vi
Last modified on 8/14/2014 at 5:08 PM
Printed on 11/15/2014 at 3:50 PM



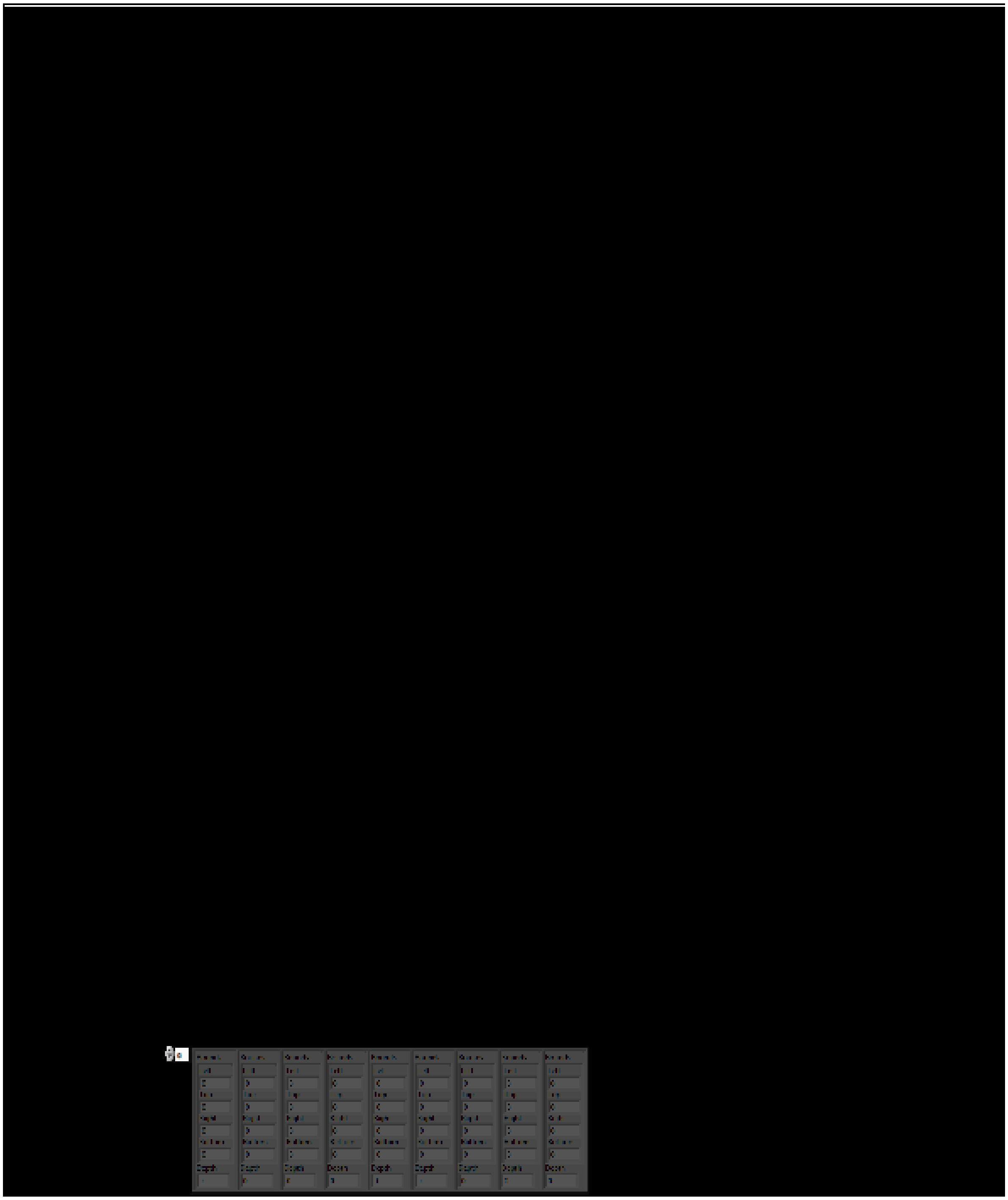
CalDraw.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\CalDraw.vi
Last modified on 8/22/2014 at 12:23 PM
Printed on 11/15/2014 at 3:50 PM

CalDraw.vi

IPD
CAL
DRAW

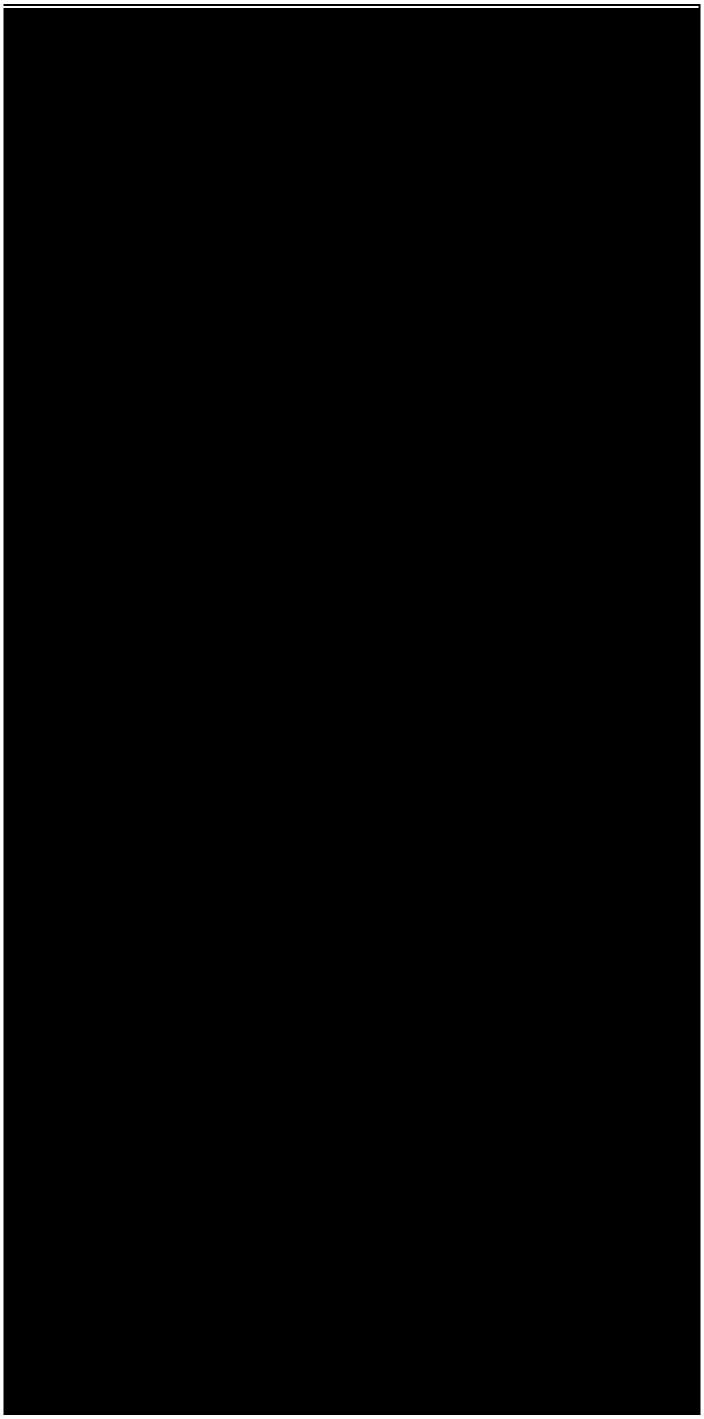
Page 2

CalDraw.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\CalDraw.vi
Last modified on 8/22/2014 at 12:23 PM
Printed on 11/15/2014 at 3:50 PM

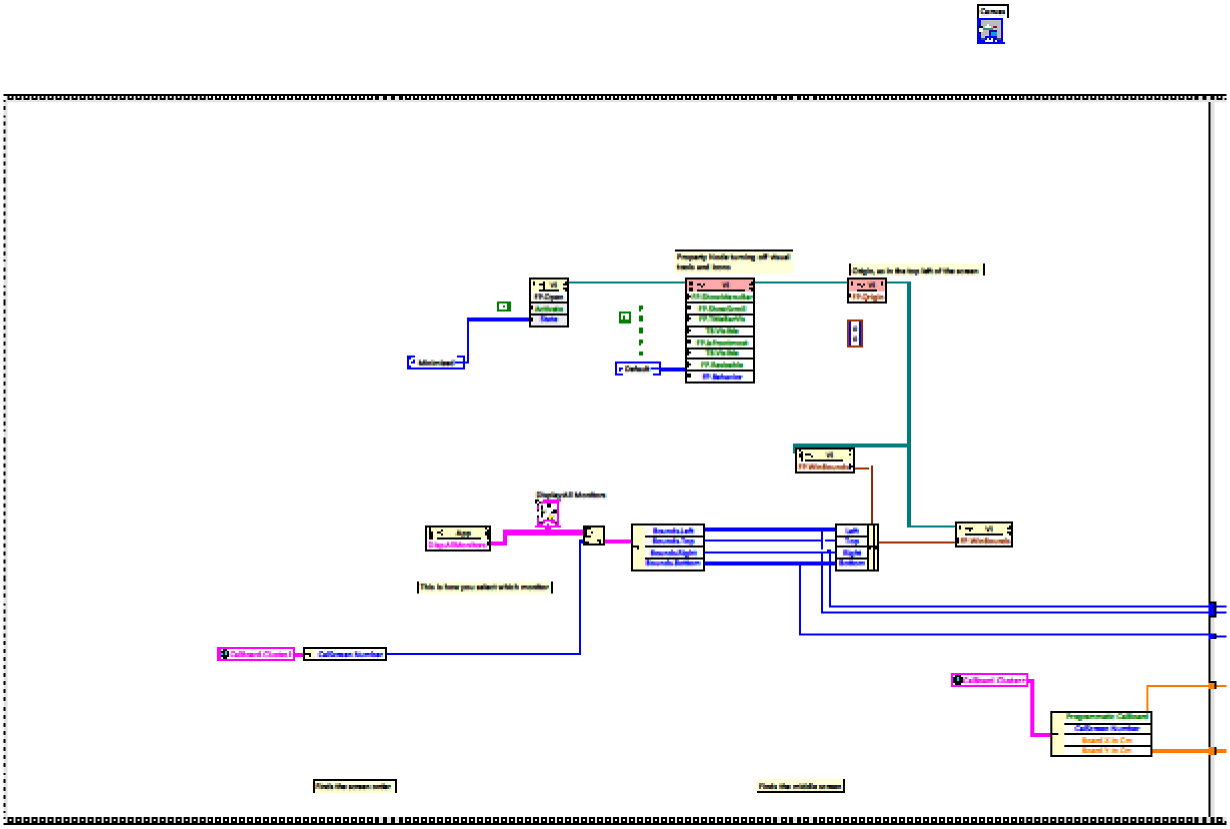


Page 3

CalDraw.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\CalDraw.vi
Last modified on 8/22/2014 at 12:23 PM
Printed on 11/15/2014 at 3:50 PM

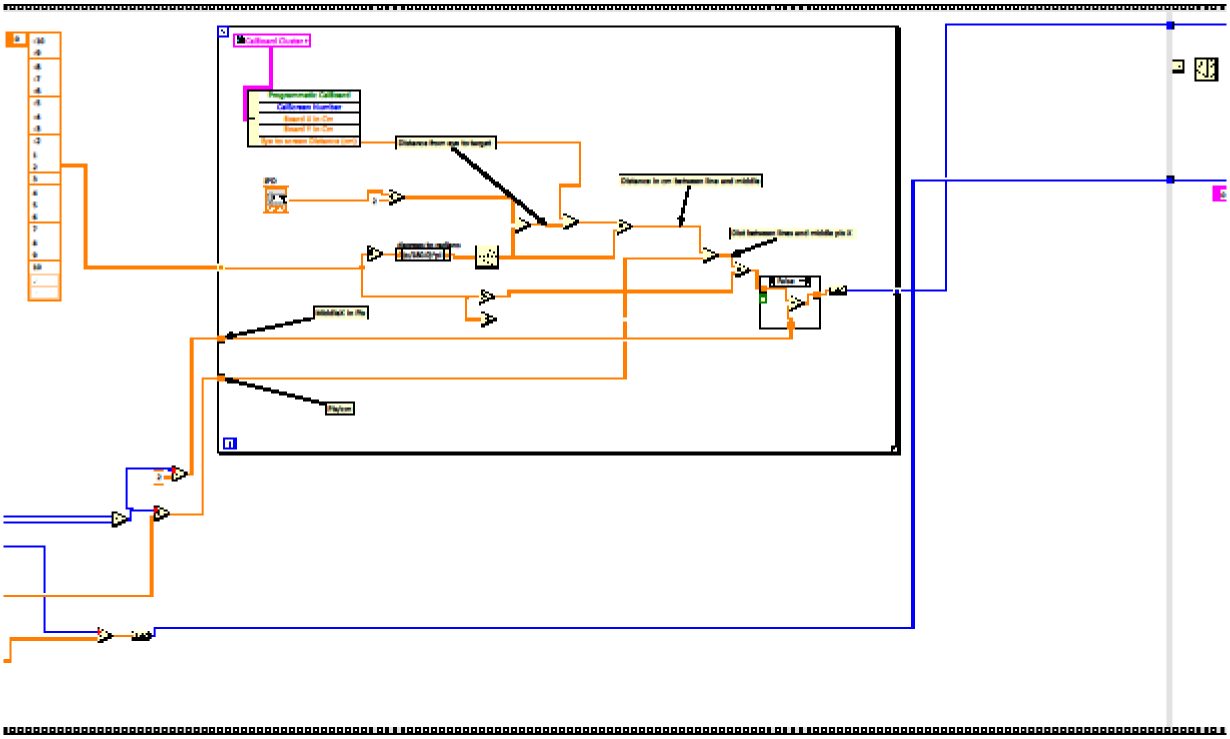


CalDraw.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\CalDraw.vi
Last modified on 8/22/2014 at 12:23 PM
Printed on 11/15/2014 at 3:50 PM



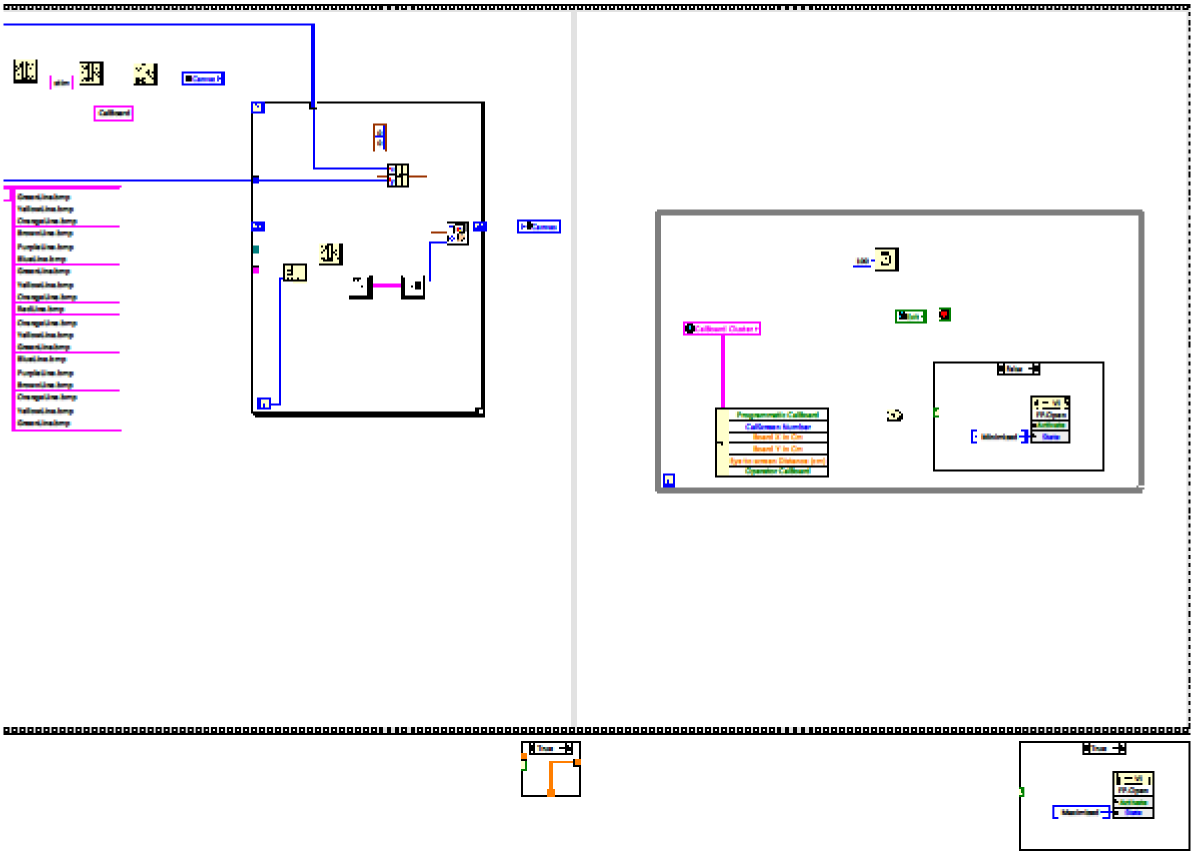
Page 5

CalDraw.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\CalDraw.vi
Last modified on 8/22/2014 at 12:23 PM
Printed on 11/15/2014 at 3:50 PM


CalBoard Cluster
Programmatic CalBoard
CalScreen Number
Board X in Cm
Board Y in Cm
Eye to screen Distance (cm)
Distance from eye to target
IPD
Distance in cm between line and middle
degrees to radians
Dist between lines and middle pix X
False
MiddleX in Px
Pix/cm

Page 6

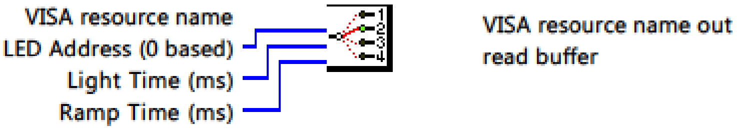
CalDraw.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\CalDraw.vi
Last modified on 8/22/2014 at 12:23 PM
Printed on 11/15/2014 at 3:50 PM

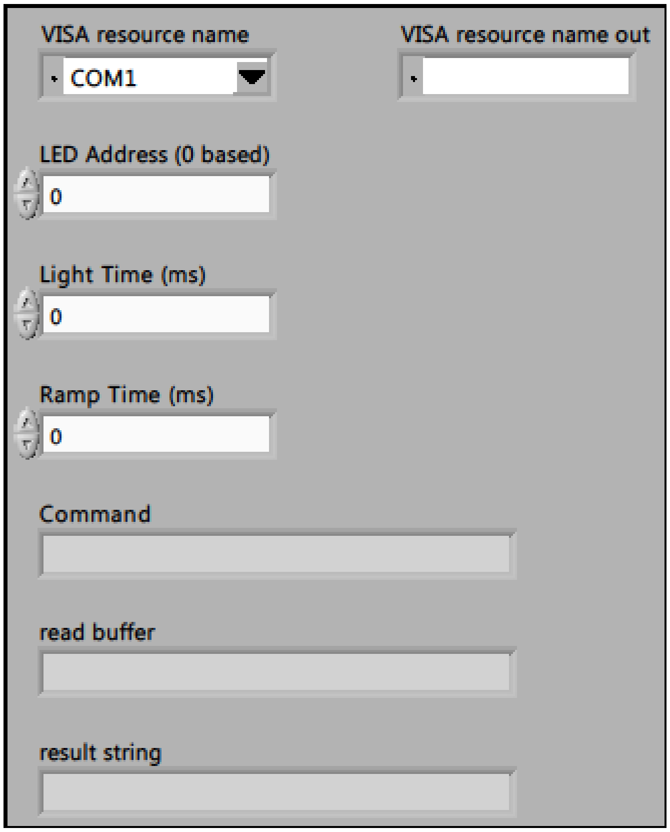


Page 1

Command Array.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Command Array.vi
Last modified on 8/12/2014 at 1:35 PM
Printed on 11/15/2014 at 3:50 PM

Command Array.vi





VISA resource name: COM1

VISA resource name out:

LED Address (0 based): 0

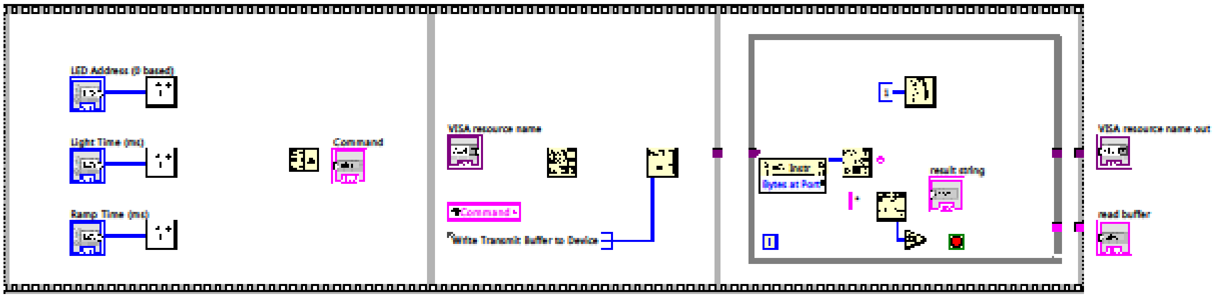
Light Time (ms): 0

Ramp Time (ms): 0

Command:

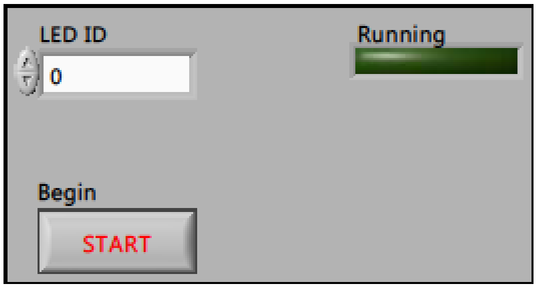
read buffer:

result string:



draw status.vi
C:\Users\Steve\Desktop\VisualEyes 2020\VI\draw status.vi
Last modified on 3/4/2014 at 6:40 PM
Printed on 11/15/2014 at 3:50 PM

draw status.vi



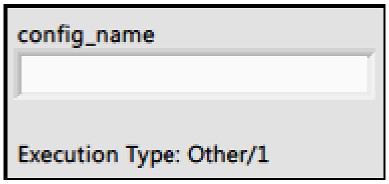
Draw.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Draw.vi
Last modified on 7/25/2014 at 5:32 PM
Printed on 11/15/2014 at 3:50 PM

Draw.vi

config_name

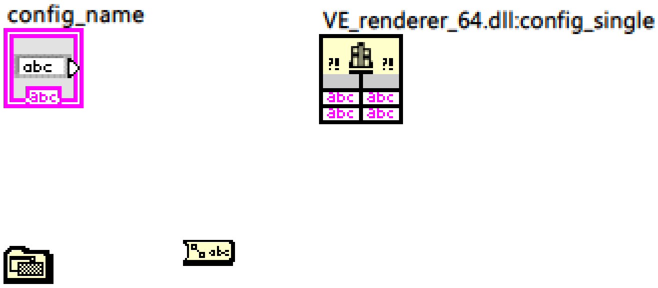
Receive Single string input that denote the LIST of draw config files.
Opens VI windows and runs draw subroutine.

VI wrapper for the Visual Eyes Direct X renderer.
This VI have Execution type of "Other 1" allowing it to run constantly.
Other VI wrap functions that set static variables of the Libary.



config_name

Execution Type: Other/1

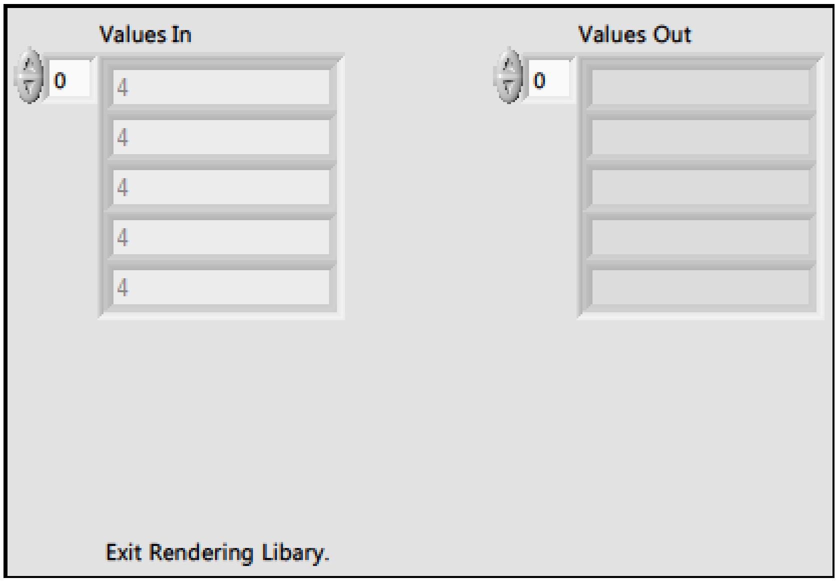


EndDraw.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\EndDraw.vi
Last modified on 4/14/2014 at 6:27 PM
Printed on 11/15/2014 at 3:50 PM

EndDraw.vi

Values Out

Causes the directX renderer to exit.
must be called before end of program.



Values In

0

4

4

4

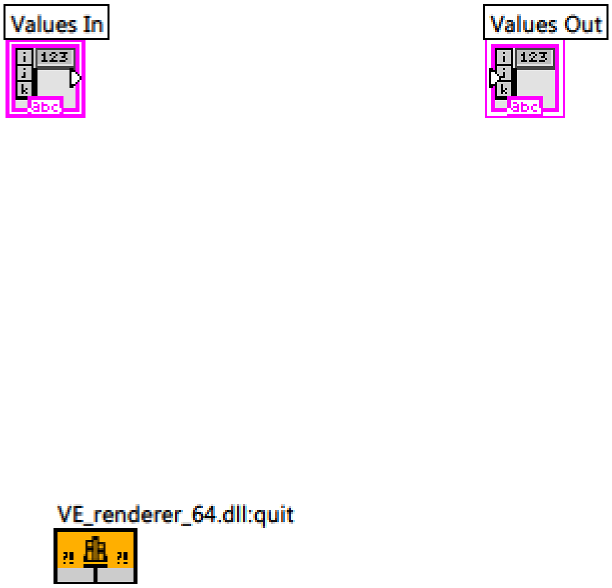
4

4

Values Out

0

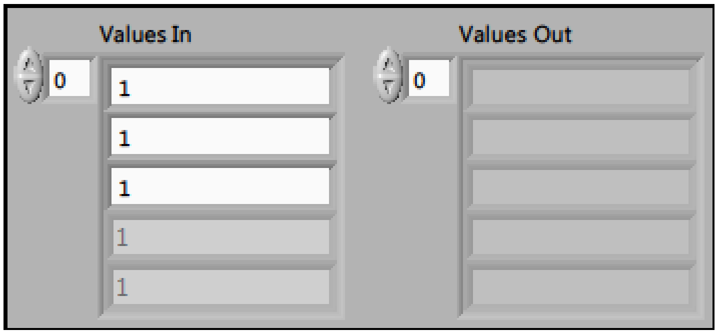
Exit Rendering Libary.



EndRepeat.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\EndRepeat.vi
Last modified on 1/30/2014 at 3:18 PM
Printed on 11/15/2014 at 3:50 PM

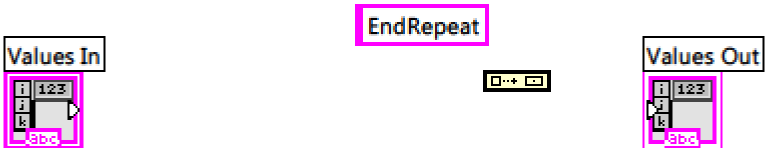
EndRepeat.vi

Values In   Values Out



Values In
0
1
1
1
1
1

Values Out
0



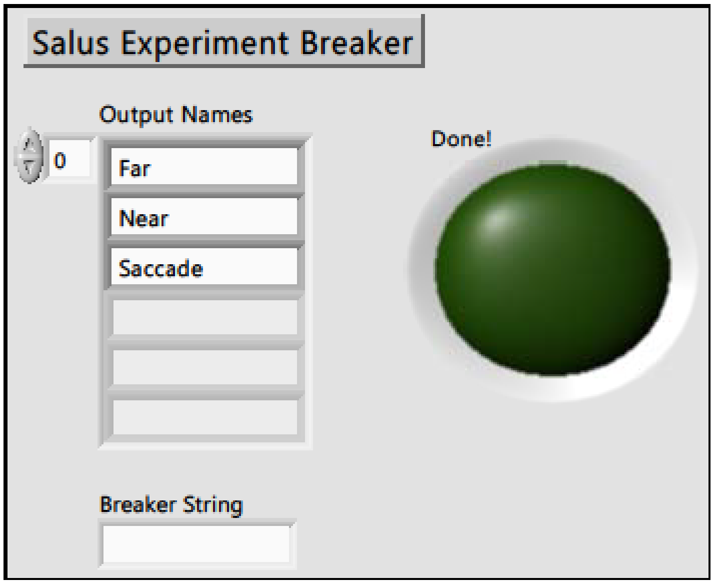
Page 1

ExpBreaker.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ExpBreaker.vi
Last modified on 8/6/2014 at 3:45 PM
Printed on 11/15/2014 at 3:50 PM

ExpBreaker.vi

Output Names
Breaker String

Done!



Salus Experiment Breaker

Output Names

0

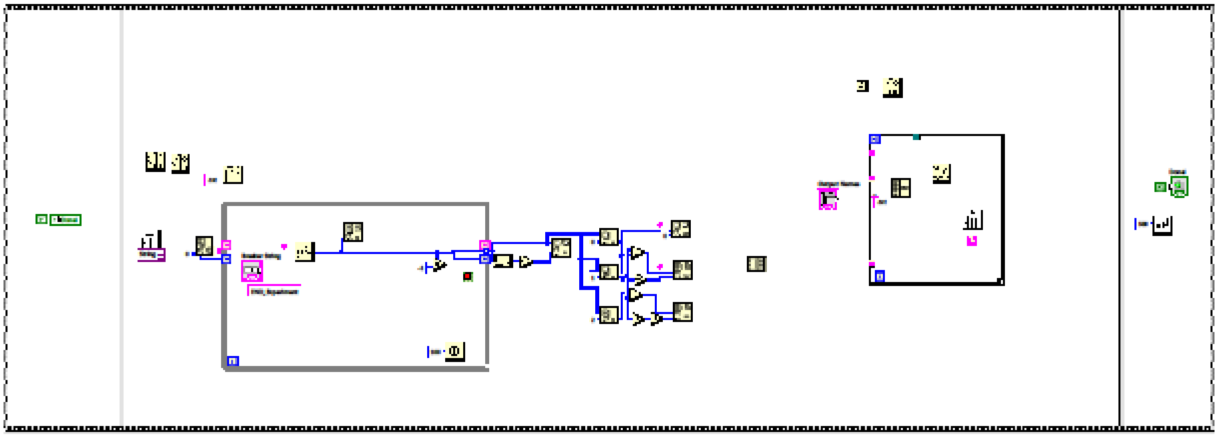
Far

Near

Saccade

Done!

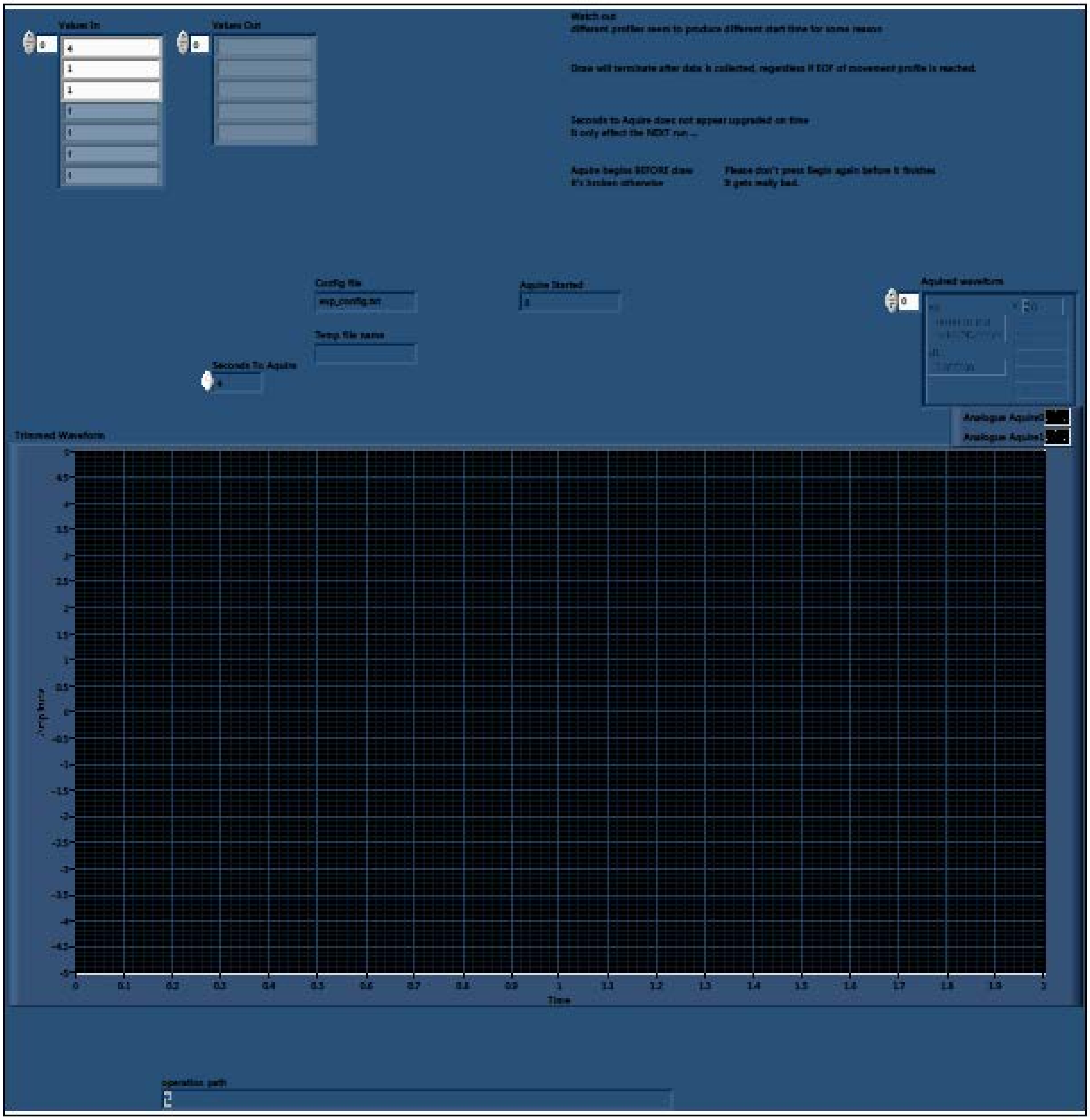
Breaker String



ExpTrial.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ExpTrial.vi
Last modified on 8/22/2014 at 12:01 PM
Printed on 11/15/2014 at 3:50 PM

ExpTrial.vi

Values In Values Out



Watch out
different profiles seem to produce different start time for some reason

Daq will terminate after data is collected, regardless if EOF of movement profile is reached.

Seconds to Aquire does not appear upgraded on time
It only affect the NEXT run ...

Aquire begins BEFORE daq
It's broken otherwise

Please don't press Begin again before it finishes
It gets really bad.

Values In: 0 | 4 | 1 | 1

Values Out: 0

Config file: exp_config.txt

Temp file name:

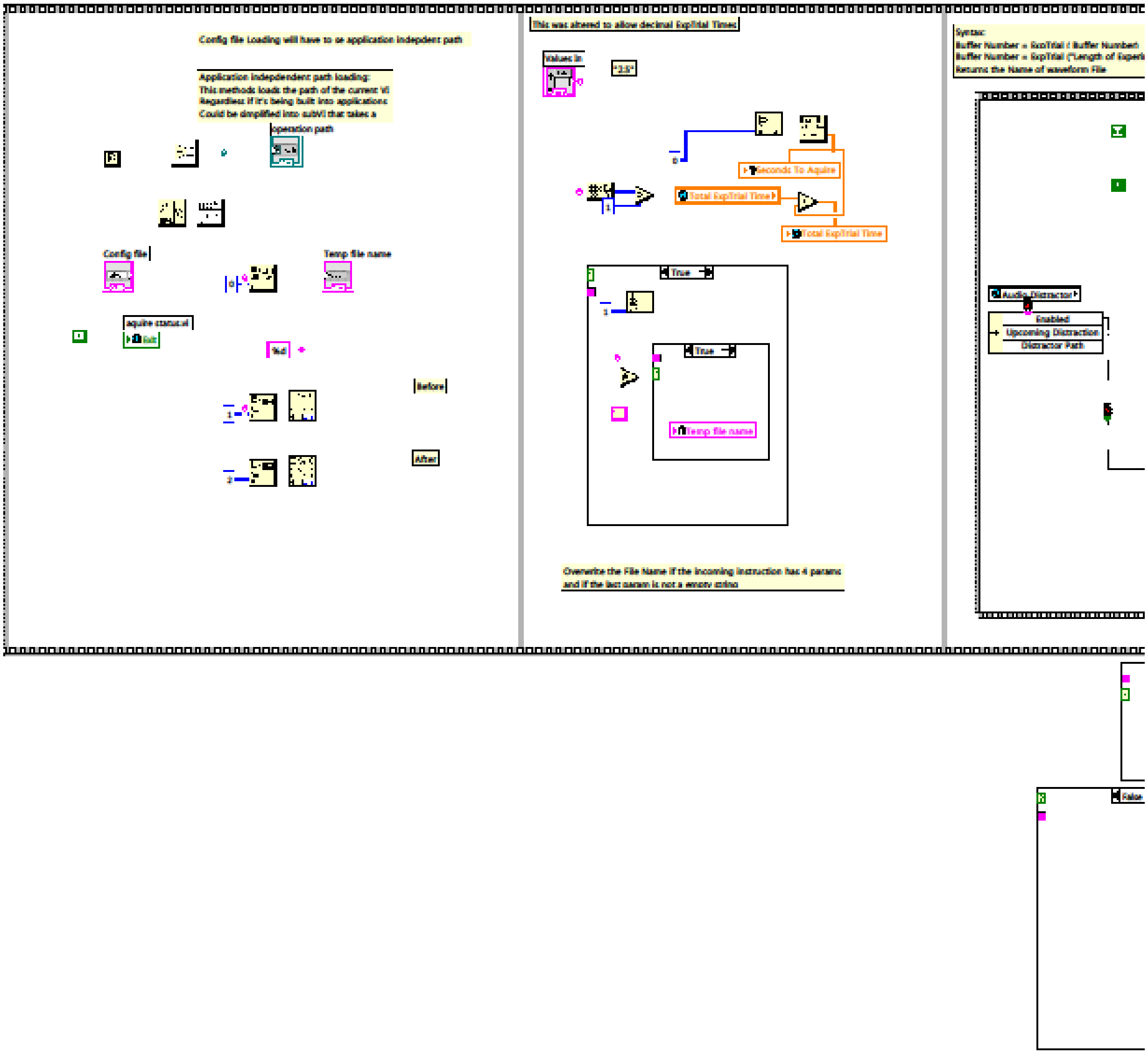
Seconds To Aquire: 4

Aquire Started: 0

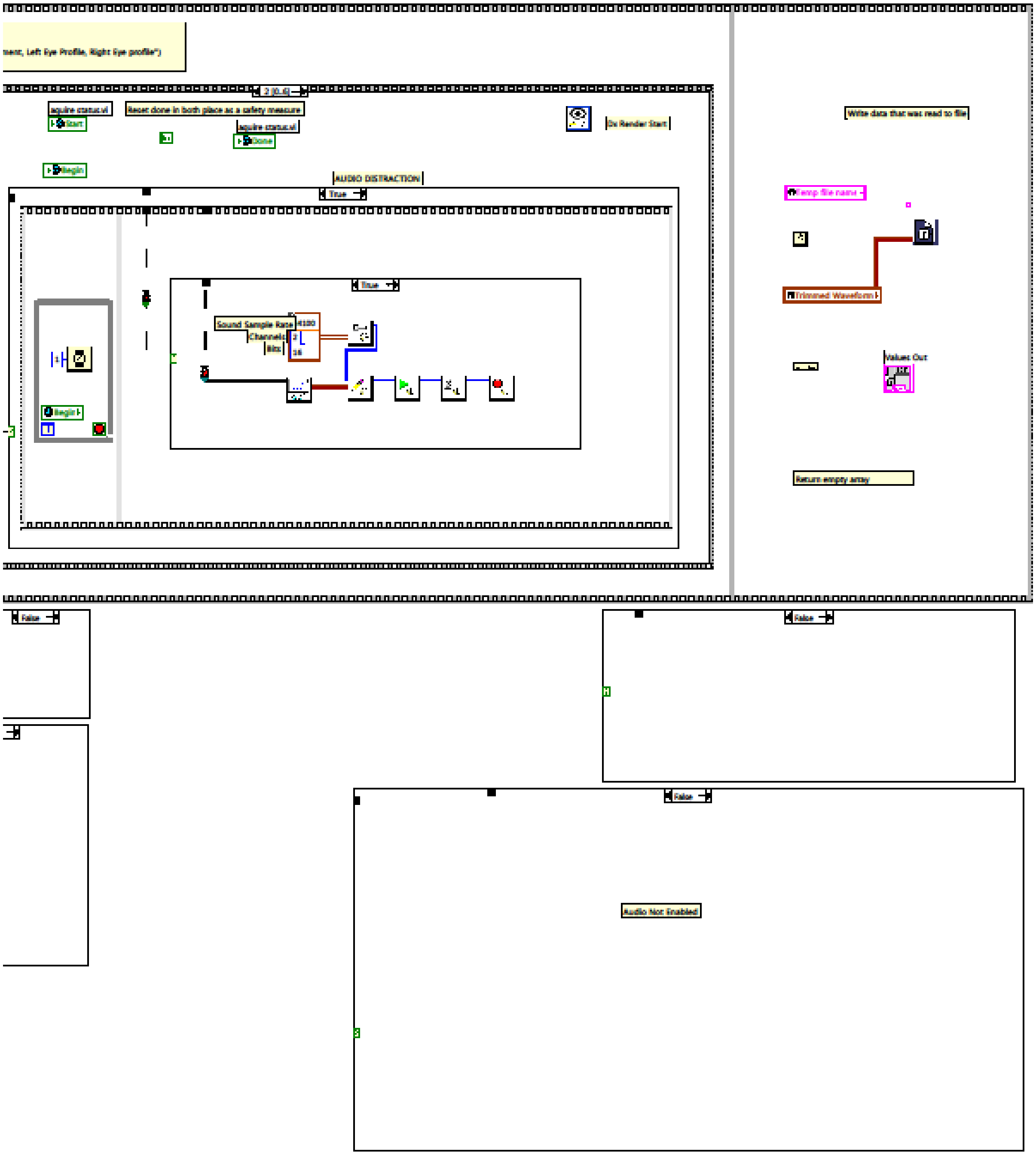
Aquired waveform: 0

operation path:

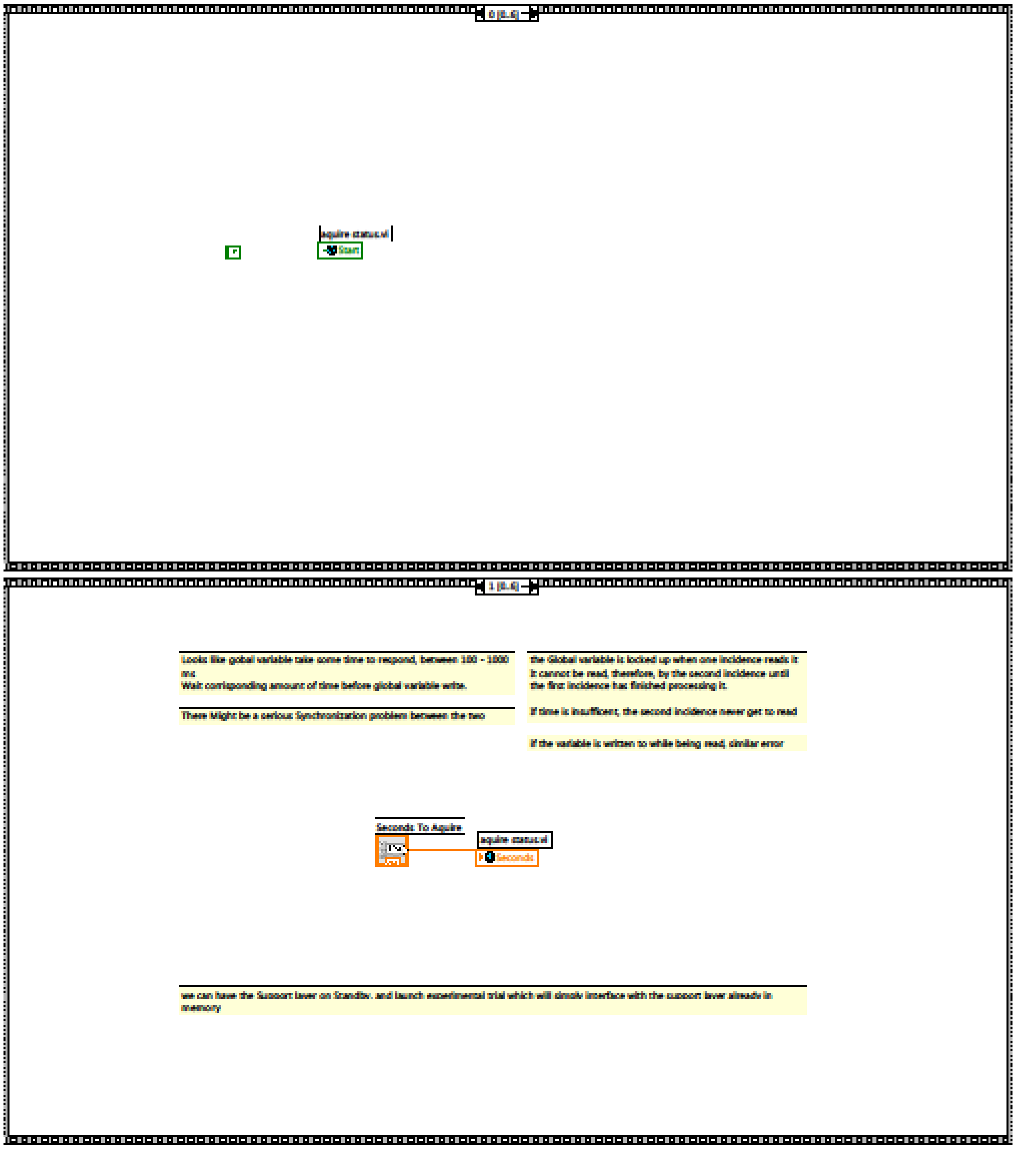
ExpTrial.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ExpTrial.vi
Last modified on 8/22/2014 at 12:01 PM
Printed on 11/15/2014 at 3:50 PM


Config file Loading will have to se application indepdent path
Application independendent path loading:
This methods loads the path of the current VI
Regardless if it's being built into applications
Could be dimplified into subVI that takes a
operation path
Config file
Temp file name
aquire status.vi
Exit
%d
Before
After
This was altered to allow decimal ExpTrial Times
Values In
"2.5"
Seconds To Aquire
Total ExpTrial Time
Total ExpTrial Time
True
True
Temp file name
Overwrite the File Name if the incoming instruction has 4 params
and if the last param is not a empty string
Syntax:
Buffer Number = ExpTrial (Buffer Number)
Buffer Number = ExpTrial ("Length of Experi
Returns the Name of waveform File
Audio Distractor
Enabled
Upcoming Distraction
Distractor Path
False

ExpTrial.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ExpTrial.vi
Last modified on 8/22/2014 at 12:01 PM
Printed on 11/15/2014 at 3:50 PM



ExpTrial.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ExpTrial.vi
Last modified on 8/22/2014 at 12:01 PM
Printed on 11/15/2014 at 3:50 PM



Looks like gobal variable take some time to respond, between 100 - 1000 ms.
Wait corrisponding amount of time before global variable write.

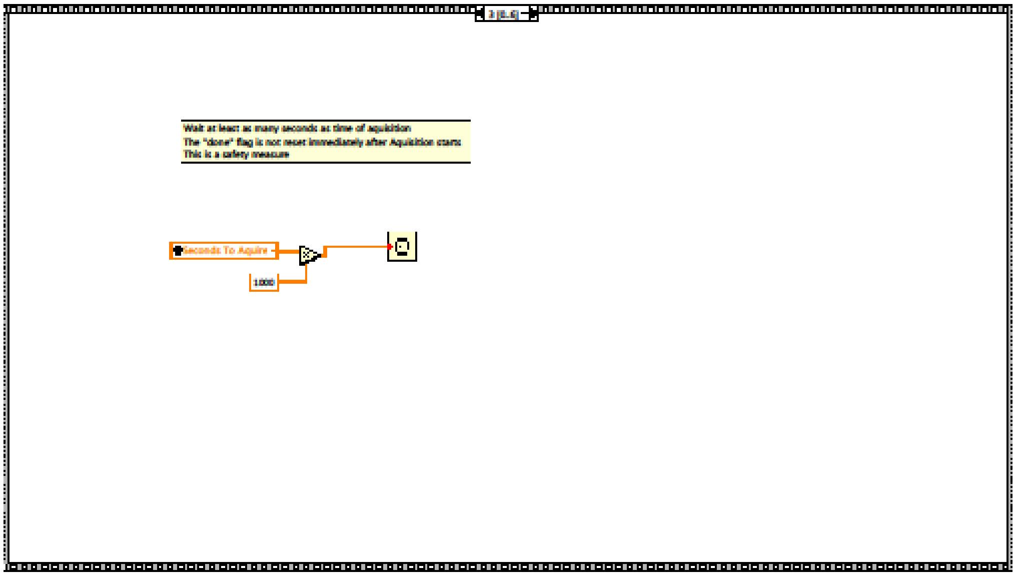
There Might be a serious Synchronization problem between the two

the Global variable is locked up when one incidence reads it
It cannot be read, therefore, by the second incidence until the first incidence has finished processing it.

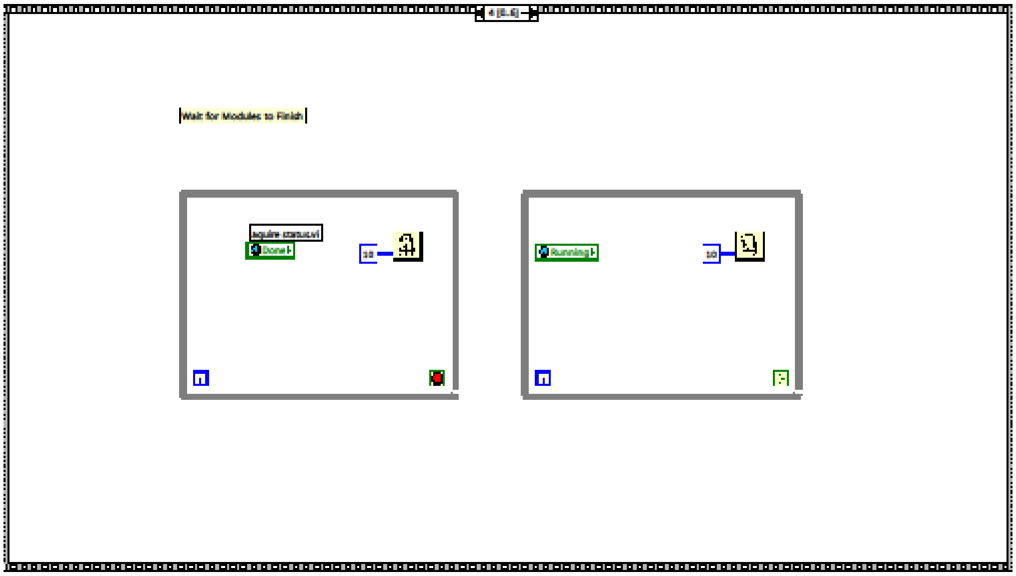
If time is insufficent, the second incidence never get to read

If the variable is writen to while being read, similar error

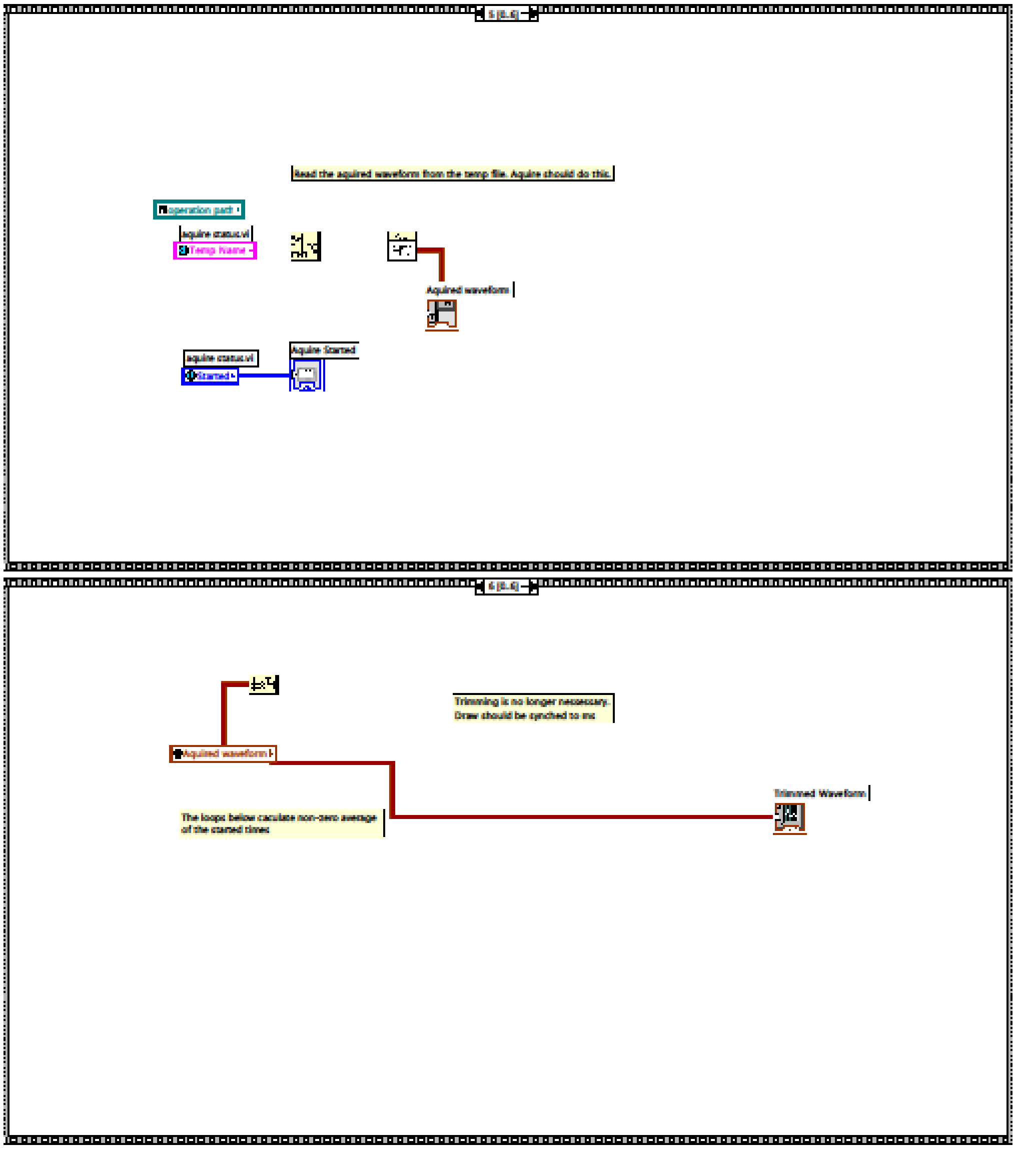
we can have the Support layer on Standby, and launch experimental trial which will simply interface with the support layer already in memory



Wait at least as many seconds as time of aquisition
The "done" flag is not reset immediately after Aquisition starts
This is a safety measure



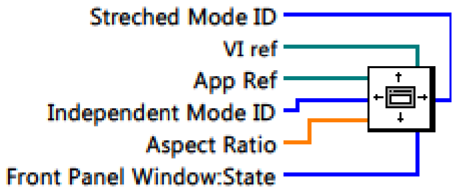
Wait for Modules to Finish


5 [0..6]
Read the aquired waveform from the temp file. Aquire should do this.
operation path
aquire status.vi
Temp Name
Aquired waveform
aquire status.vi
Started
Aquire Started
6 [0..6]
Trimming is no longer necessary.
Draw should be synched to ms
Aquired waveform
The loops below caculate non-zero average
of the started times
Trimmed Waveform

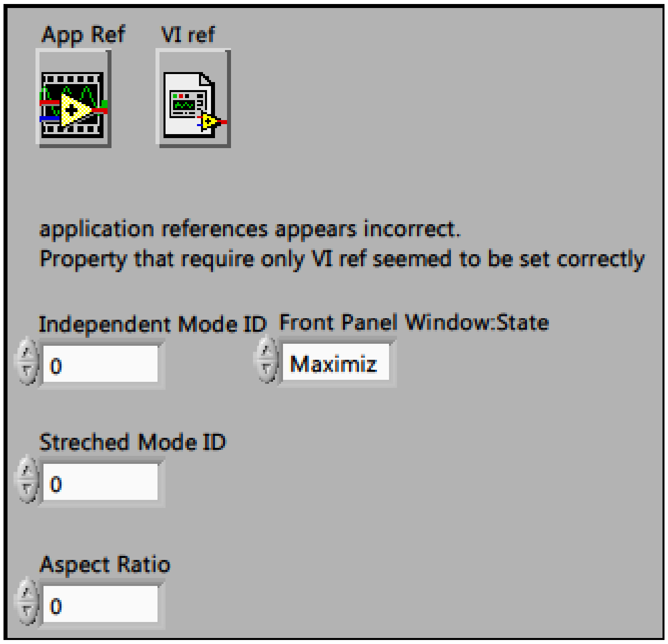
Page 1

fpbehave.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\fpbehave.vi
Last modified on 2/27/2014 at 12:38 PM
Printed on 11/15/2014 at 3:50 PM

fpbehave.vi



This VI sets Behavior of Front Panel of the refered VI.
It simplify the task of setting similar front Panel Behavior to multiple VIs.



application references appears incorrect.
Property that require only VI ref seemed to be set correctly

Independent Mode ID: 0

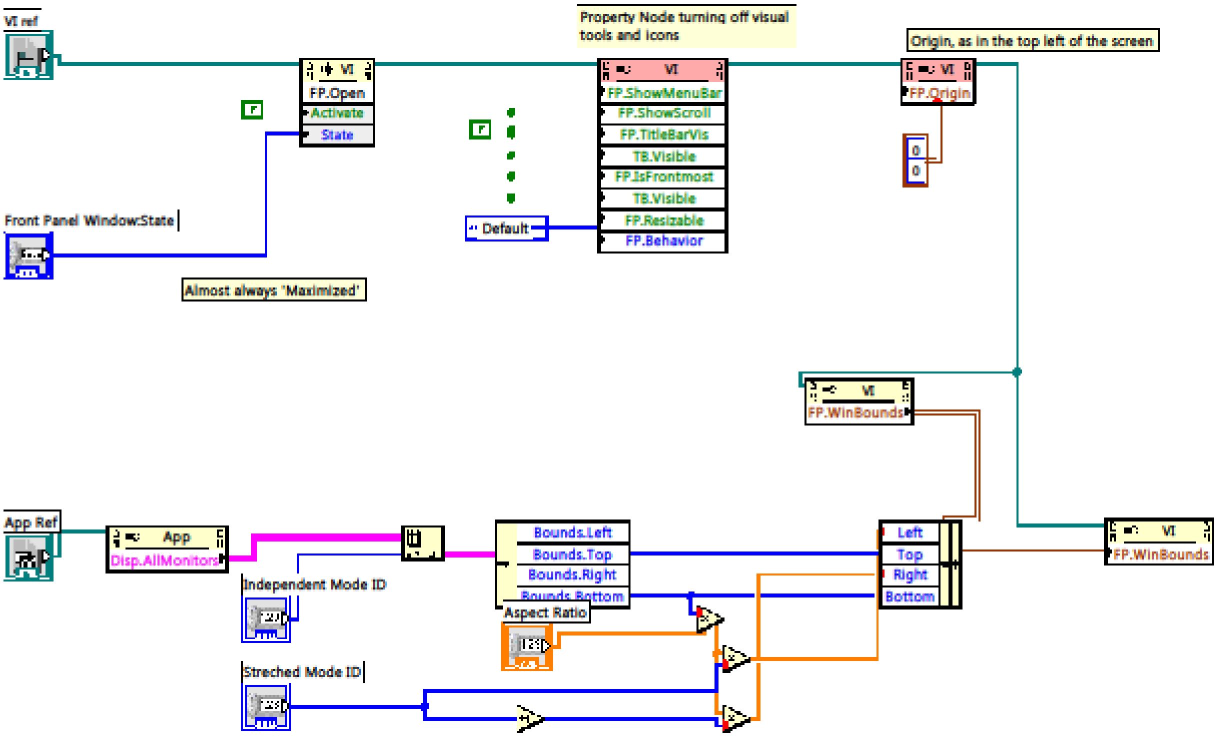
Front Panel Window:State: Maximiz

Streched Mode ID: 0

Aspect Ratio: 0

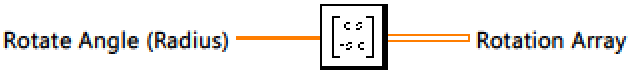
Page 2

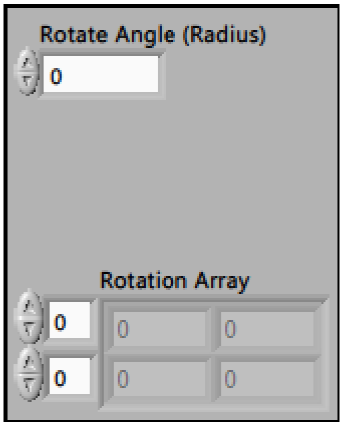
fpbehave.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\fpbehave.vi
Last modified on 2/27/2014 at 12:38 PM
Printed on 11/15/2014 at 3:50 PM

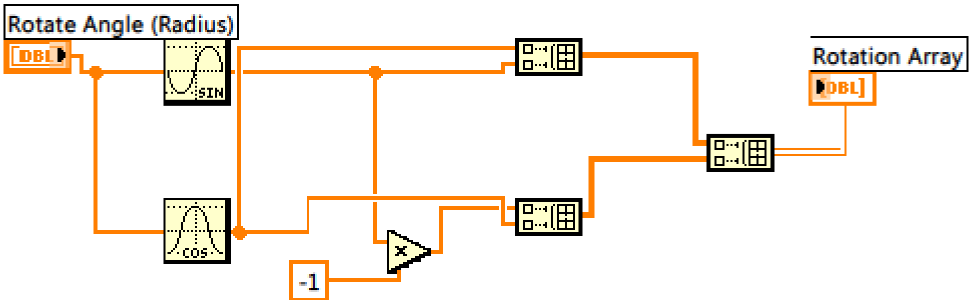


getrarray.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\getrarray.vi
Last modified on 2/12/2014 at 12:52 PM
Printed on 11/15/2014 at 3:50 PM

getrarray.vi



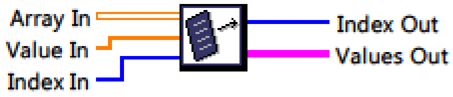


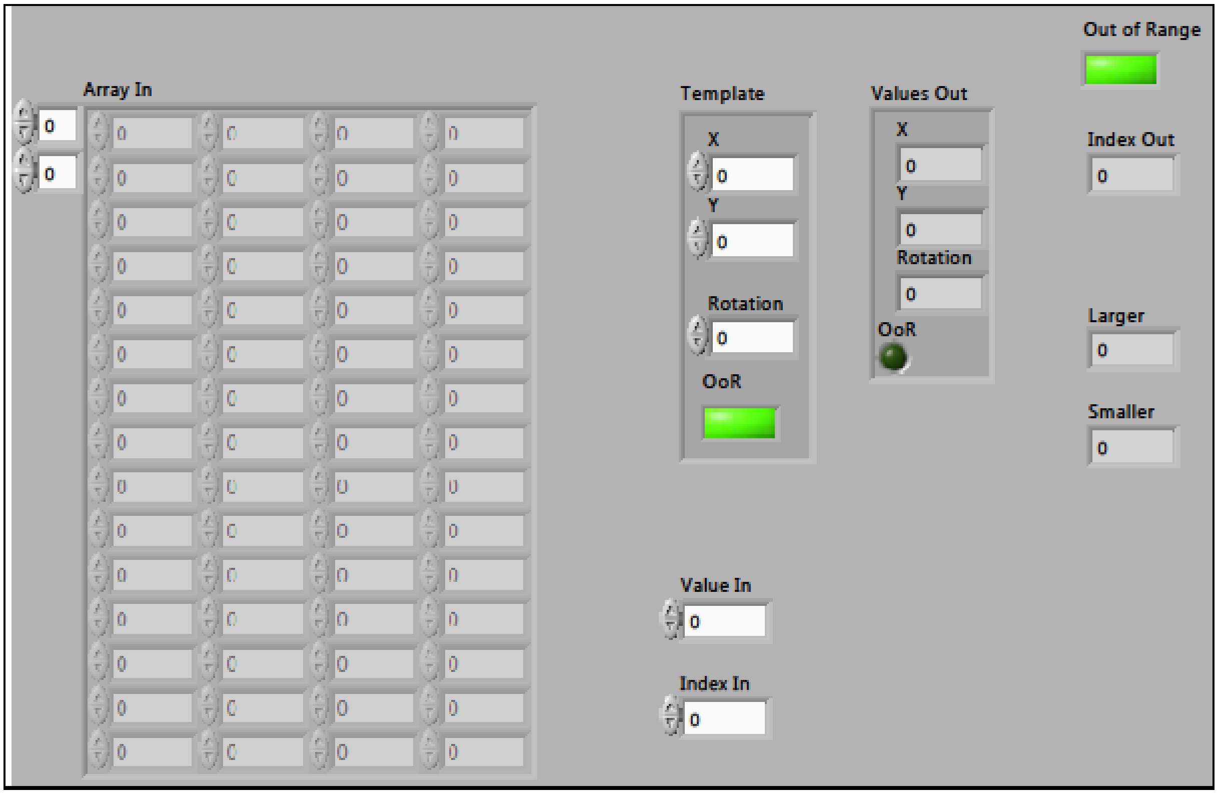


Page 1

Interpret Instructions.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Interpret Instructions.vi
Last modified on 2/12/2014 at 12:52 PM
Printed on 11/15/2014 at 3:50 PM

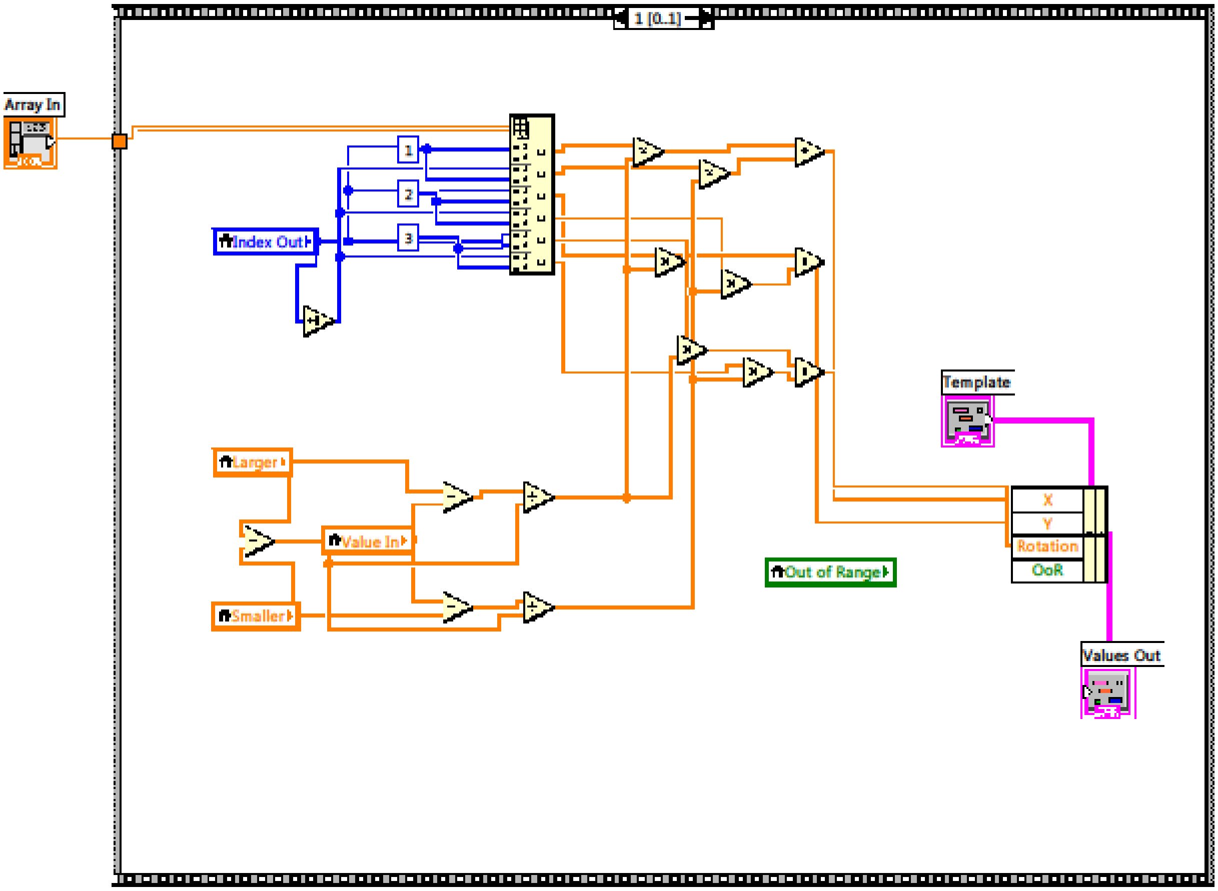
Interpret Instructions.vi



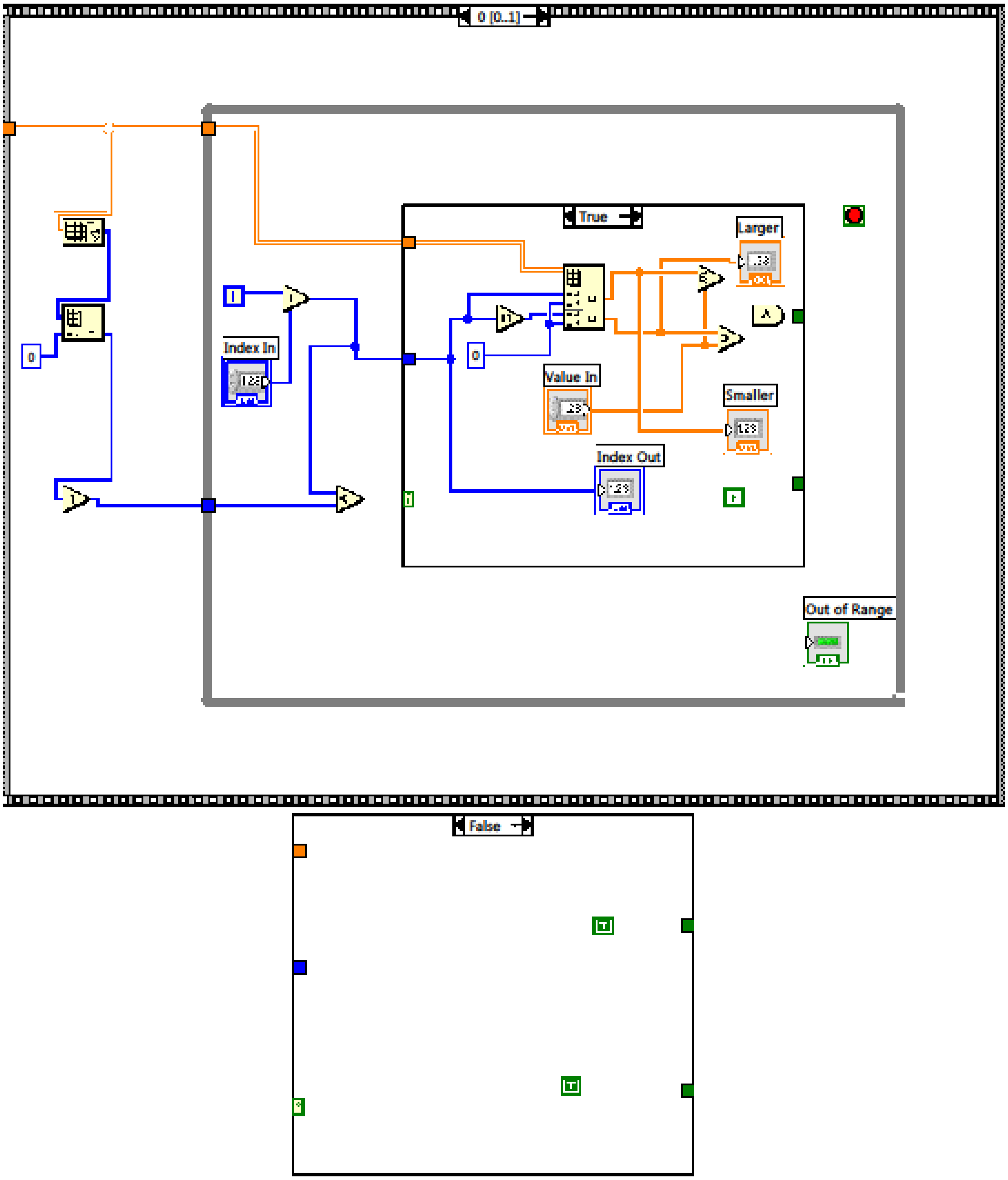


Page 2

Interpret Instructions.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Interpret Instructions.vi
Last modified on 2/12/2014 at 12:52 PM
Printed on 11/15/2014 at 3:50 PM

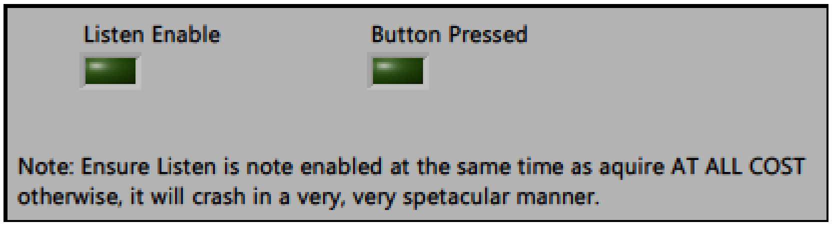


Interpret Instructions.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Interpret Instructions.vi
Last modified on 2/12/2014 at 12:52 PM
Printed on 11/15/2014 at 3:50 PM



listen status.vi
C:\Users\Steve\Desktop\VisualEyes 2020\VI\listen status.vi
Last modified on 2/12/2014 at 12:44 PM
Printed on 11/15/2014 at 3:51 PM

listen status.vi



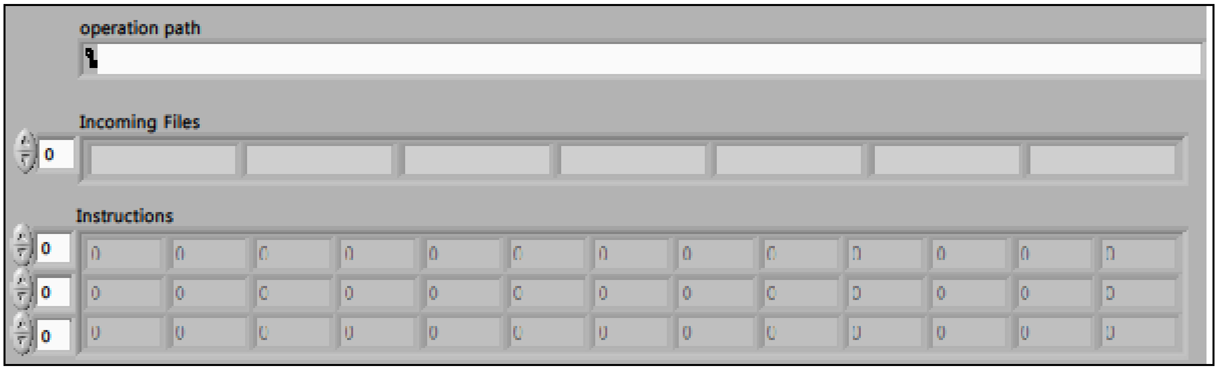
Note: Ensure Listen is note enabled at the same time as aquire AT ALL COST otherwise, it will crash in a very, very spetacular manner.

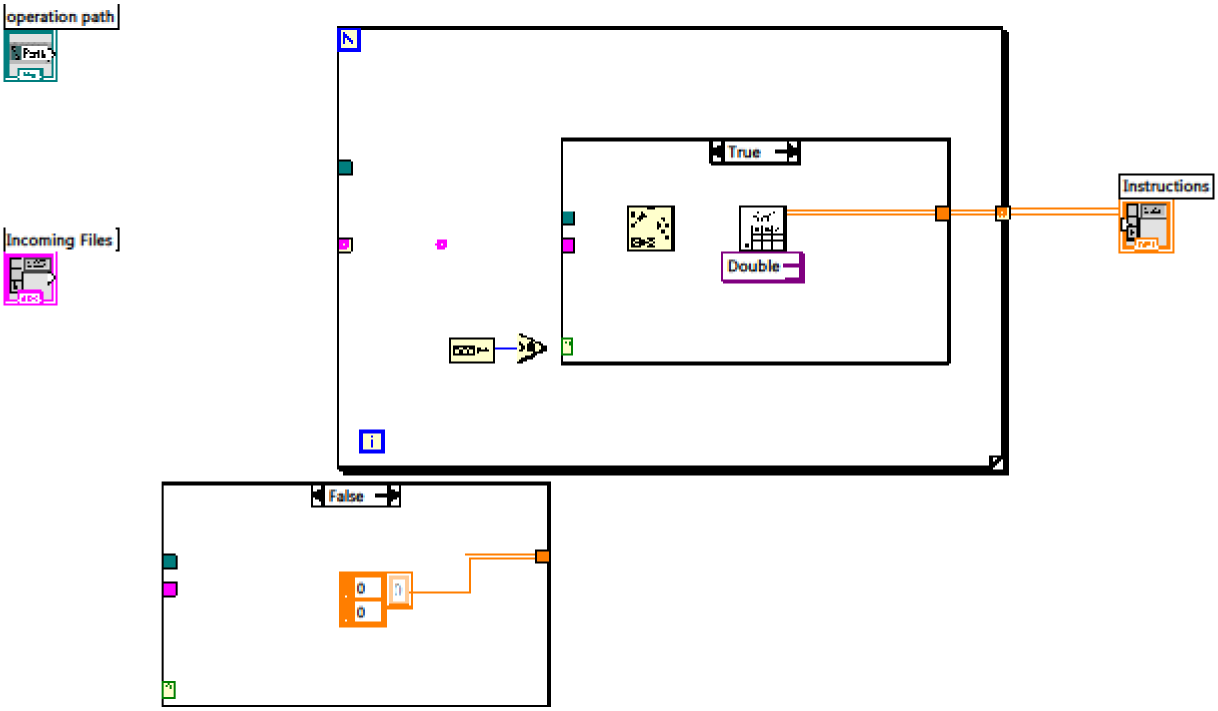
Load Instructions.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Load Instructions.vi
Last modified on 8/12/2014 at 12:27 PM
Printed on 11/15/2014 at 3:51 PM

Load Instructions.vi

operation path
Incoming Files

Instructions

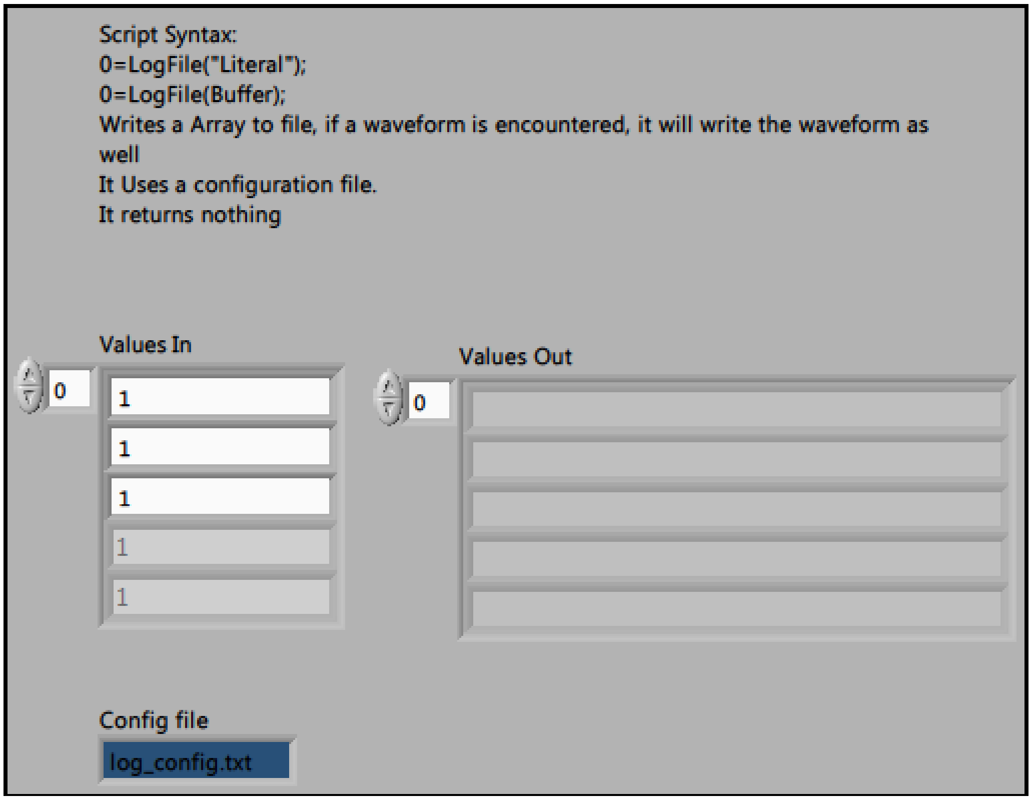




LogFile.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\LogFile.vi
Last modified on 8/20/2014 at 2:51 PM
Printed on 11/15/2014 at 3:51 PM

LogFile.vi

Values In Values Out



Script Syntax:
0=LogFile("Literal");
0=LogFile(Buffer);
Writes a Array to file, if a waveform is encountered, it will write the waveform as well
It Uses a configuration file.
It returns nothing

Values In

0

1
1
1
1
1

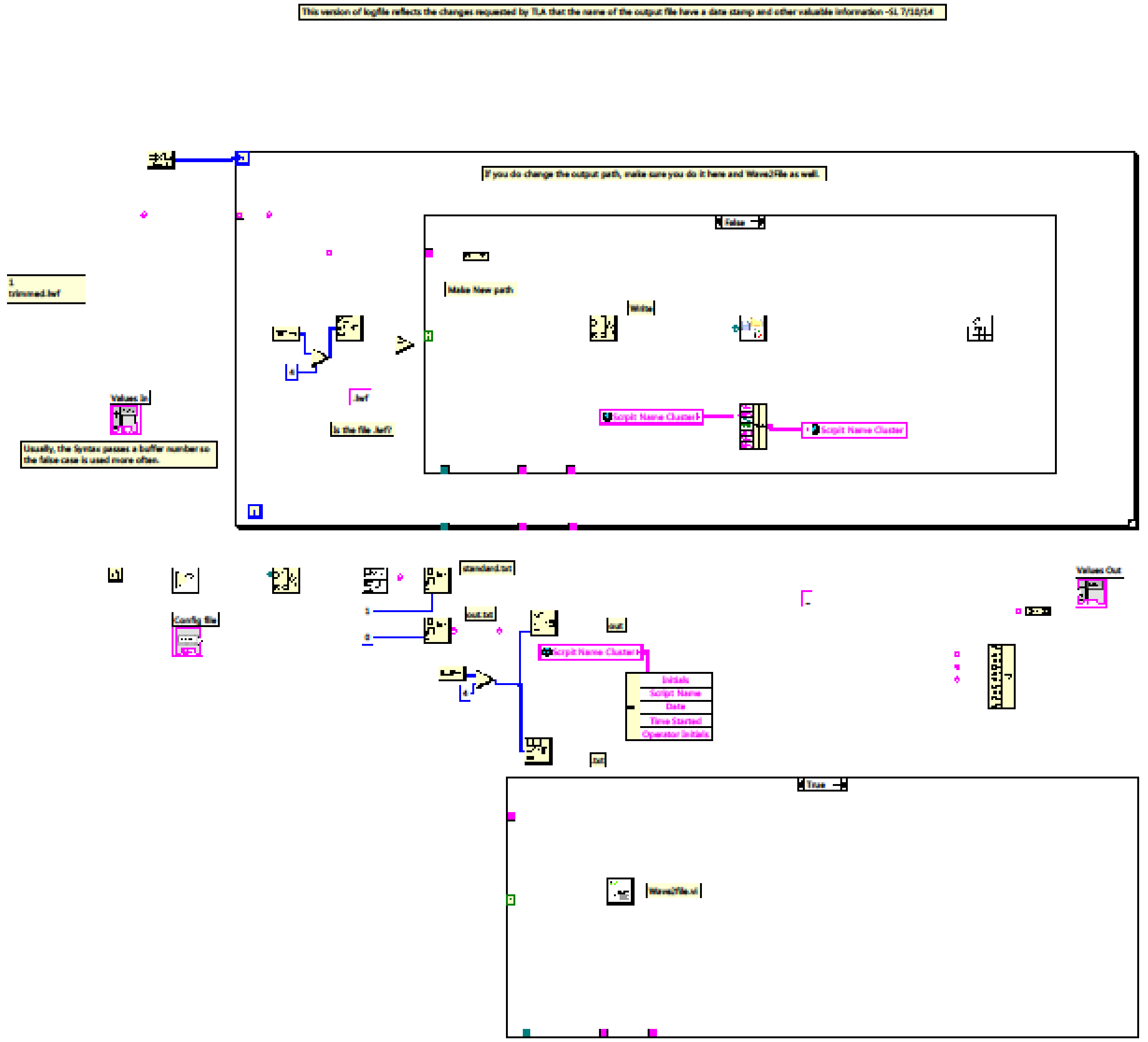
Values Out

0

Config file

log_config.txt

LogFile.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\LogFile.vi
Last modified on 8/20/2014 at 2:51 PM
Printed on 11/15/2014 at 3:51 PM



This version of logfile reflects the changes requested by TLA that the name of the output file have a date stamp and other valuable information -SL 7/10/14

If you do change the output path, make sure you do it here and Wave2File as well.

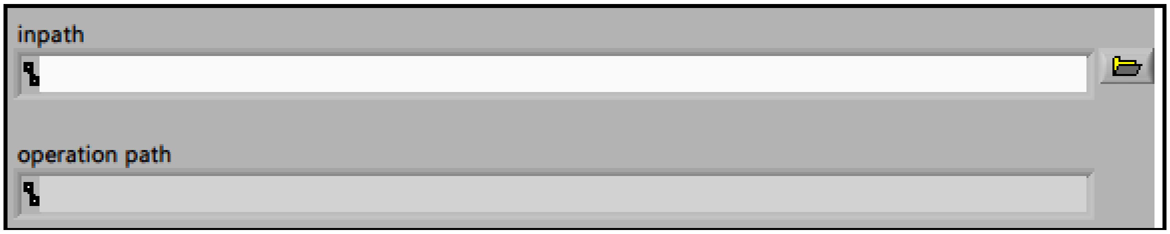
Usually, the Syntax passes a buffer number so the false case is used more often.

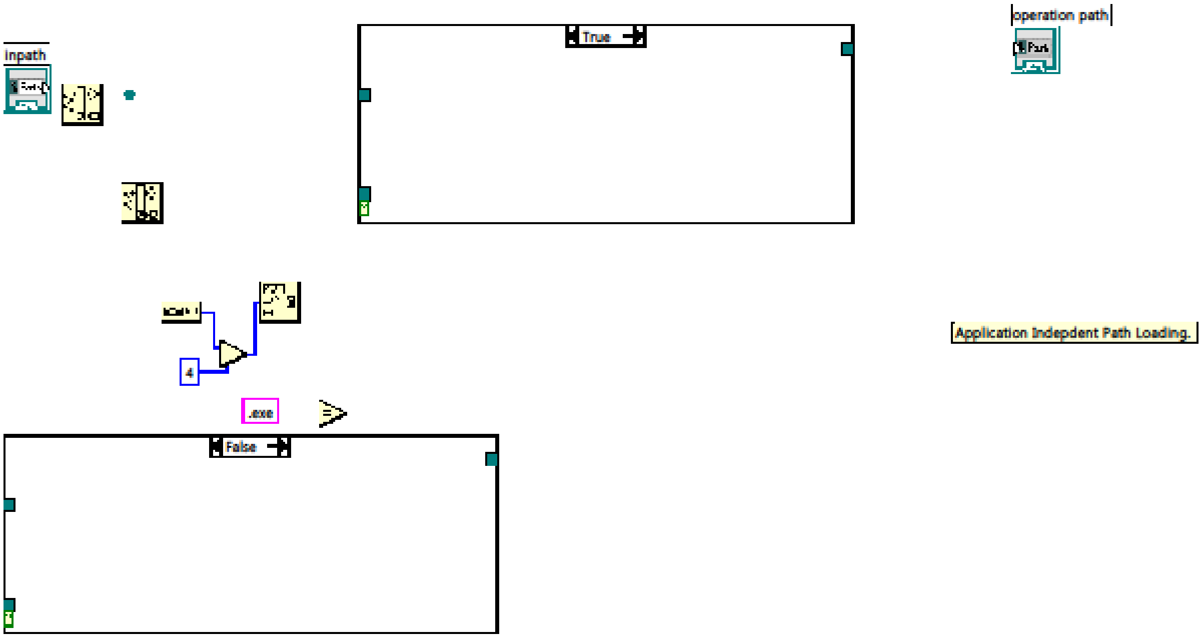
Page 1

oppath.vi
C:\Users\Steve\Desktop\VisualEyes 2020\VI\oppath.vi
Last modified on 2/12/2014 at 12:52 PM
Printed on 11/15/2014 at 3:51 PM

oppath.vi

inpath operation path



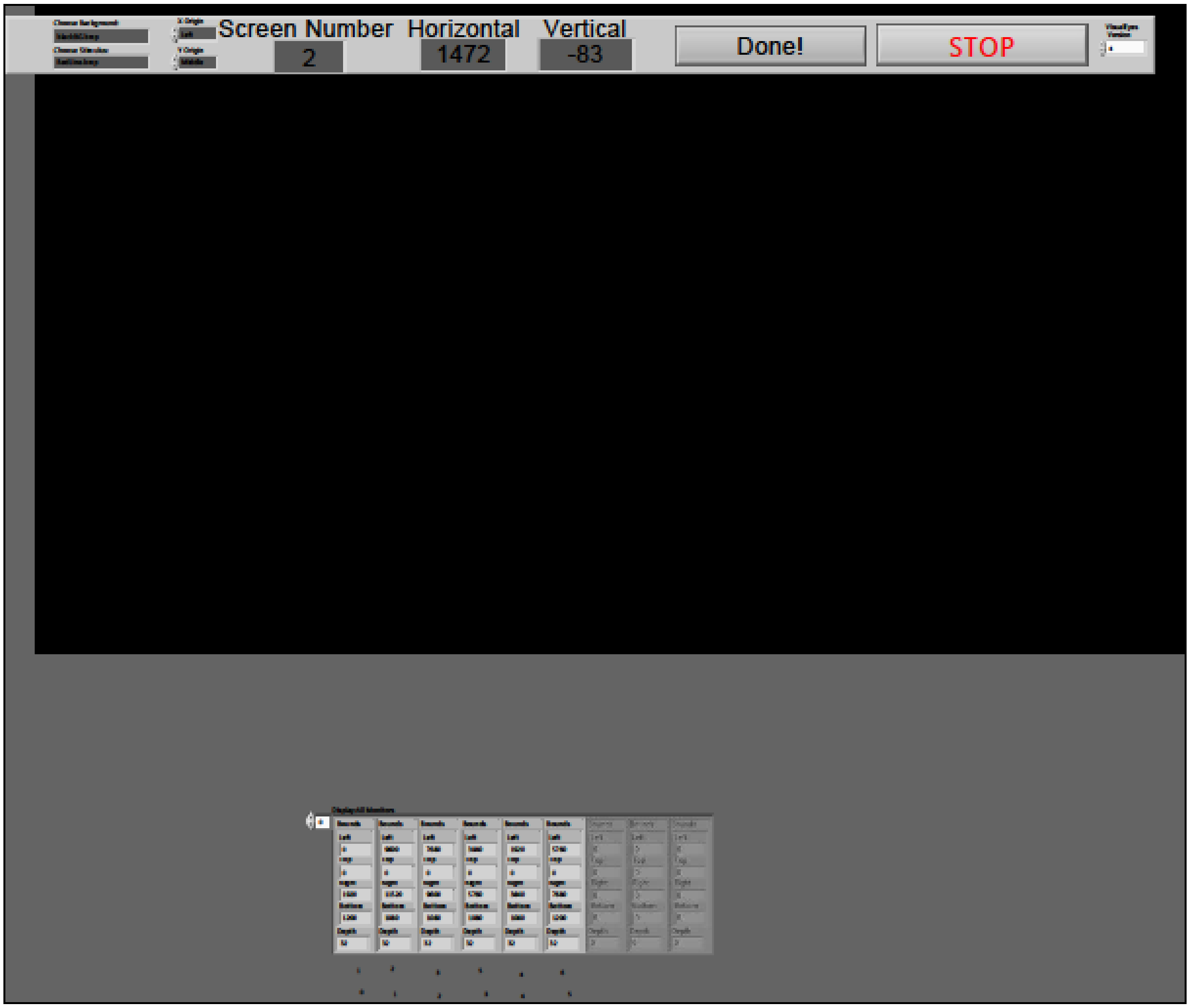


Application Indepdent Path Loading.

Page 1

Pix2Deg2020.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Pix2Deg2020.vi
Last modified on 8/18/2014 at 3:47 PM
Printed on 11/15/2014 at 3:51 PM

Pix2Deg2020.vi

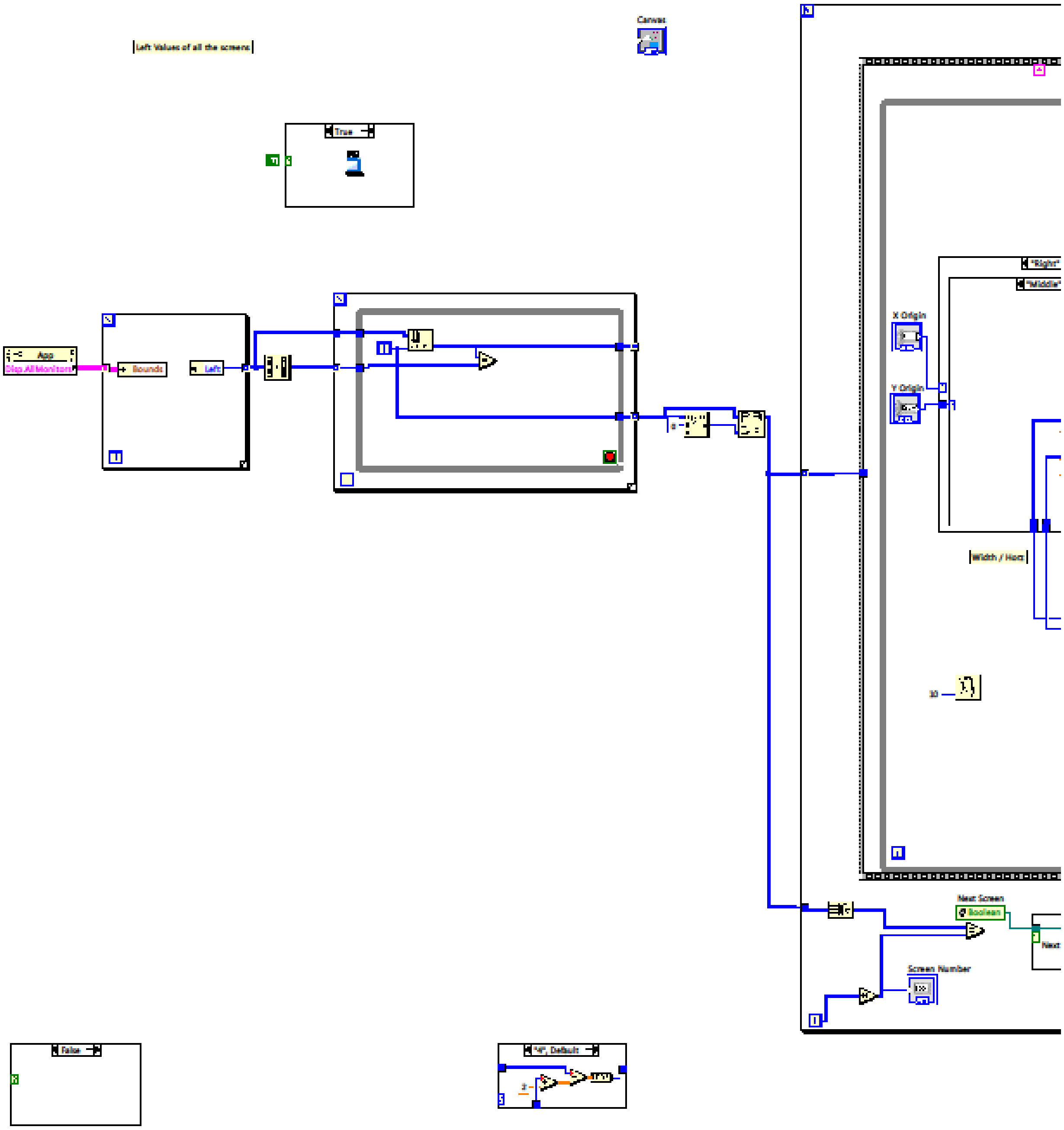


Screen Number: 2

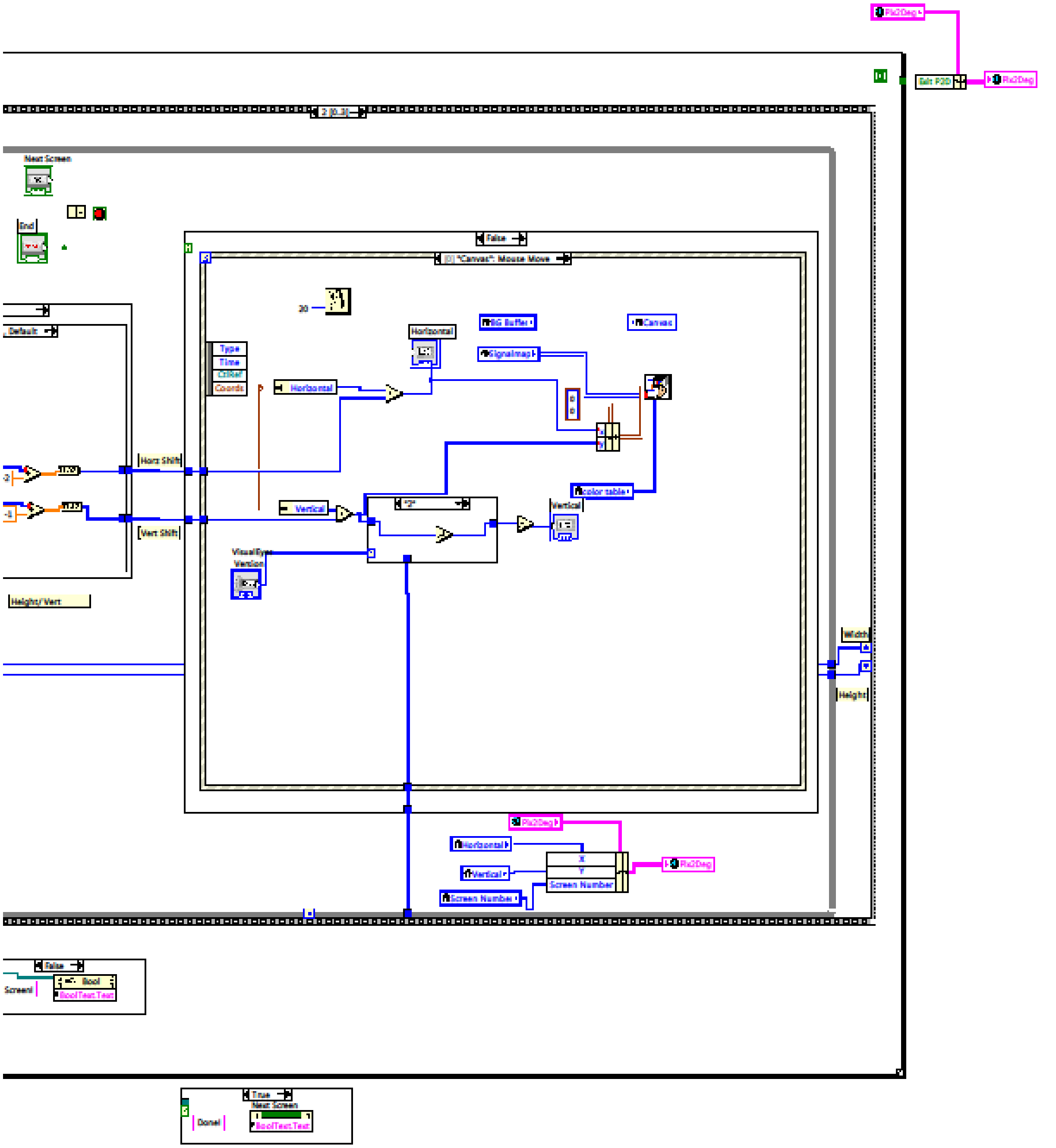
Horizontal: 1472

Vertical: -83

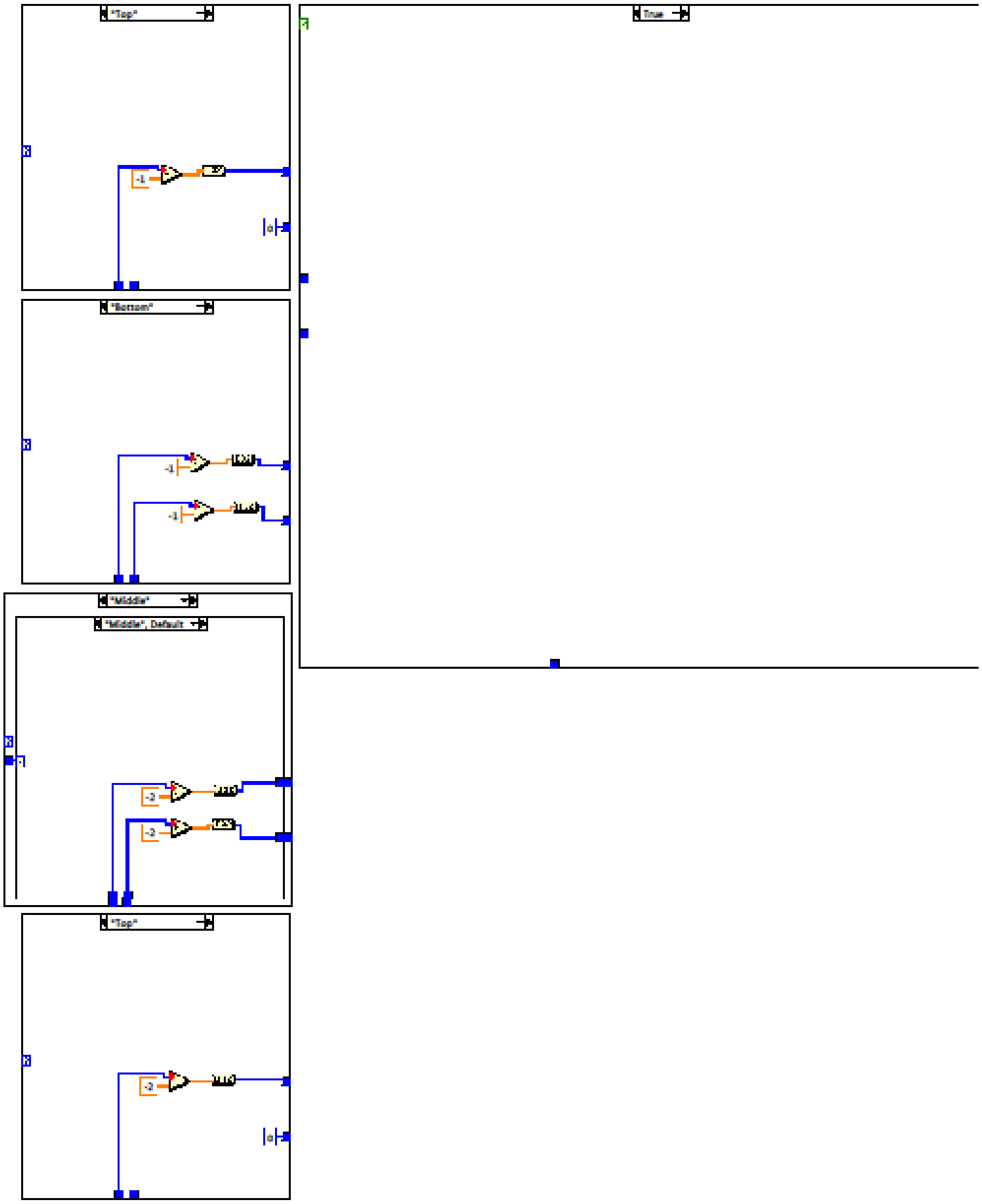
Pix2Deg2020.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Pix2Deg2020.vi
Last modified on 8/18/2014 at 3:47 PM
Printed on 11/15/2014 at 3:51 PM



Pix2Deg2020.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Pix2Deg2020.vi
Last modified on 8/18/2014 at 3:47 PM
Printed on 11/15/2014 at 3:51 PM

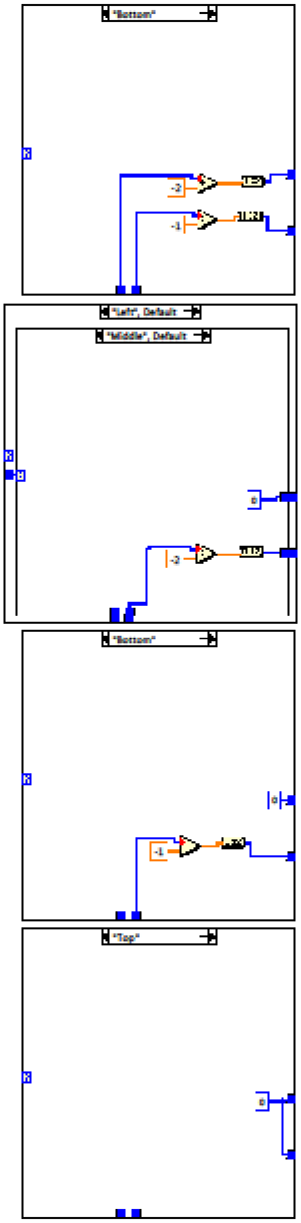


Pix2Deg2020.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Pix2Deg2020.vi
Last modified on 8/18/2014 at 3:47 PM
Printed on 11/15/2014 at 3:51 PM

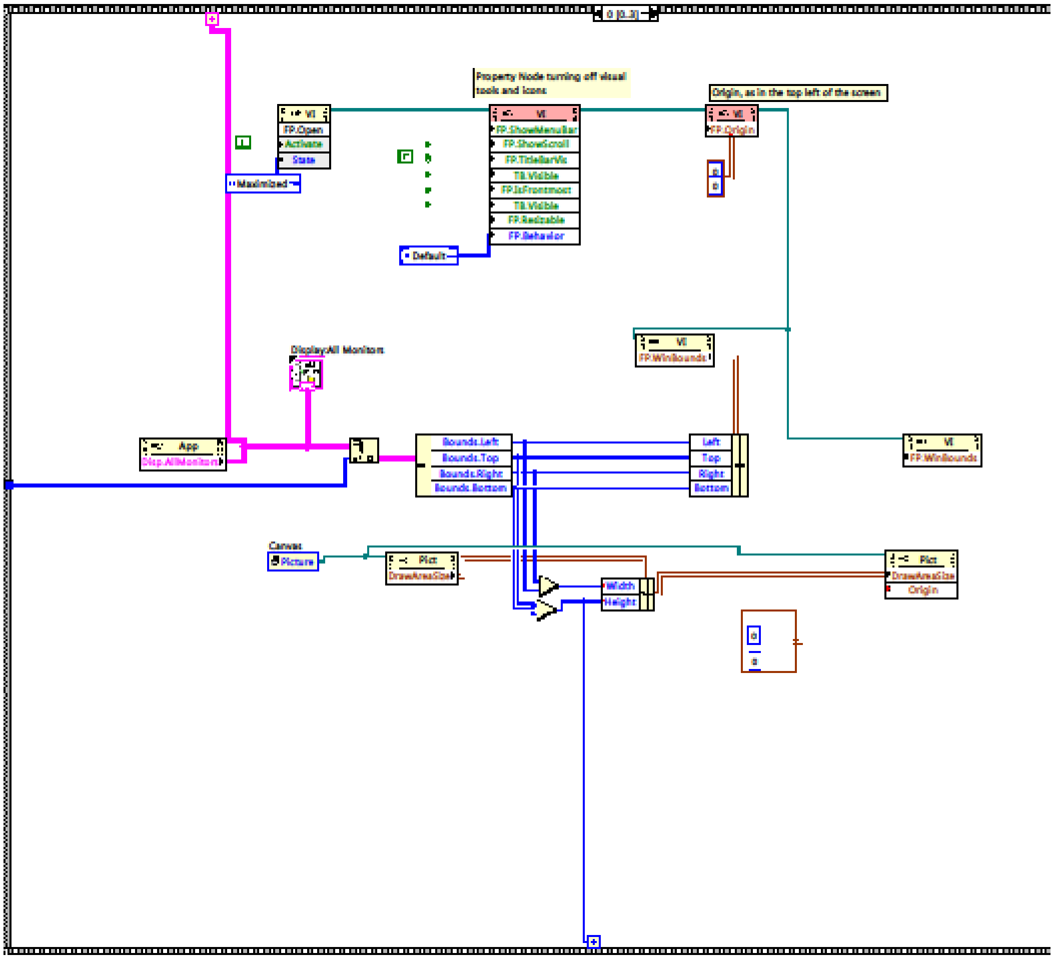


Pix2Deg2020.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Pix2Deg2020.vi
Last modified on 8/18/2014 at 3:47 PM
Printed on 11/15/2014 at 3:51 PM

Pix2Deg2020.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Pix2Deg2020.vi
Last modified on 8/18/2014 at 3:47 PM
Printed on 11/15/2014 at 3:51 PM



Pix2Deg2020.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Pix2Deg2020.vi
Last modified on 8/18/2014 at 3:47 PM
Printed on 11/15/2014 at 3:51 PM

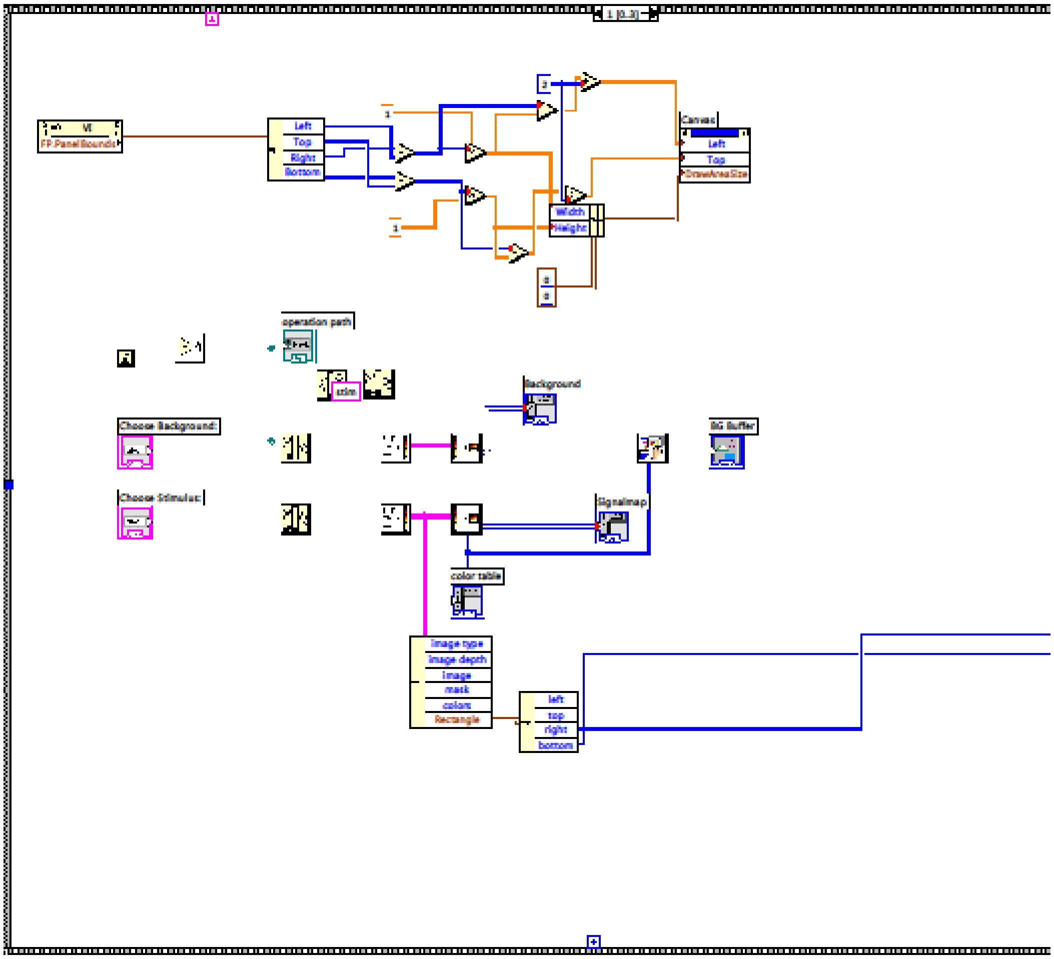


Property Node turning off visual tools and icons

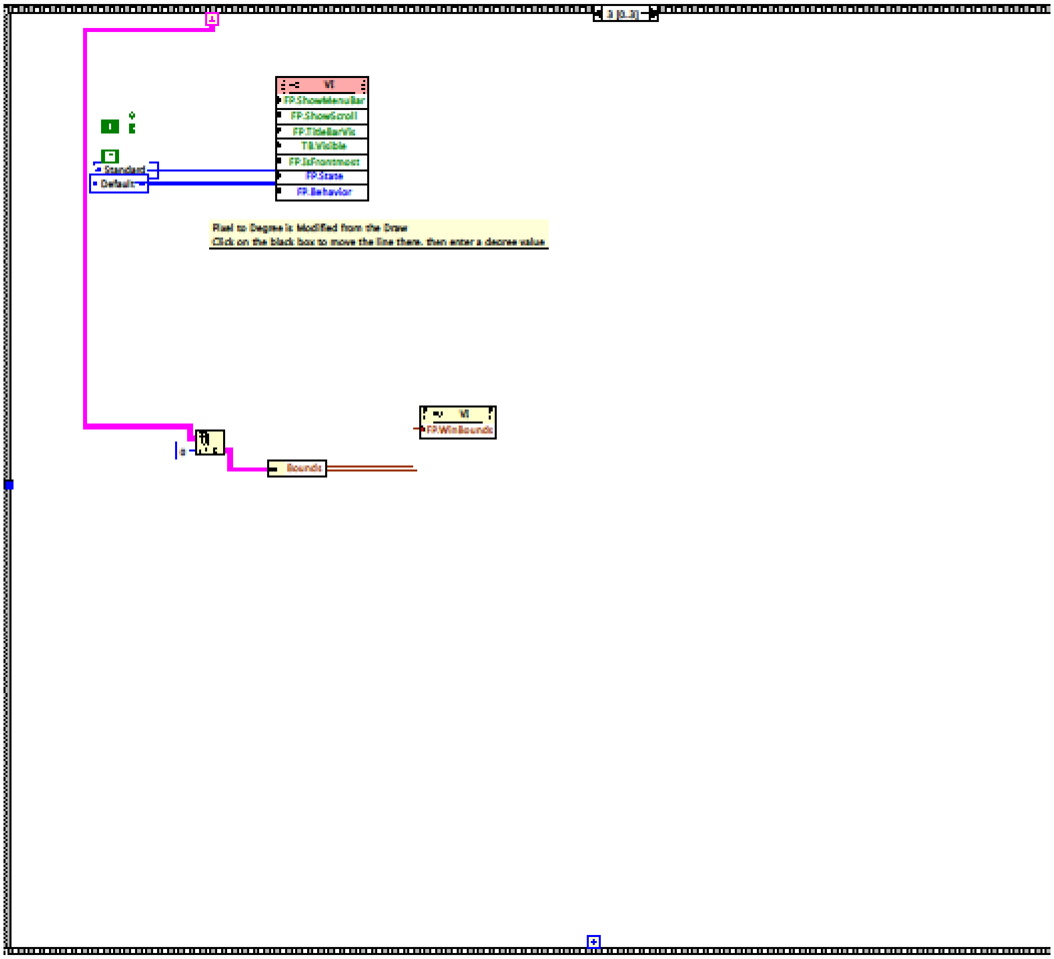
Origin, as in the top left of the screen

Page 8

Pix2Deg2020.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Pix2Deg2020.vi
Last modified on 8/18/2014 at 3:47 PM
Printed on 11/15/2014 at 3:51 PM


1 [0..3]
VI
FP.PanelBounds
Left
Top
Right
Bottom
2
1
1
Canvas
Left
Top
DrawAreaSize
Width
Height
0
0
operation path
stim
Background
Choose Background:
Choose Stimulus:
BG Buffer
Signalmap
color table
image type
image depth
image
mask
colors
Rectangle
left
top
right
bottom

Pix2Deg2020.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Pix2Deg2020.vi
Last modified on 8/18/2014 at 3:47 PM
Printed on 11/15/2014 at 3:51 PM

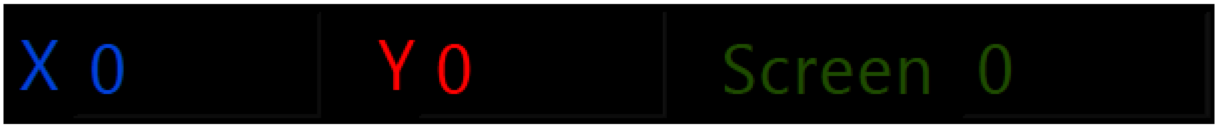


Pixel to Degree is Modified from the Draw
Click on the black box to move the line there. then enter a degree value

Page 1

Pix2DegControlScreen.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Pix2DegControlScreen.vi
Last modified on 8/18/2014 at 3:47 PM
Printed on 11/15/2014 at 3:51 PM

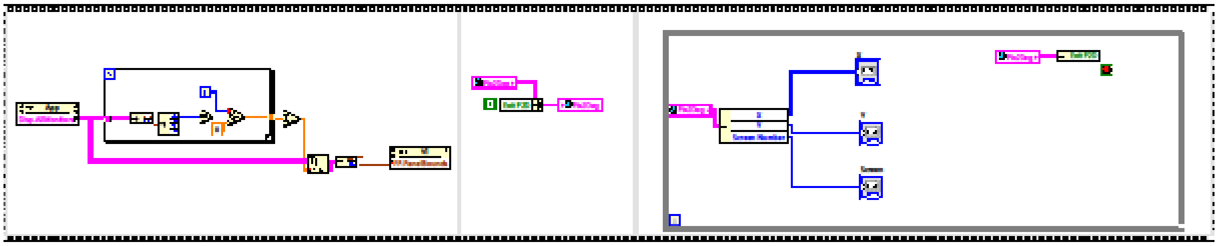
Pix2DegControlScreen.vi



X 0

Y 0

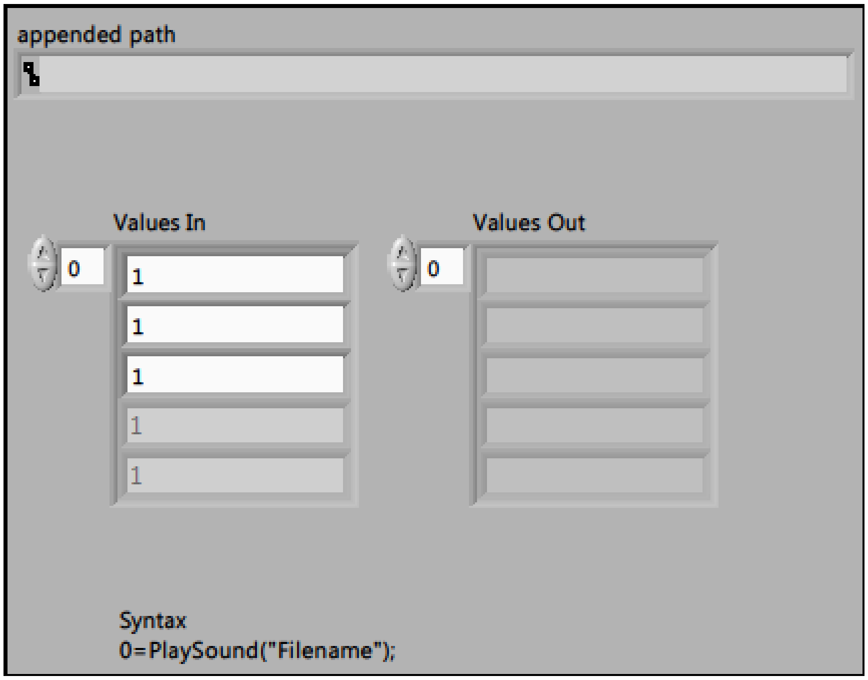
Screen 0



PlaySound.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\PlaySound.vi
Last modified on 2/26/2014 at 3:43 PM
Printed on 11/15/2014 at 3:51 PM

PlaySound.vi

Values In   Values Out

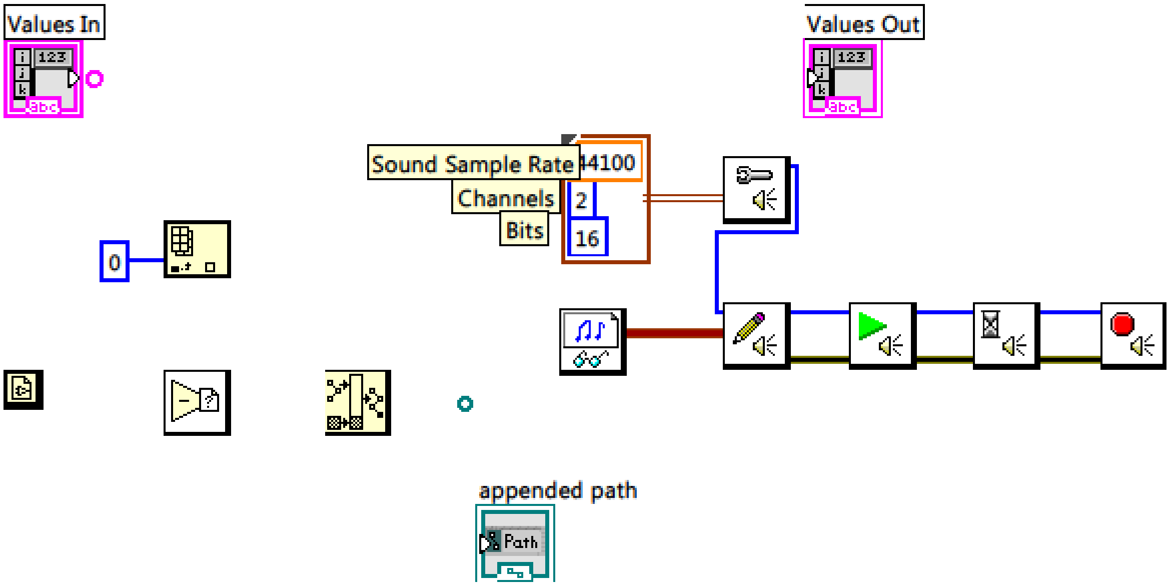


appended path

Values In: 0 | 1, 1, 1, 1, 1

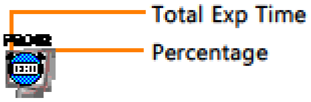
Values Out: 0

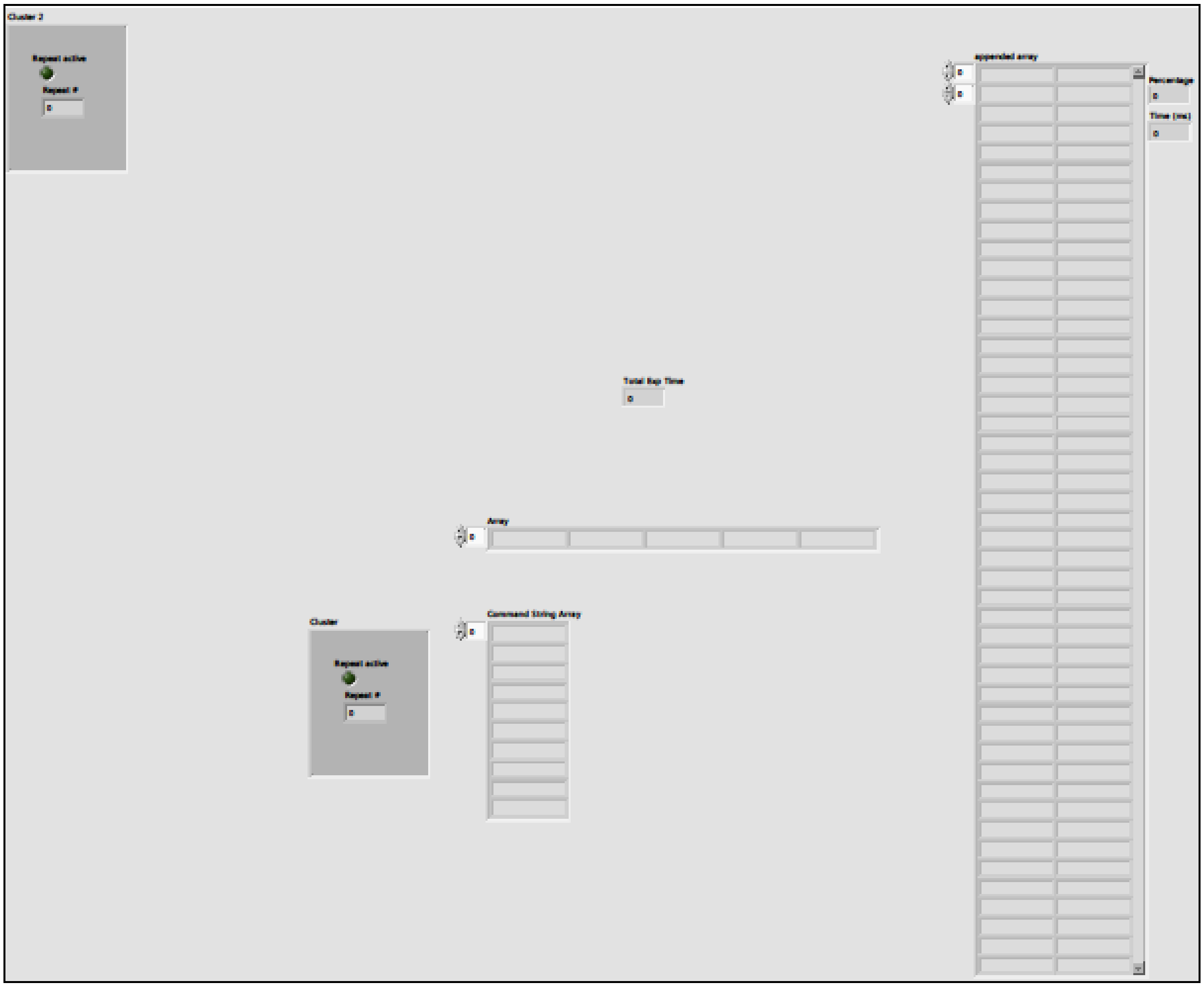
Syntax
0=PlaySound("Filename");



ProgBar2.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ProgBar2.vi
Last modified on 8/22/2014 at 12:23 PM
Printed on 11/15/2014 at 3:51 PM

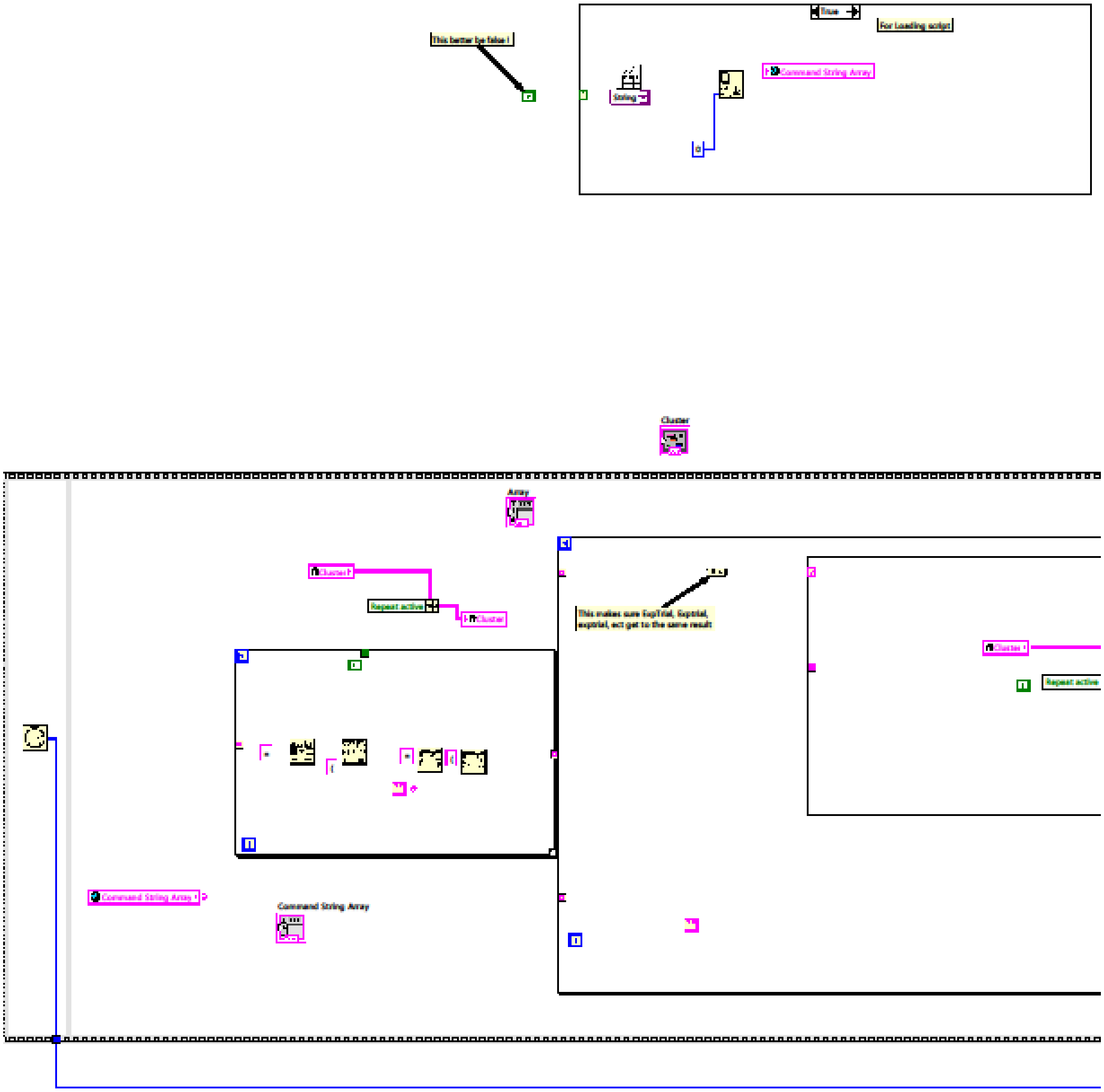
ProgBar2.vi





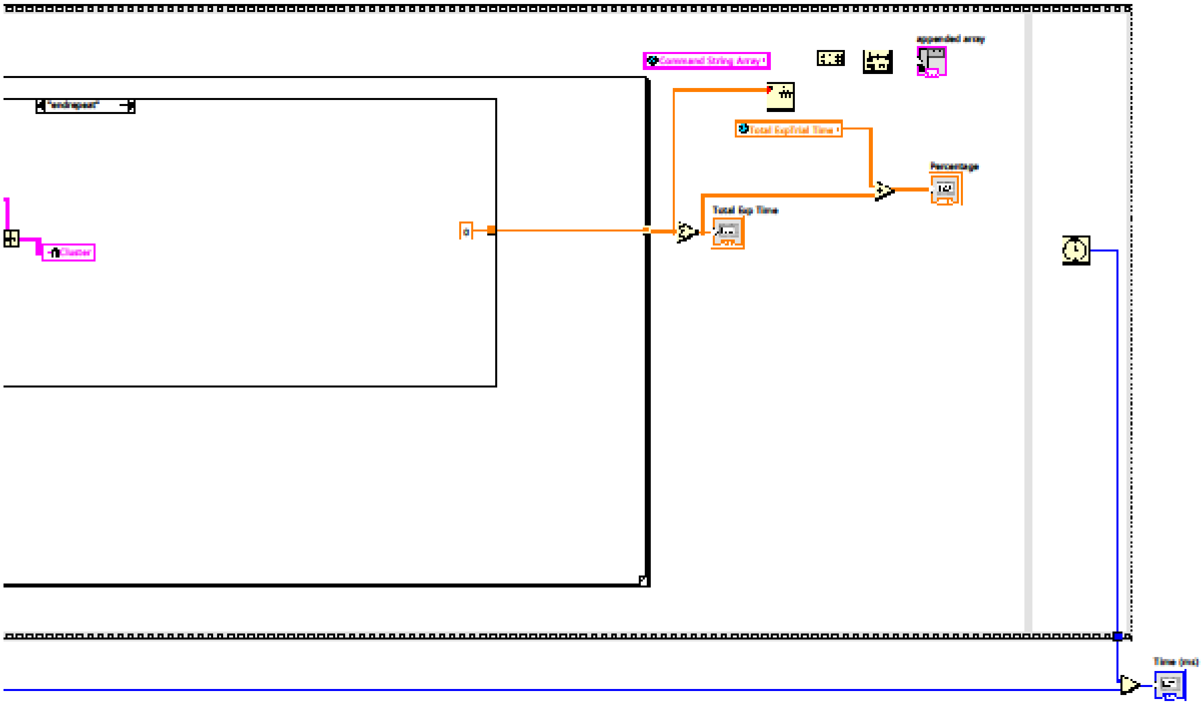
Page 2

ProgBar2.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ProgBar2.vi
Last modified on 8/22/2014 at 12:23 PM
Printed on 11/15/2014 at 3:51 PM

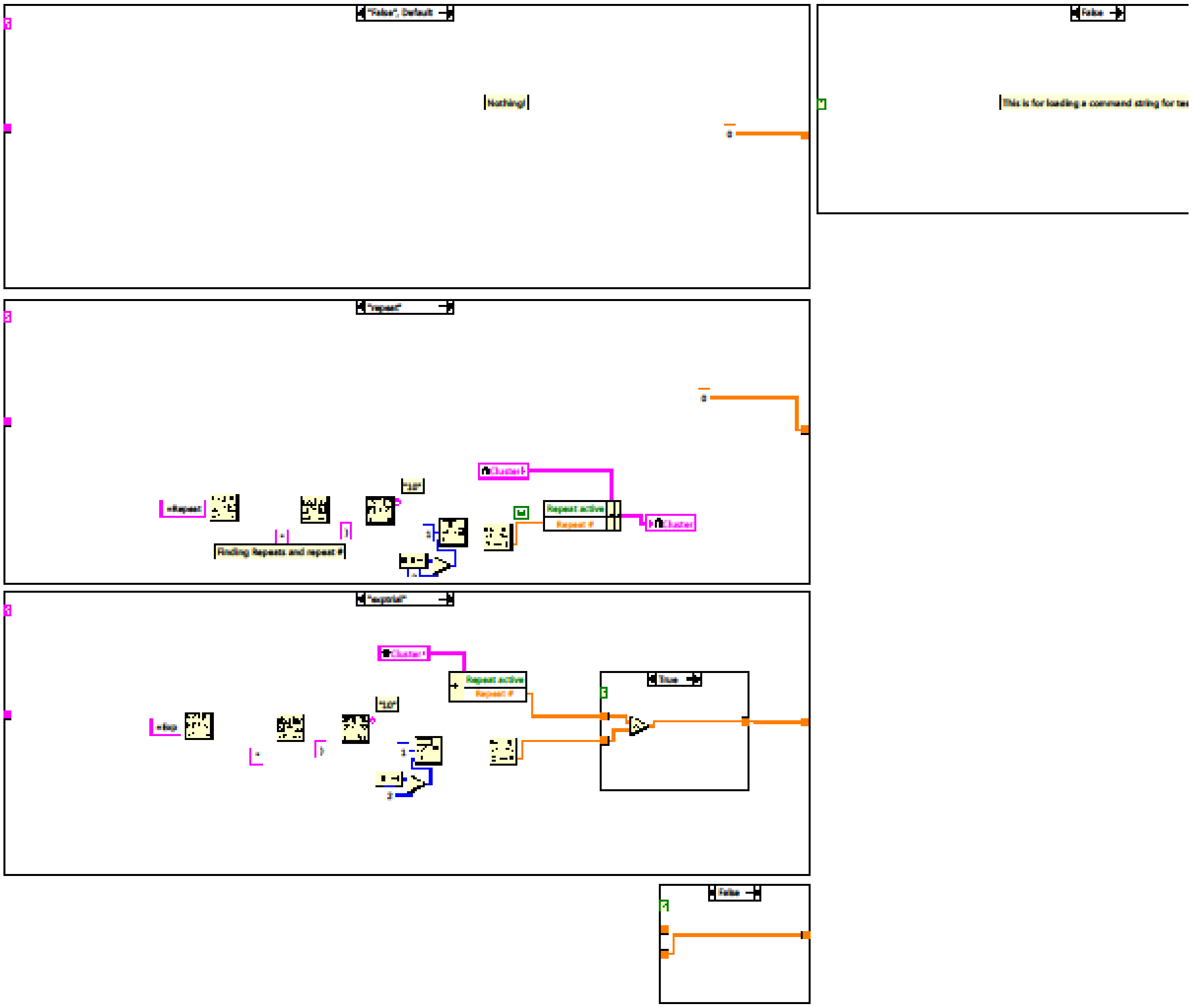


Page 3

ProgBar2.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ProgBar2.vi
Last modified on 8/22/2014 at 12:23 PM
Printed on 11/15/2014 at 3:51 PM



ProgBar2.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ProgBar2.vi
Last modified on 8/22/2014 at 12:23 PM
Printed on 11/15/2014 at 3:51 PM

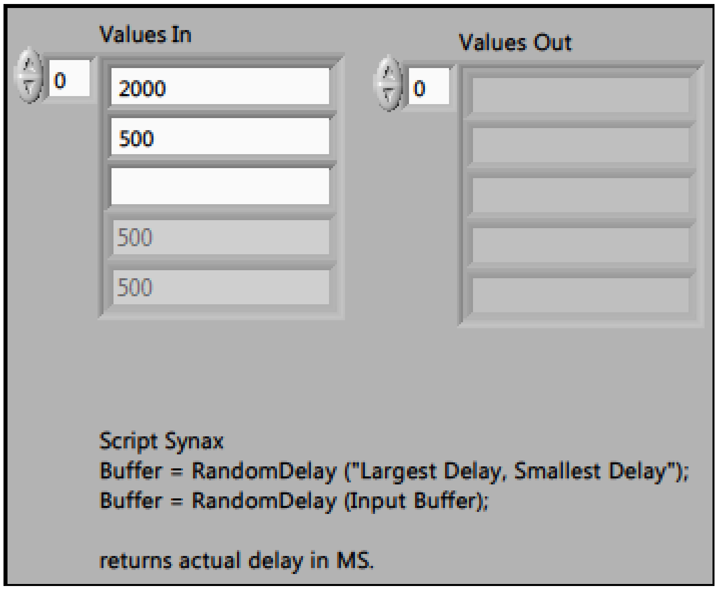


ProgBar2.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ProgBar2.vi
Last modified on 8/22/2014 at 12:23 PM
Printed on 11/15/2014 at 3:51 PM

RandomDelay.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\RandomDelay.vi
Last modified on 2/27/2014 at 1:01 PM
Printed on 11/15/2014 at 3:51 PM

RandomDelay.vi

Values In Values Out



Values In

0

2000

500

500

500

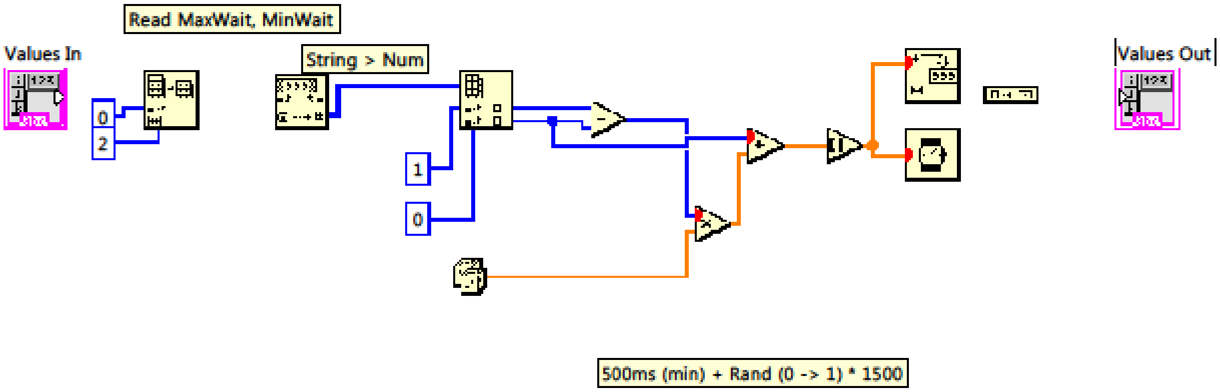
Values Out

0

Script Synax
Buffer = RandomDelay ("Largest Delay, Smallest Delay");
Buffer = RandomDelay (Input Buffer);

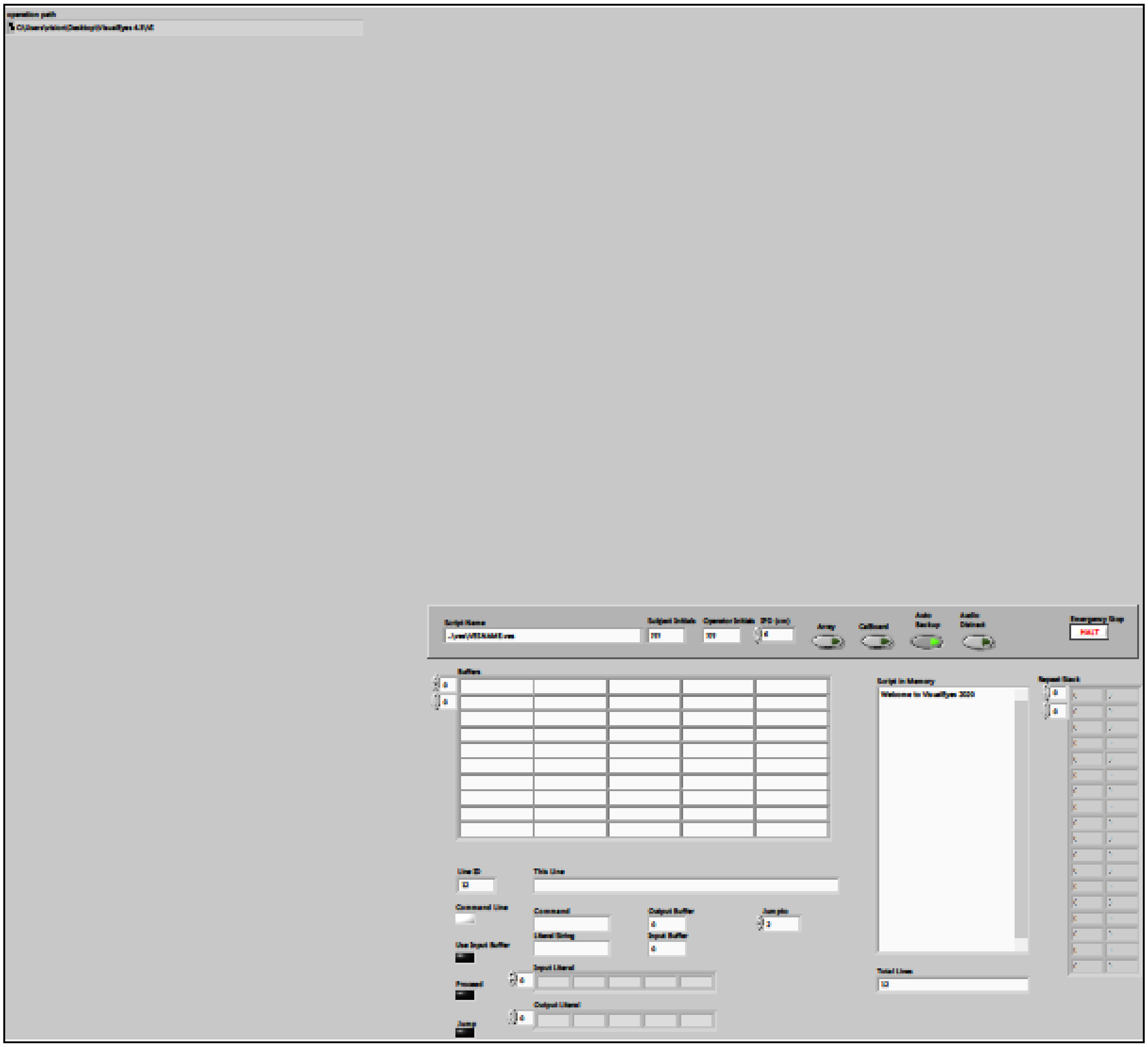
returns actual delay in MS.

Syntax:
BufferNumber=RandomDelay("MinDelay,MaxDelay")
Returns Actual amount of time delayed



ReadScript.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ReadScript.vi
Last modified on 8/22/2014 at 2:16 PM
Printed on 11/15/2014 at 3:51 PM

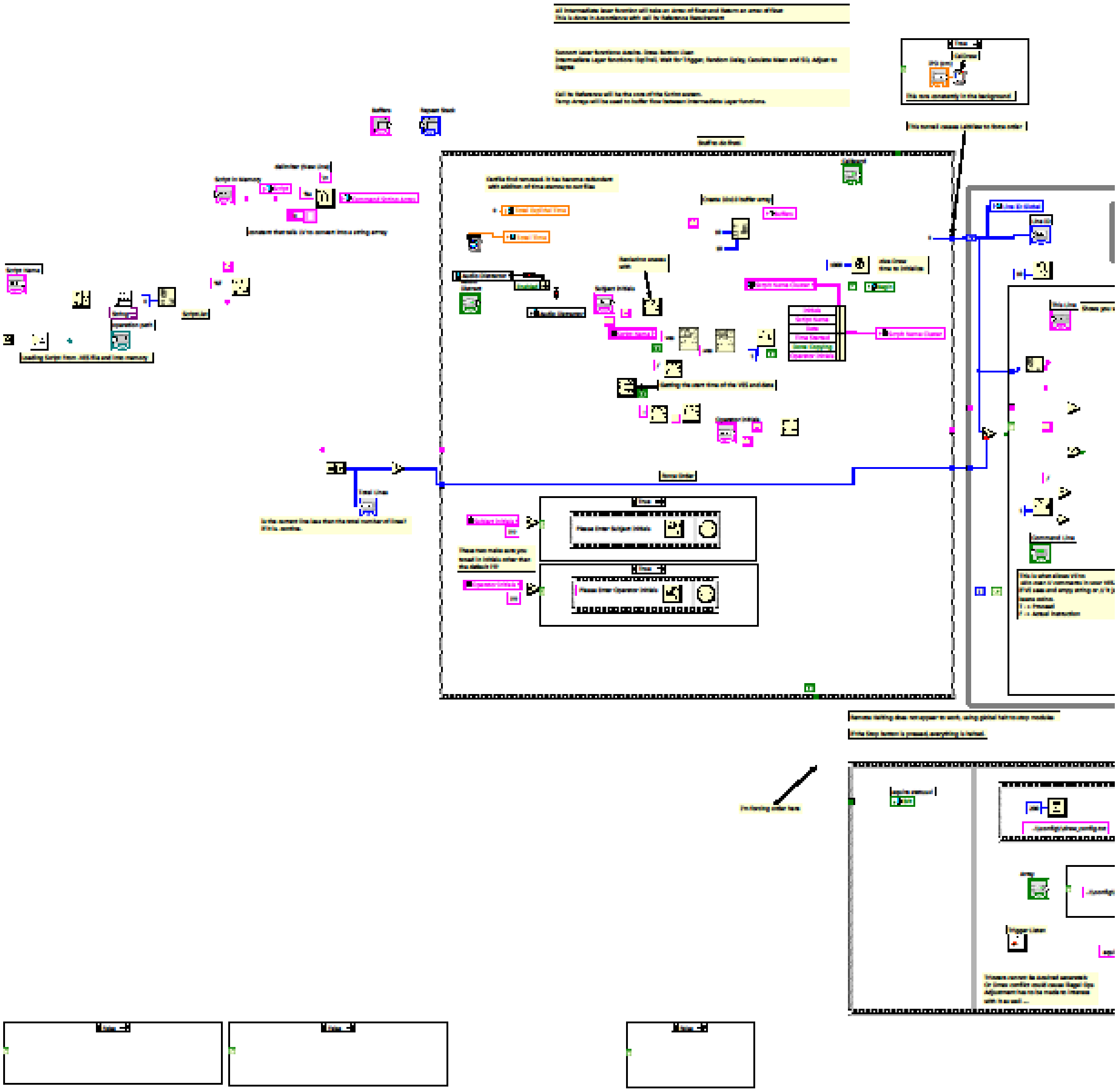
ReadScript.vi



Welcome to VisualEyes 2020

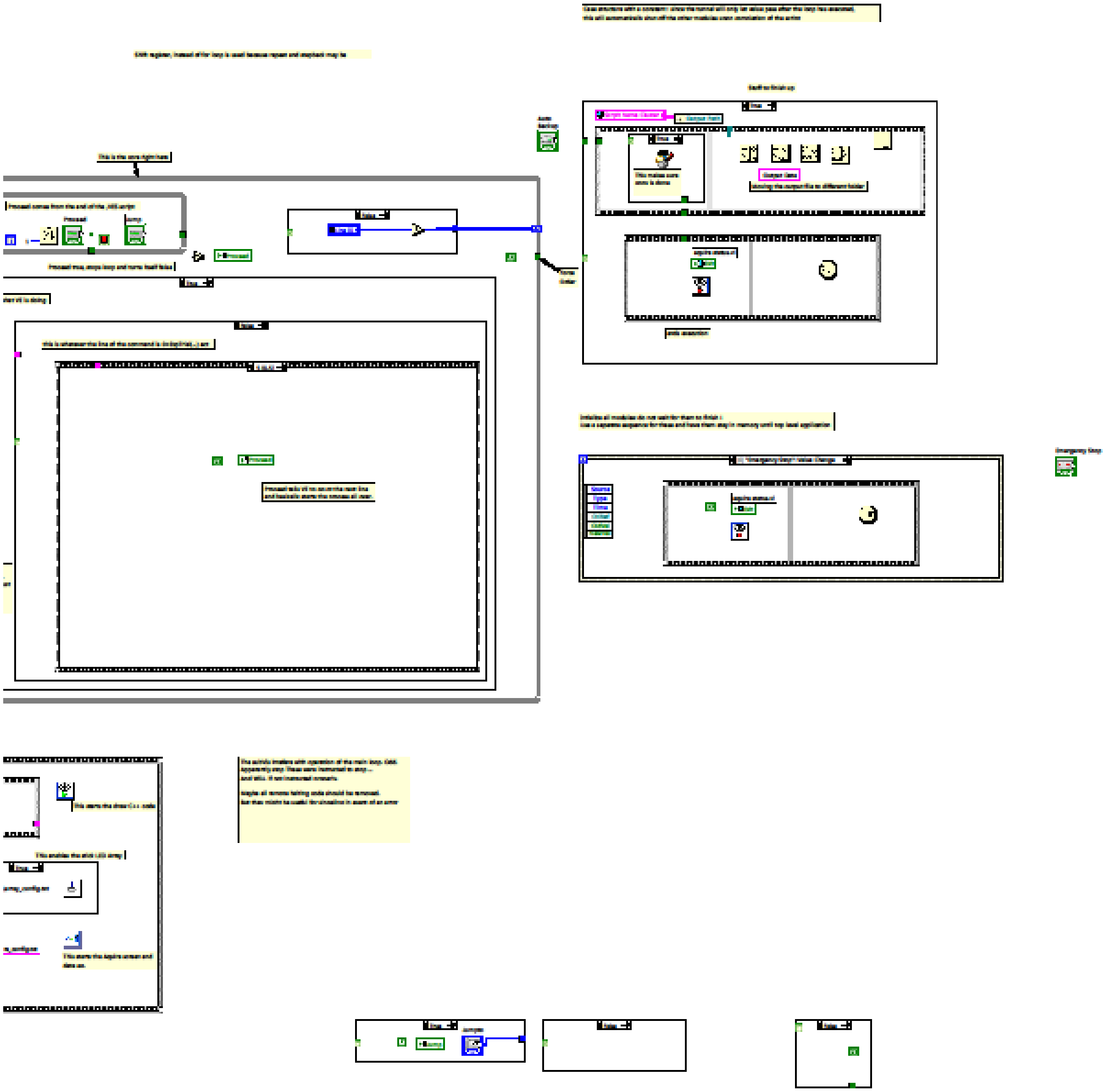
Page 2

ReadScript.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ReadScript.vi
Last modified on 8/22/2014 at 2:16 PM
Printed on 11/15/2014 at 3:51 PM



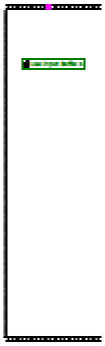
Page 3

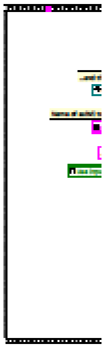
ReadScript.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ReadScript.vi
Last modified on 8/22/2014 at 2:16 PM
Printed on 11/15/2014 at 3:51 PM



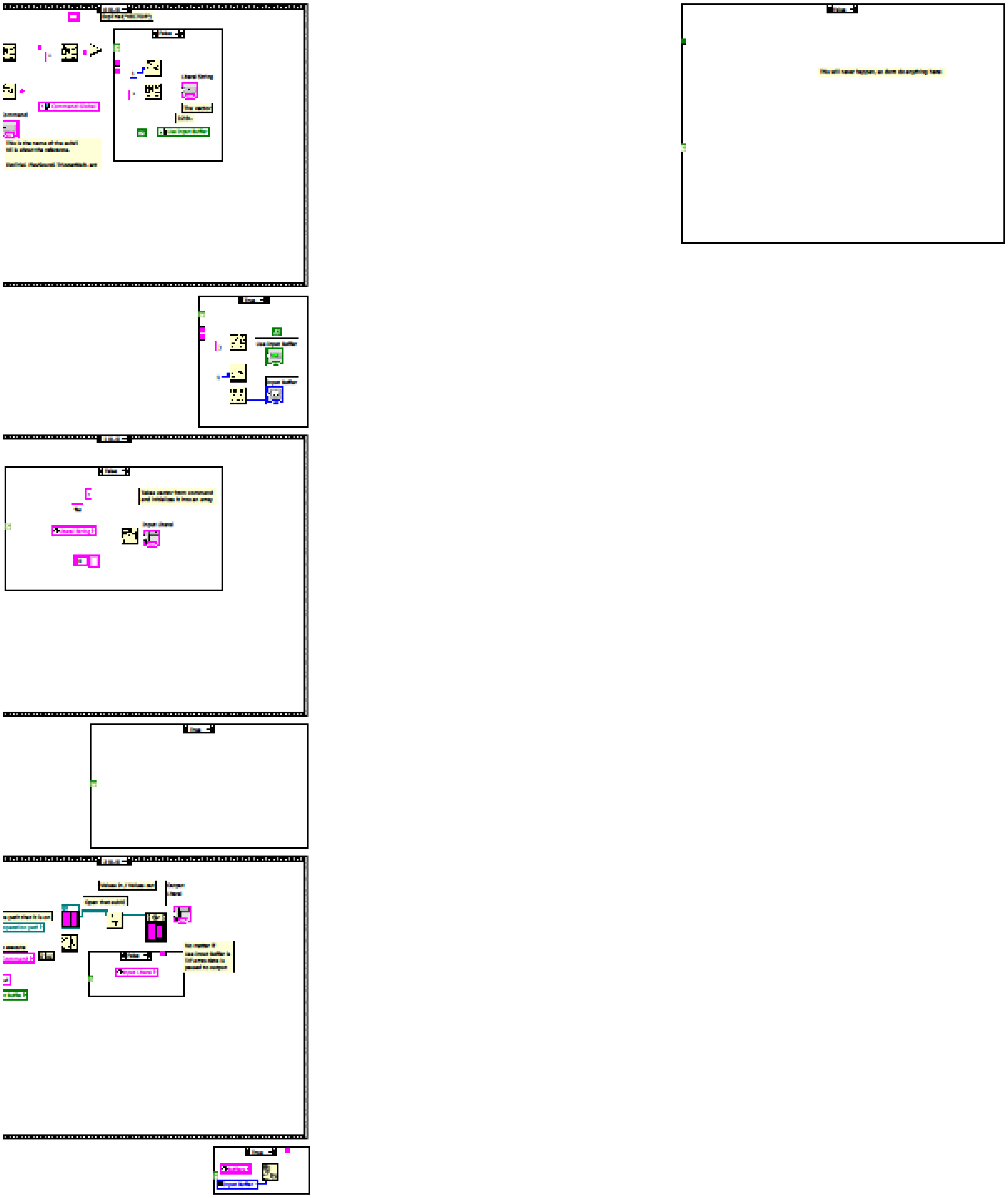
Page 4

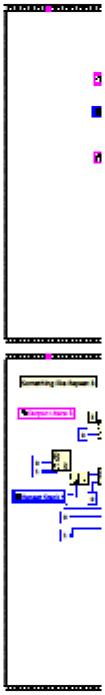
ReadScript.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ReadScript.vi
Last modified on 8/22/2014 at 2:16 PM
Printed on 11/15/2014 at 3:51 PM



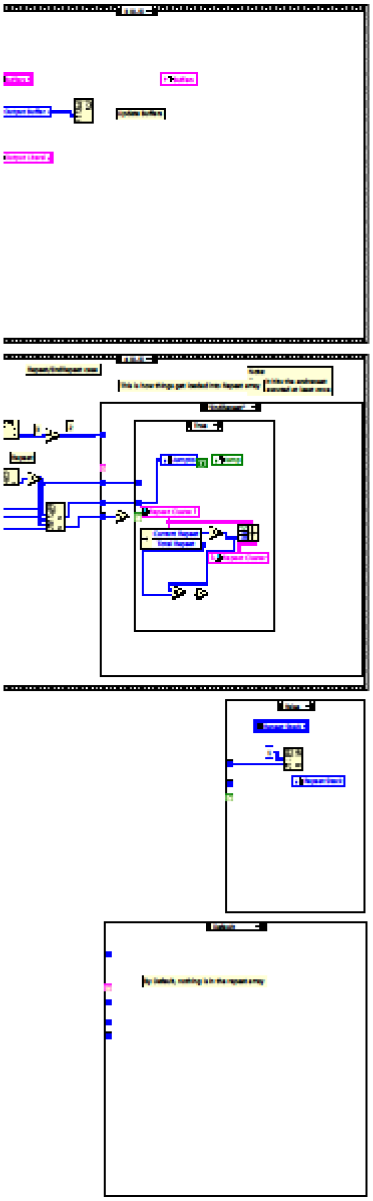


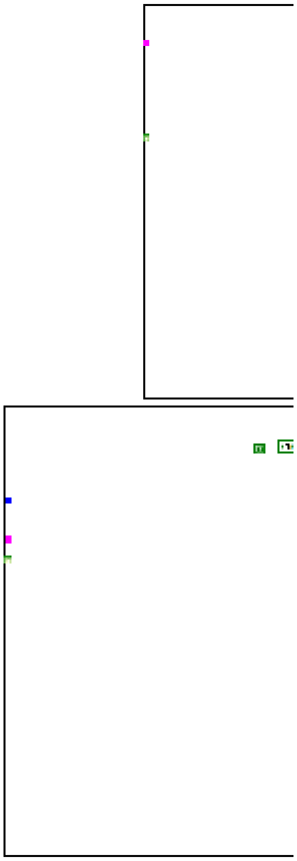
ReadScript.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ReadScript.vi
Last modified on 8/22/2014 at 2:16 PM
Printed on 11/15/2014 at 3:51 PM



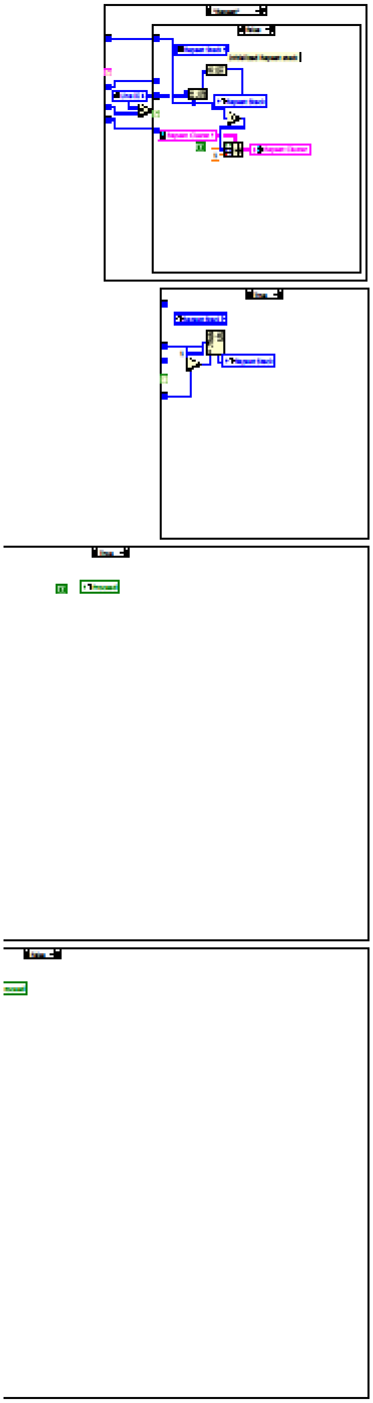


ReadScript.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ReadScript.vi
Last modified on 8/22/2014 at 2:16 PM
Printed on 11/15/2014 at 3:51 PM


Repeat/EndRepeat case
This is how things get loaded into Repeat array
Note:
EndRepeat
True
Repeat
Current Repeat
Total Repeat
Repeat Count
By Default, nothing is in the repeat array



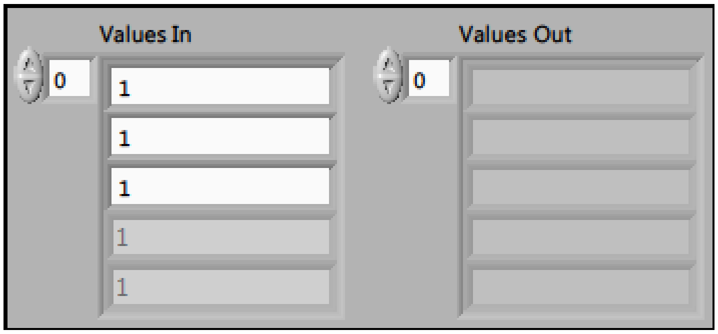
ReadScript.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\ReadScript.vi
Last modified on 8/22/2014 at 2:16 PM
Printed on 11/15/2014 at 3:51 PM



Repeat.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Repeat.vi
Last modified on 7/18/2014 at 4:23 PM
Printed on 11/15/2014 at 3:51 PM

Repeat.vi

Values In Values Out



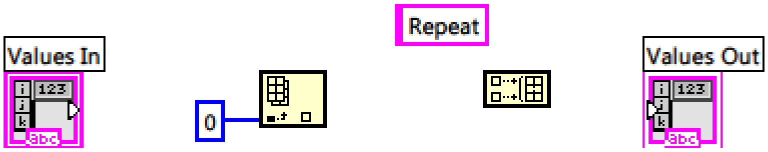
Values In

0

1
1
1
1
1

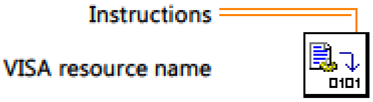
Values Out

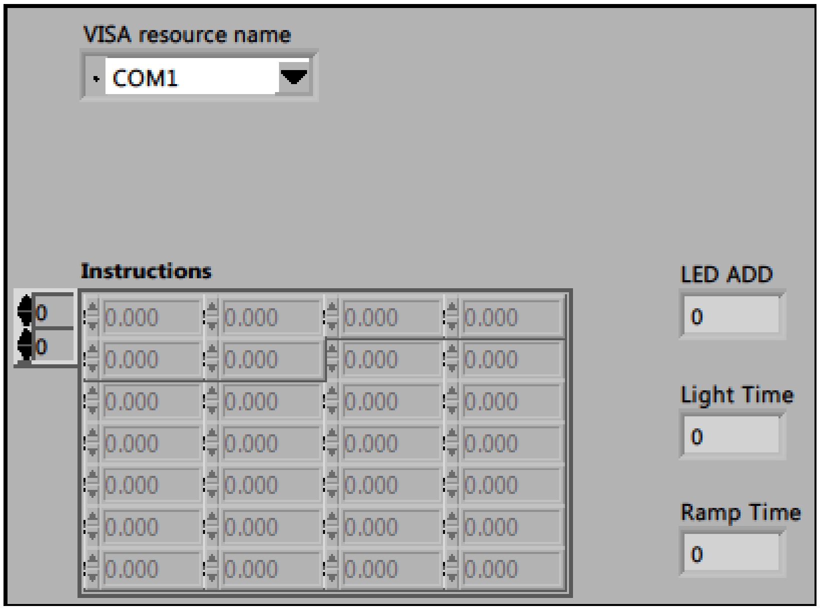
0



Send Instructions.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Send Instructions.vi
Last modified on 8/12/2014 at 1:21 PM
Printed on 11/15/2014 at 3:51 PM

Send Instructions.vi





VISA resource name
COM1

Instructions

0
0

0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000

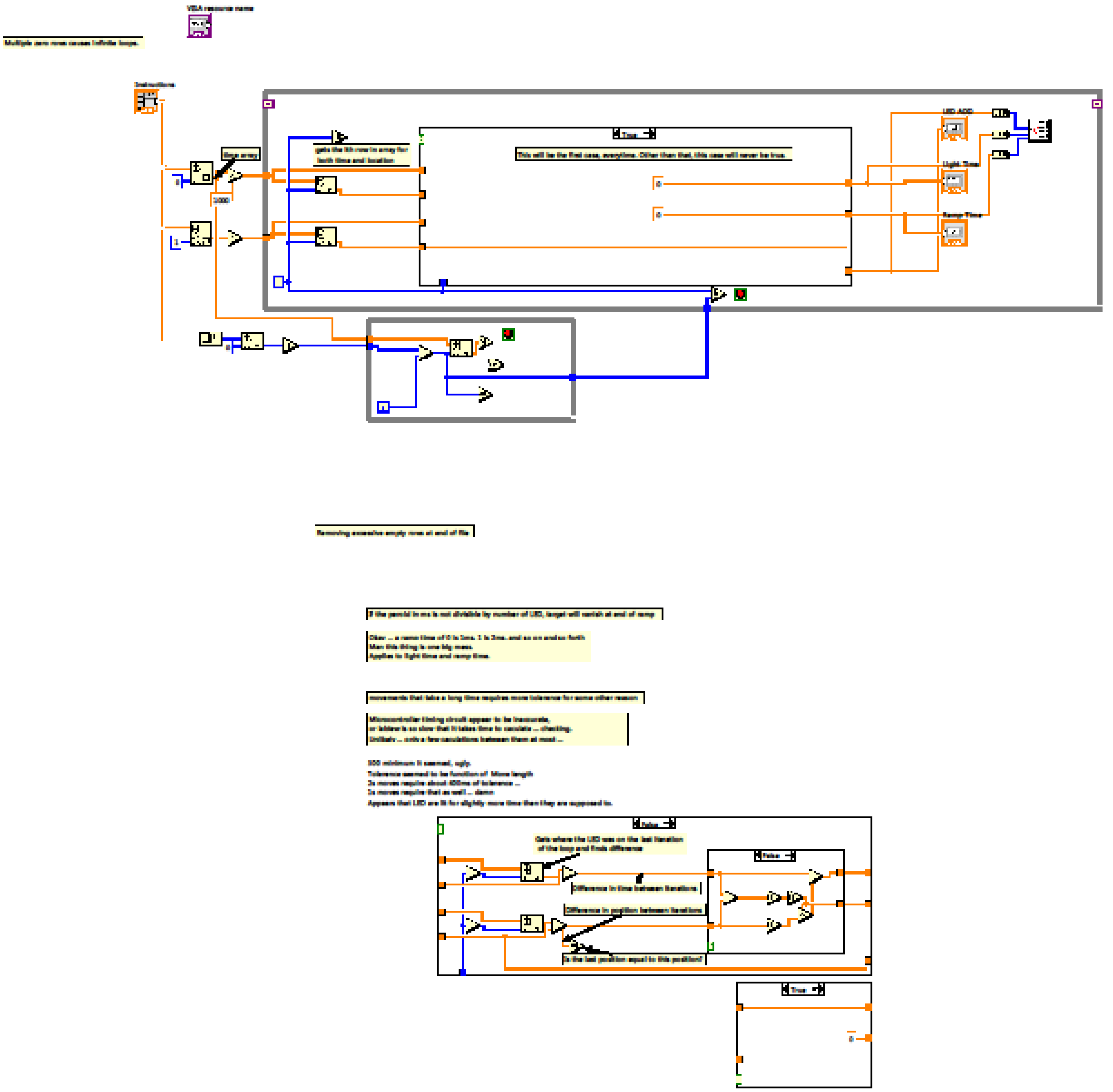
LED ADD
0

Light Time
0

Ramp Time
0

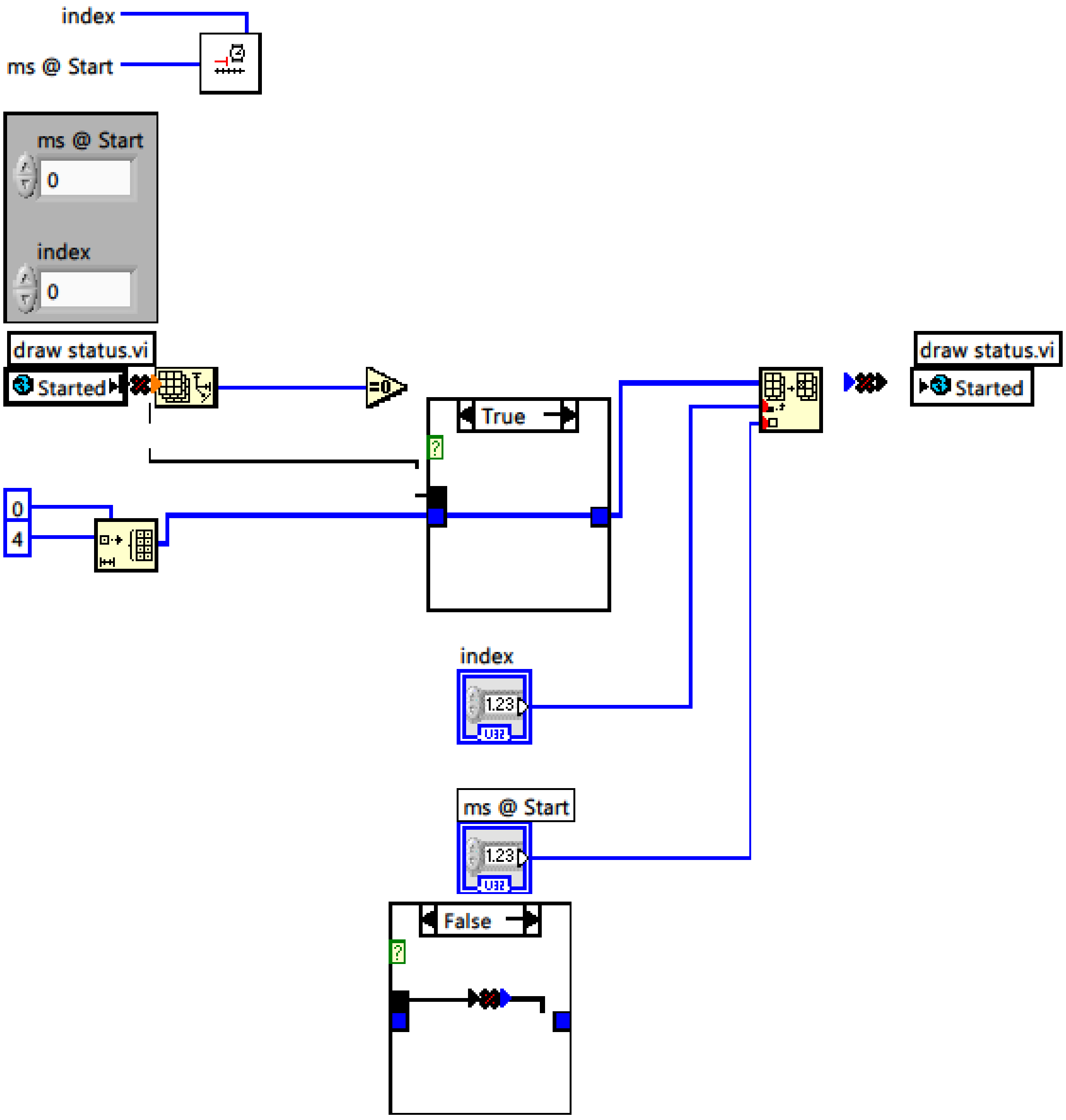
Page 2

Send Instructions.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Send Instructions.vi
Last modified on 8/12/2014 at 1:21 PM
Printed on 11/15/2014 at 3:51 PM


VISA resource name
Multiple zero rows causes infinite loops.
Instructions
Base array
1000
0
1
gets the ith row in array for both time and location
True
This will be the first case, everytime. Other than that, this case will never be true.
0
0
LED ADD
Light Time
Ramp Time
Removing excessive empty rows at end of file
If the period in ms is not divisible by number of LED, target will vanish at end of ramp
Okay ... a ramp time of 0 is 1ms. 1 is 2ms. and so on and so forth
Man this thing is one big mess.
Applies to light time and ramp time.
movements that take a long time requires more tolerance for some other reason
Microcontroller timing circuit appear to be inaccurate,
or labview is so slow that it takes time to calculate ... checking.
Unlikely ... only a few calculations between them at most ...
300 minimum it seemed, ugly.
Tolerance seemed to be function of Move length
2s moves require about 400ms of tolerance ...
3s moves require that as well ... damn
Appears that LED are lit for slightly more time than they are supposed to.
False
Gets where the LED was on the last iteration of the loop and finds difference
Difference in time between iterations
Difference in position between iterations
Is the last position equal to this position?
False
True
0

Set Start Time.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\Set Start Time.vi
Last modified on 2/12/2014 at 12:52 PM
Printed on 11/15/2014 at 3:51 PM

Set Start Time.vi

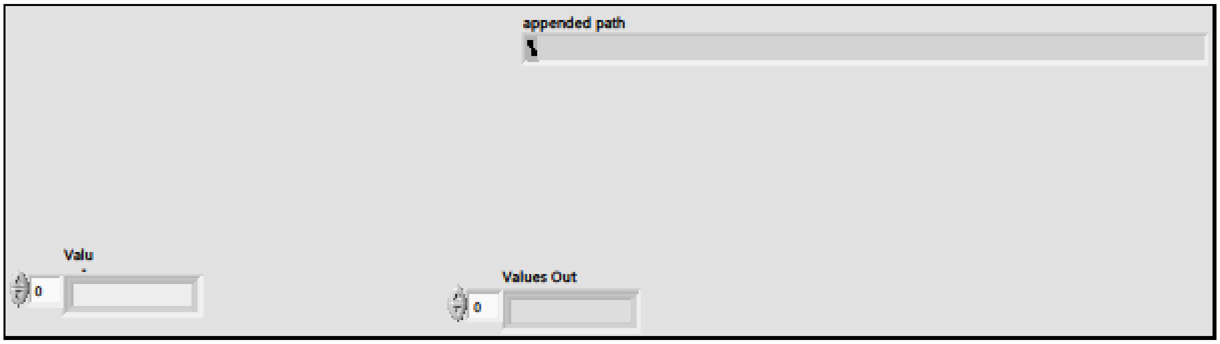


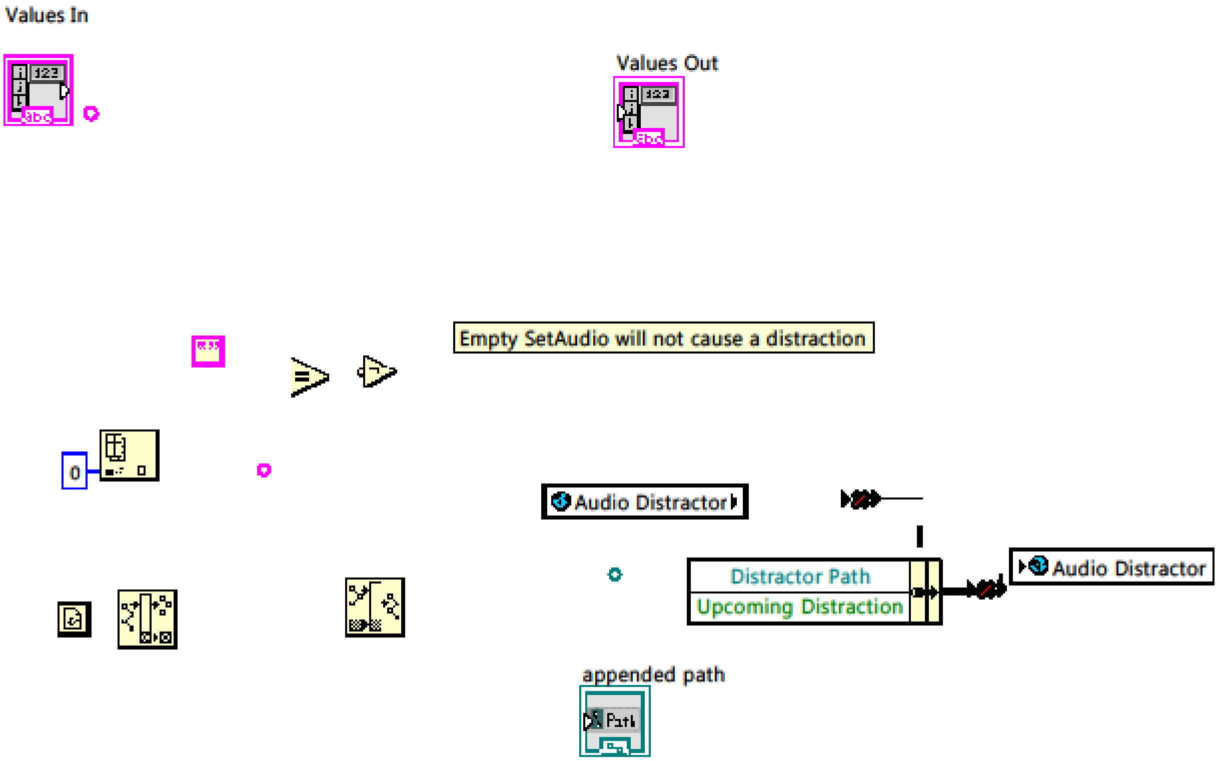
Page 1

SetAudio.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\SetAudio.vi
Last modified on 8/22/2014 at 12:06 PM
Printed on 11/15/2014 at 3:51 PM

SetAudio.vi

Values In Values Out

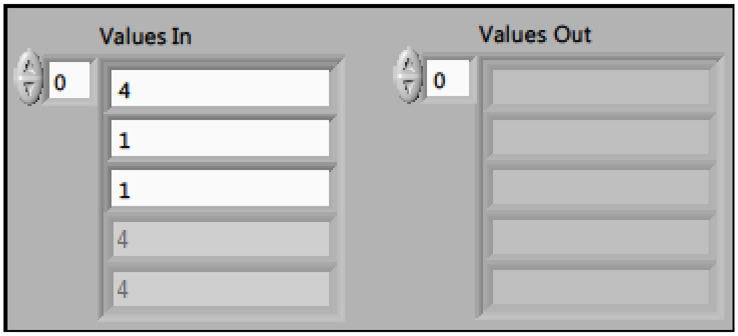


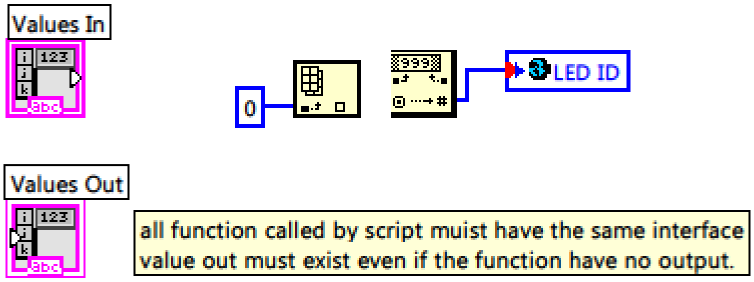


SetLED.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\SetLED.vi
Last modified on 8/12/2014 at 12:48 PM
Printed on 11/15/2014 at 3:51 PM

SetLED.vi

Values In Values Out





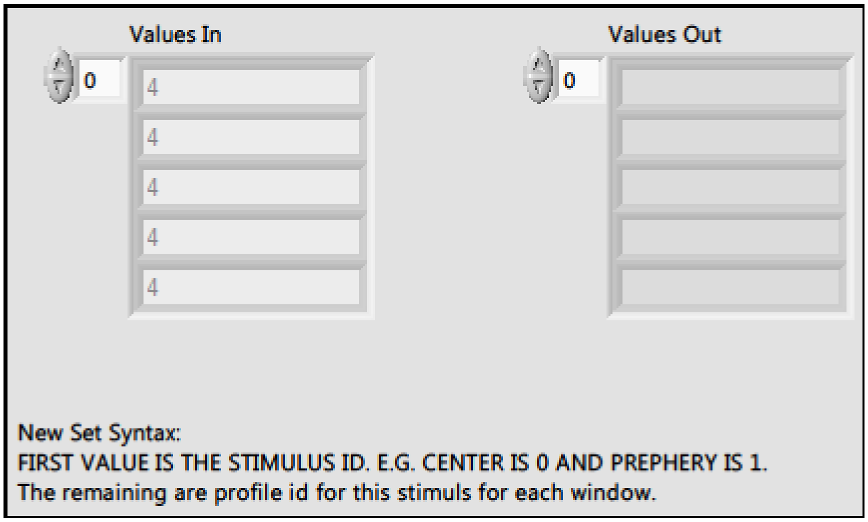
all function called by script muist have the same interface
value out must exist even if the function have no output.

SetStimulus.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\SetStimulus.vi
Last modified on 2/27/2014 at 1:50 PM
Printed on 11/15/2014 at 3:52 PM

SetStimulus.vi

Values In Values Out

Sets Profiles for stimulus. Must be called before StartDraw or ExpTrial



Values In

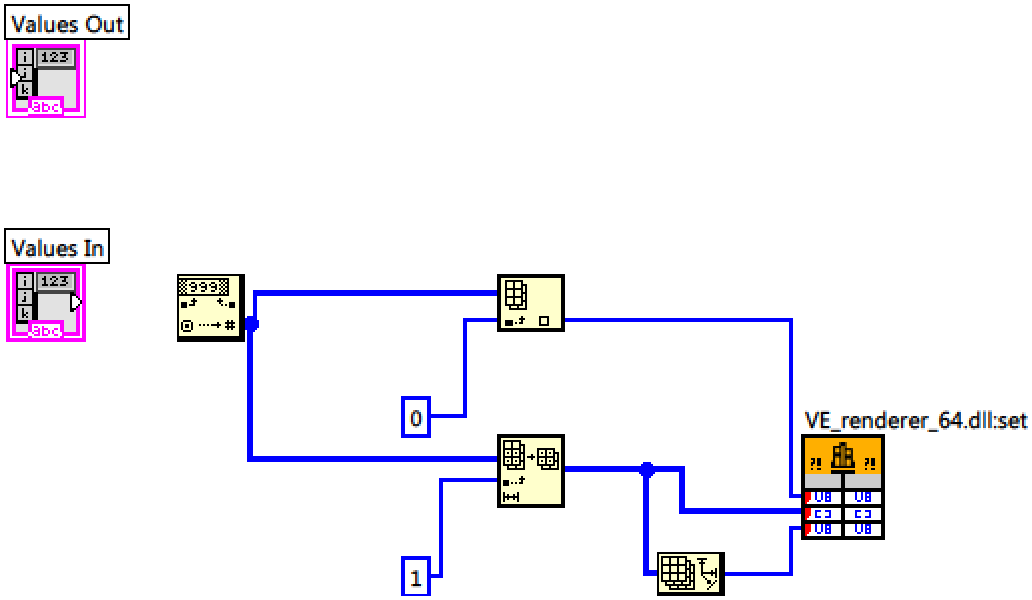
0

4
4
4
4
4

Values Out

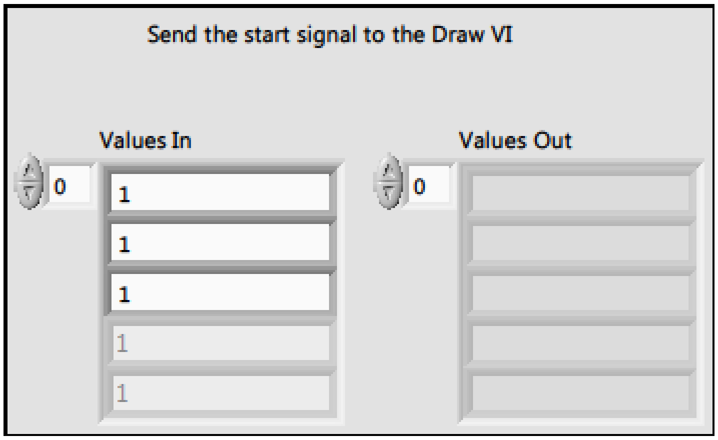
0

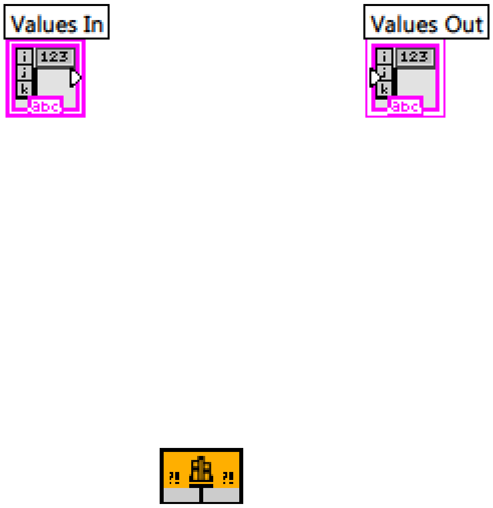
New Set Syntax:
FIRST VALUE IS THE STIMULUS ID. E.G. CENTER IS 0 AND PREPHERY IS 1.
The remaining are profile id for this stimuls for each window.



StartDraw.vi

Sends the start signal to DX Renderer.



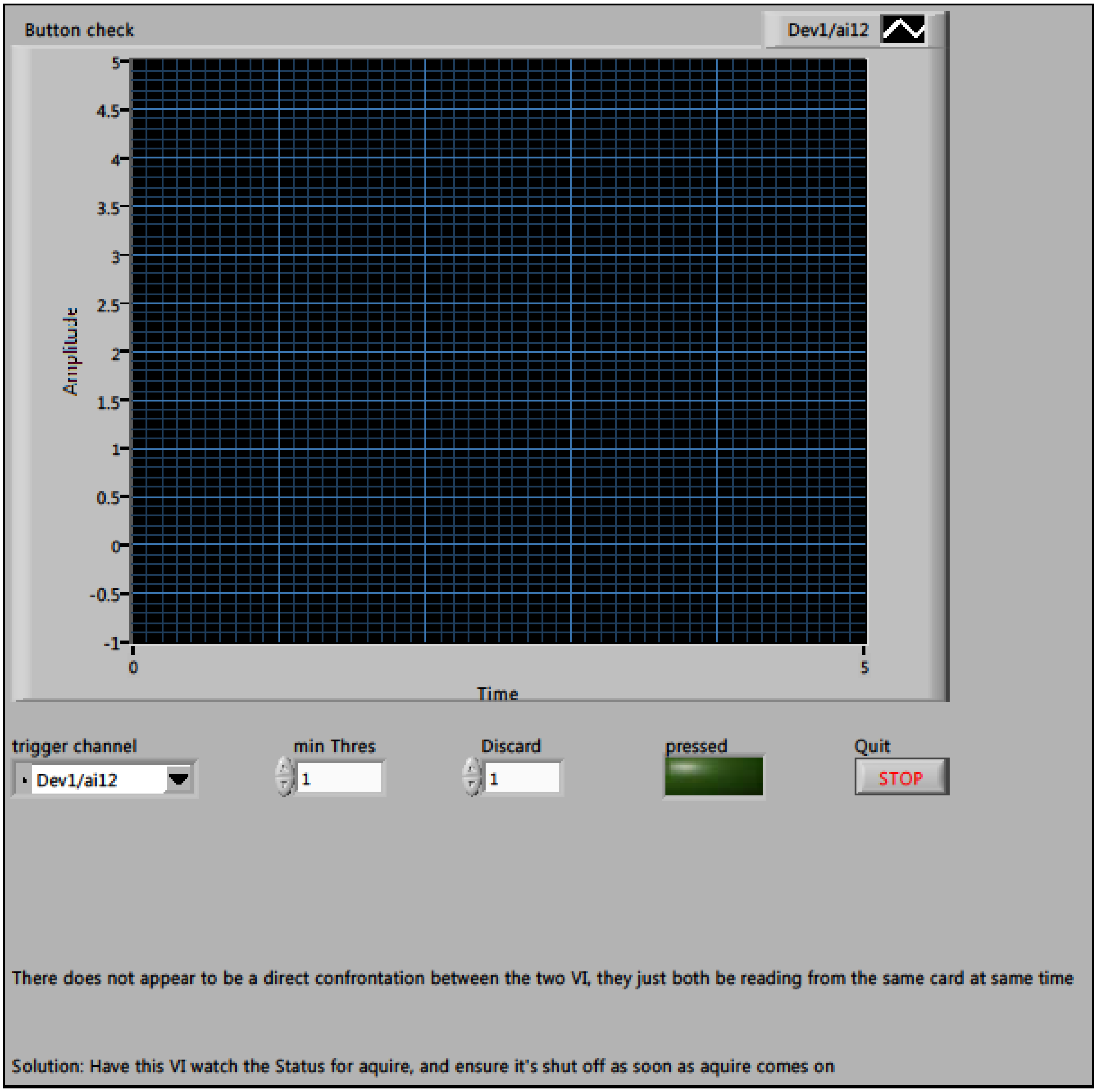


Page 1

TriggerListen.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\TriggerListen.vi
Last modified on 4/14/2014 at 2:41 PM
Printed on 11/15/2014 at 3:52 PM

TriggerListen.vi

trigger channel



trigger channel: Dev1/ai12

min Thres: 1

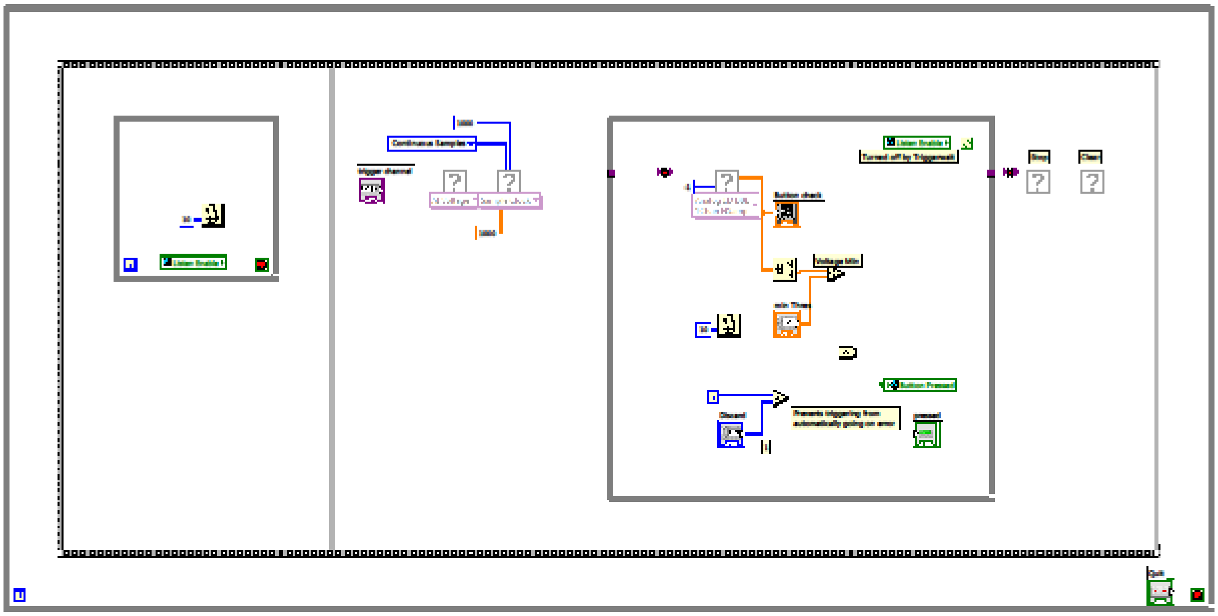
Discard: 1

There does not appear to be a direct confrontation between the two VI, they just both be reading from the same card at same time

Solution: Have this VI watch the Status for aquire, and ensure it's shut off as soon as aquire comes on

TriggerListen.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\TriggerListen.vi
Last modified on 4/14/2014 at 2:41 PM
Printed on 11/15/2014 at 3:52 PM

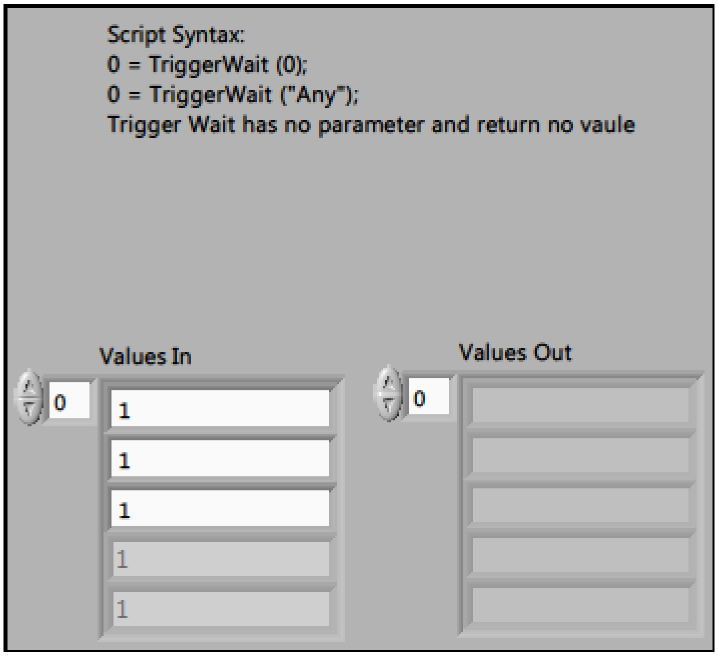
Page 2



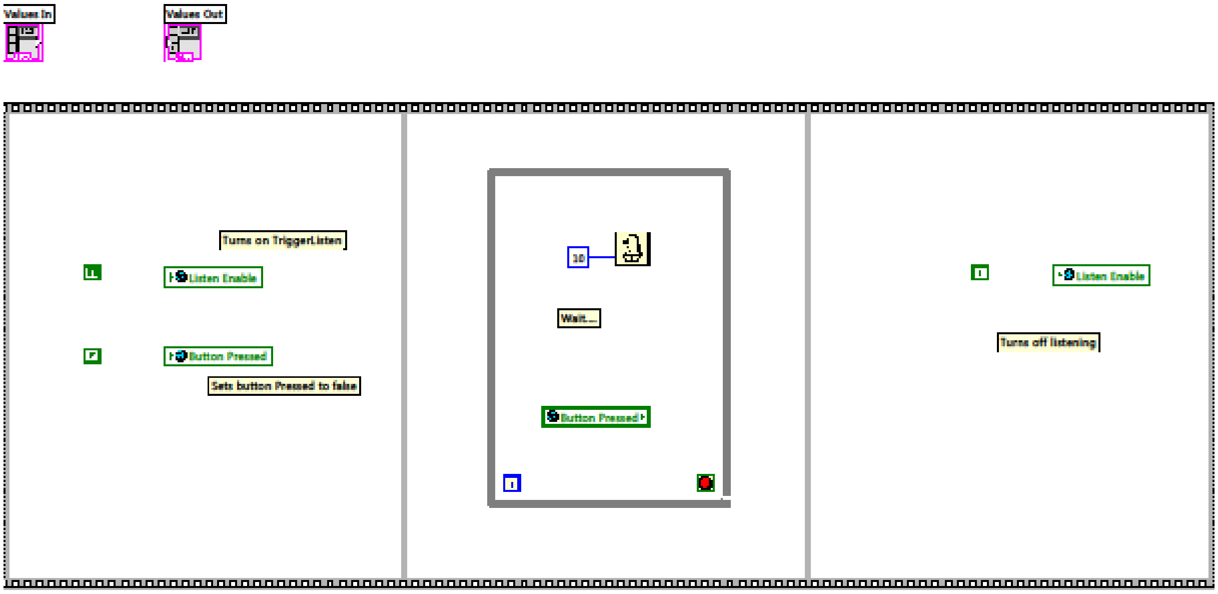
TriggerWait.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\TriggerWait.vi
Last modified on 4/24/2014 at 4:04 PM
Printed on 11/15/2014 at 3:52 PM

TriggerWait.vi

Values In   Values Out



Script Syntax:
0 = TriggerWait (0);
0 = TriggerWait ("Any");
Trigger Wait has no parameter and return no vaule

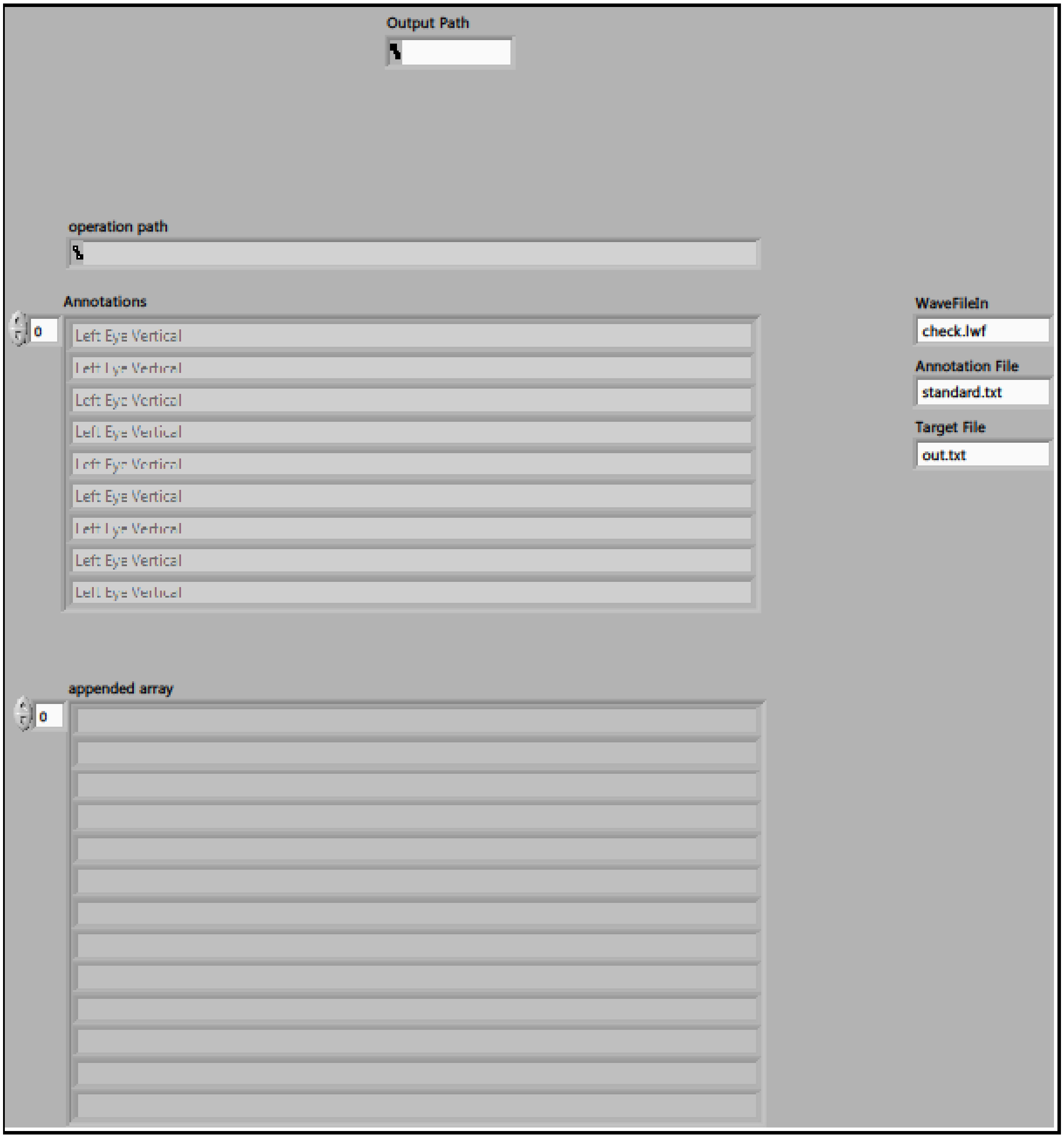


wave2file.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\wave2file.vi
Last modified on 7/17/2014 at 10:42 AM
Printed on 11/15/2014 at 3:52 PM

wave2file.vi

WaveFileIn
Output Path
Annotation File
Target File

wave2file.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\wave2file.vi
Last modified on 7/17/2014 at 10:42 AM
Printed on 11/15/2014 at 3:52 PM



Output Path

operation path

Annotations
0
Left Eye Vertical
Left Eye Vertical
Left Eye Vertical
Left Eye Vertical
Left Eye Vertical
Left Eye Vertical
Left Eye Vertical
Left Eye Vertical
Left Eye Vertical

WaveFileIn
check.lwf

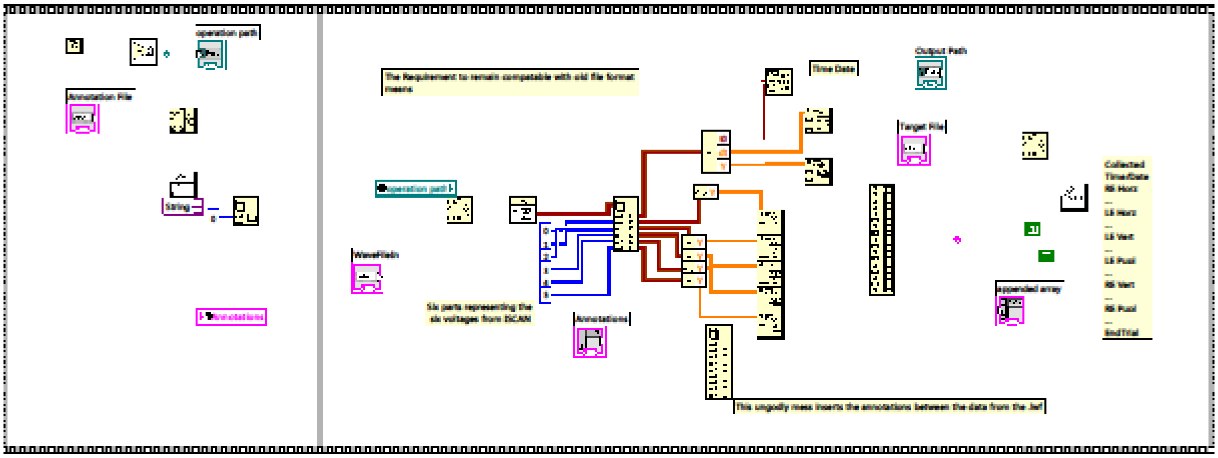
Annotation File
standard.txt

Target File
out.txt

appended array
0

Page 3

wave2file.vi
C:\Users\Steve\Google Drive\Masters Thesis\VisualEyes 2020\VI\wave2file.vi
Last modified on 7/17/2014 at 10:42 AM
Printed on 11/15/2014 at 3:52 PM



The Requirement to remain compatable with old file format means

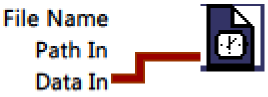
Six parts representing the six voltages from ISCAN

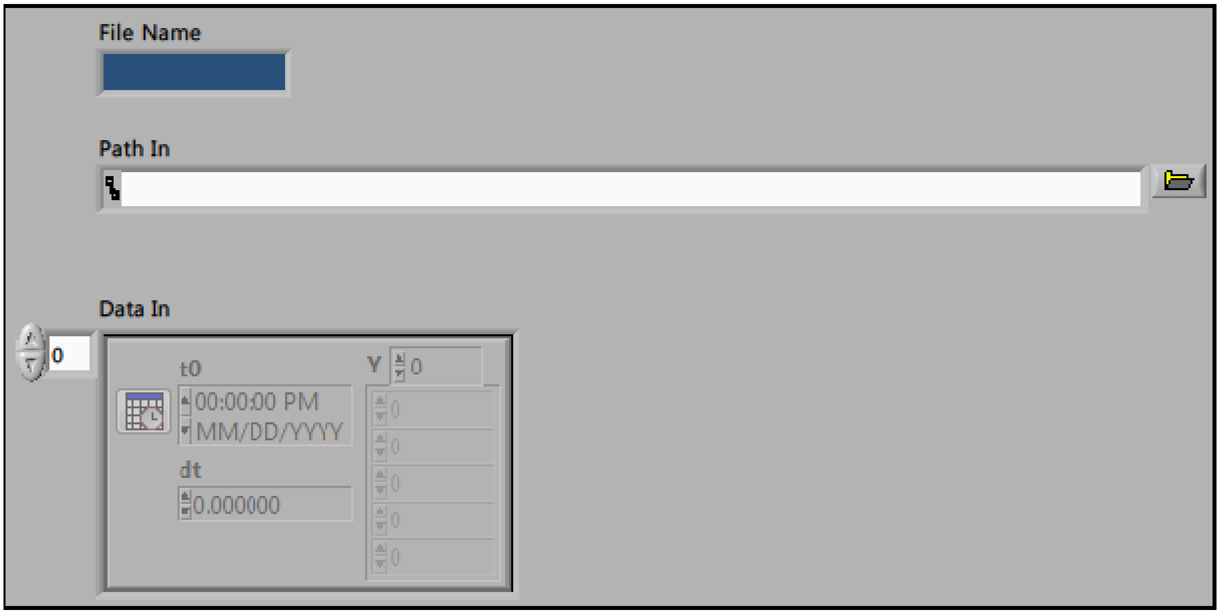
This ungodly mess inserts the annotations between the data from the .twf

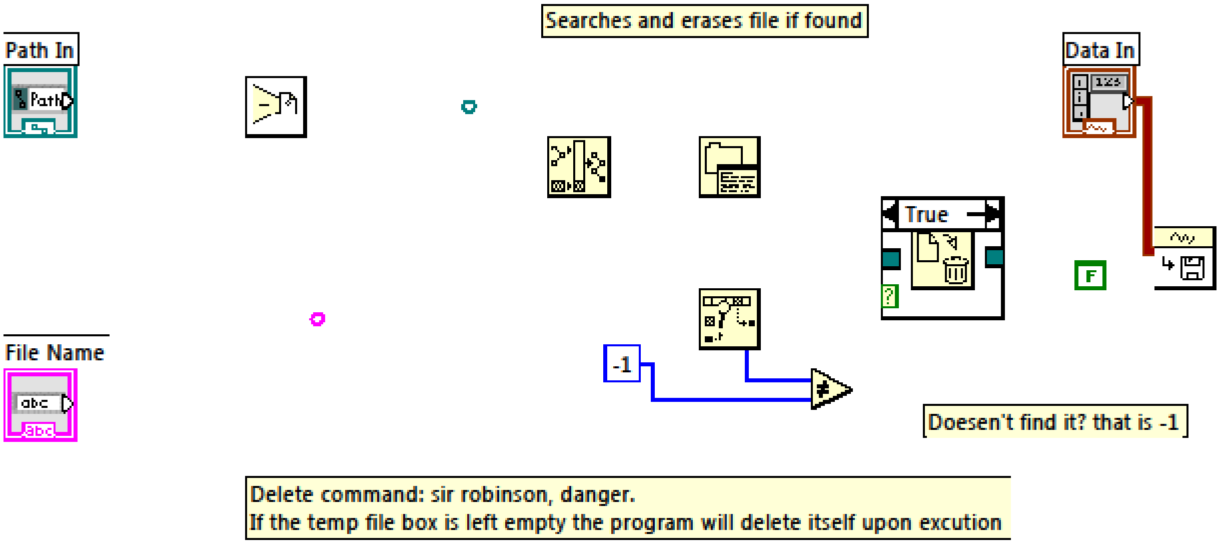
Page 1

write temp file.vi
C:\Users\Steve\Desktop\VisualEyes 2020\VI\write temp file.vi
Last modified on 7/15/2014 at 3:50 PM
Printed on 11/15/2014 at 3:52 PM

write temp file.vi



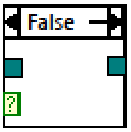




Searches and erases file if found

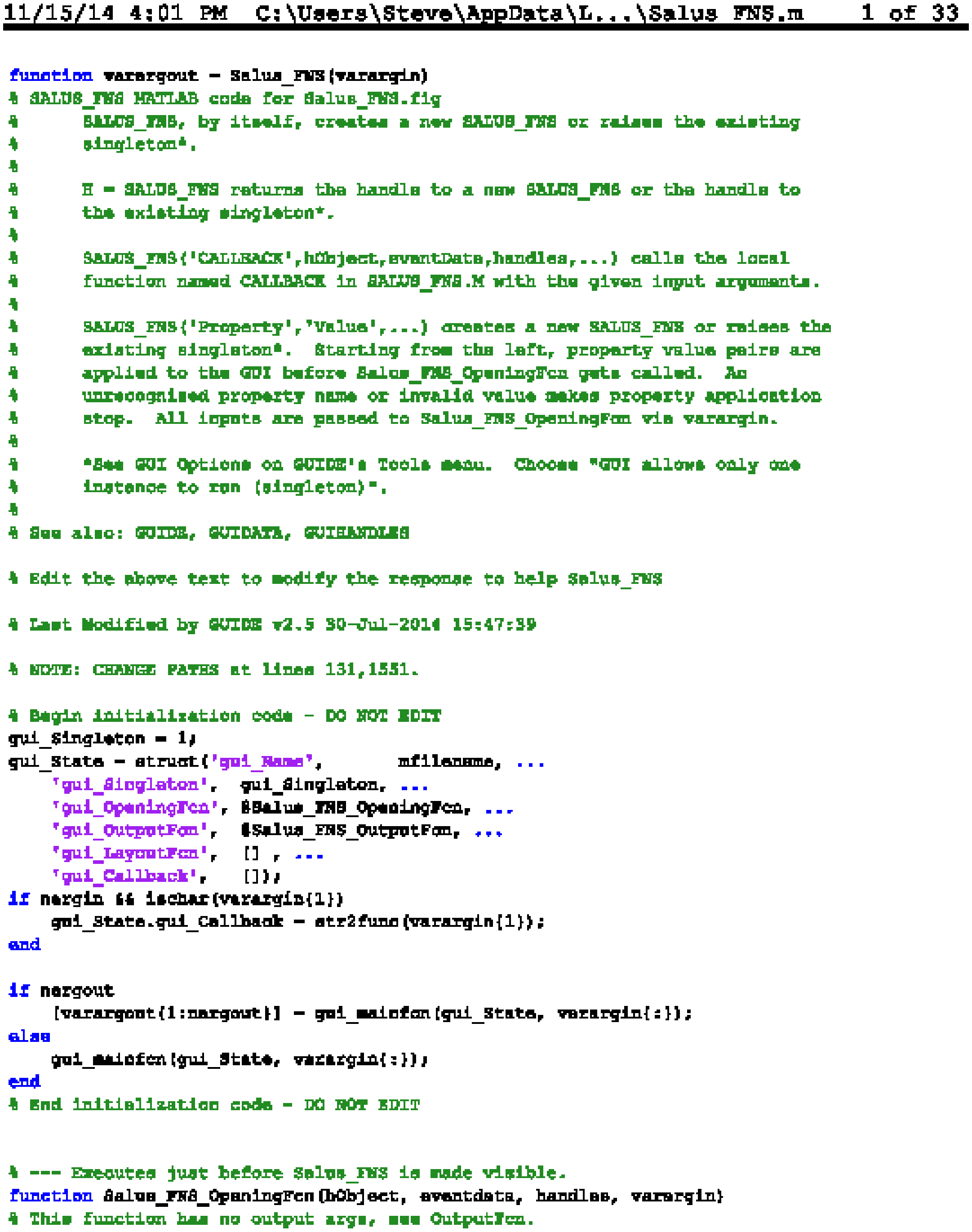
Doesen't find it? that is -1

Delete command: sir robinson, danger.
If the temp file box is left empty the program will delete itself upon excution


False
?

B.1 MATLAB GUI Code

In part B, all MATLAB GUI code is documented including the main GUI and its respective sub-functions.



11/15/14 4:01 PM C:\Users\Steve\AppData\L...\Salus FNS.m   1 of 33

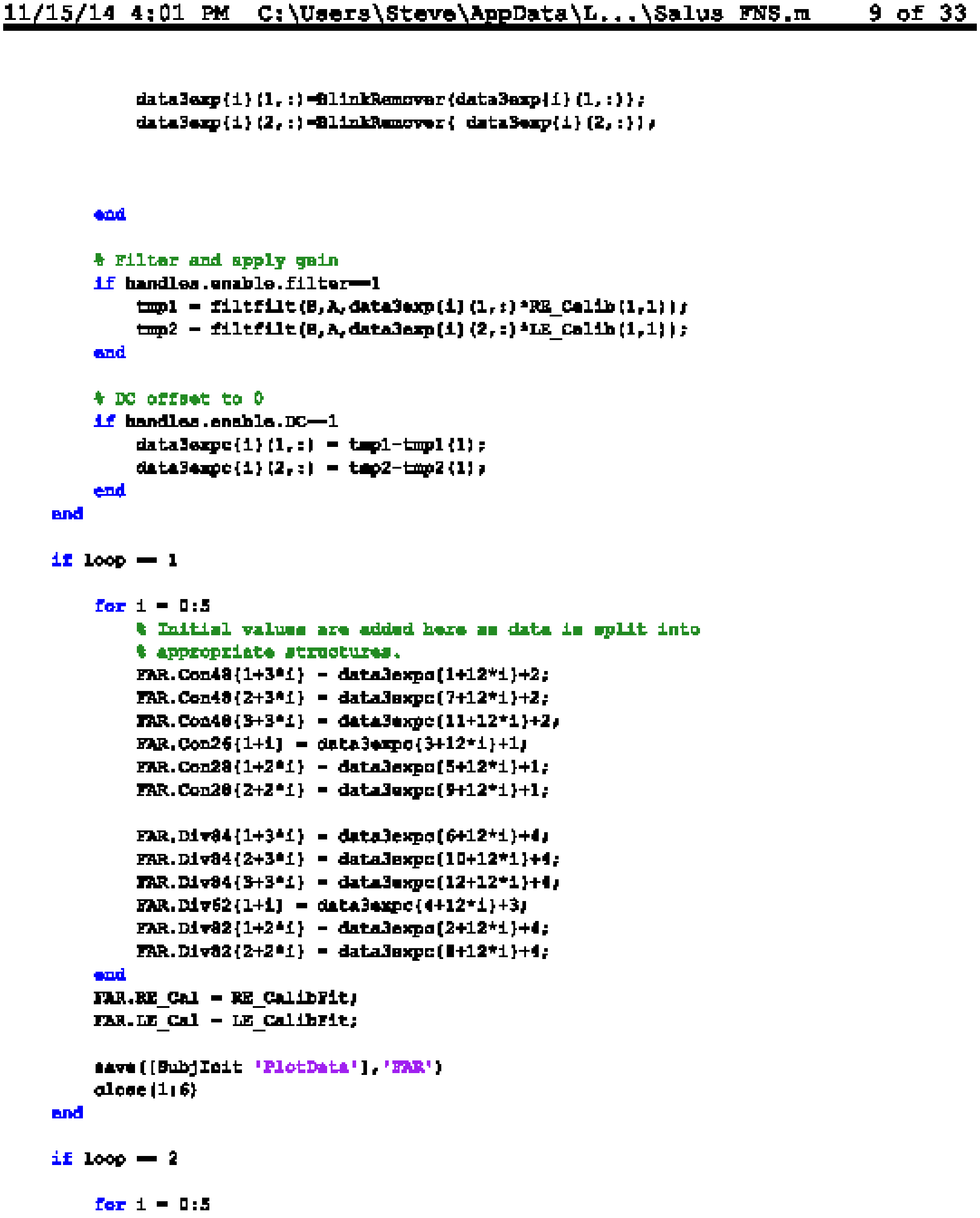
```
function varargout = Salus_FNS(varargin)
% SALUS_FNS MATLAB code for Salus_FNS.fig
%      SALUS_FNS, by itself, creates a new SALUS_FNS or raises the existing
%      singleton*.
%
%      H = SALUS_FNS returns the handle to a new SALUS_FNS or the handle to
%      the existing singleton*.
%
%      SALUS_FNS('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in SALUS_FNS.M with the given input arguments.
%
%      SALUS_FNS('Property','Value',...) creates a new SALUS_FNS or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Salus_FNS_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Salus_FNS_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Salus_FNS

% Last Modified by GUIDE v2.5 30-Jul-2014 15:47:39

% NOTE: CHANGE PATHS at lines 131,1551.

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @Salus_FNS_OpeningFcn, ...
    'gui_OutputFcn',  @Salus_FNS_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Salus_FNS is made visible.
function Salus_FNS_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Salus_FNS (see VARARGIN)


handles.switchbool.Con48=zeros(1,18);
handles.switchbool.Con26=zeros(1,18);
handles.switchbool.Con28=zeros(1,18);
handles.switchbool.Div84=zeros(1,18);
handles.switchbool.Div62=zeros(1,18);
handles.switchbool.Div82=zeros(1,18);

handles.switchbool.Con812=zeros(1,18);
handles.switchbool.Con610=zeros(1,18);
handles.switchbool.Con612=zeros(1,18);
handles.switchbool.Div128=zeros(1,18);
handles.switchbool.Div106=zeros(1,18);
handles.switchbool.Div126=zeros(1,18);

handles.switchbool.M2R5=zeros(1,18);
handles.switchbool.R2M5=zeros(1,18);
handles.switchbool.M2L5=zeros(1,18);
handles.switchbool.L2M5=zeros(1,18);
handles.switchbool.M2R10=zeros(1,18);
handles.switchbool.R2M10=zeros(1,18);
handles.switchbool.M2L10=zeros(1,18);
handles.switchbool.L2M10=zeros(1,18);

handles.switchbool.currentbool=ones(1,18);

handles.t=0:.002:5-.002;
handles.trial=[];

handles.enable.DC=1;
handles.enable.filter=1;
handles.enable.blink=1;
handles.enable.dropdata=1;
handles.enable.buttons=ones(1,18);

handles.issec=[];

handles.filter.fc=25;
handles.filter.order=10;
handles.droptime=3;


% Choose default command line output for Salus_FNS
handles.output = hObject;

% Update handles structure
```

```
guidata(hObject, handles);

% UIWAIT makes Salus_FNS wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Salus_FNS_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% THIS IS THE LOAD BUTTON.
function pushbutton23_Callback(hObject, eventdata, handles)
PATH = 'C:\Users\nicthibodeaux'; % This is where uigetfile will start.
[handles.FileIO.FileName,handles.FileIO.PathName] = ...
    uigetfile('*.*','Select the raw data file',...
    PATH);
handles.FileIO.datatmp = txt2mat([handles.FileIO.PathName,handles.FileIO.FileName],0);
set(handles.pushbutton1,'enable','on');
set(hObject,'backgroundcolor','g');
guidata(hObject, handles);

% THIS IS THE RUN BUTTON.
function pushbutton1_Callback(hObject, eventdata, handles)

FileName=handles.FileIO.FileName;
PathName=handles.FileIO.PathName;
datatmp=handles.FileIO.datatmp;

SubjInit=FileName(6:8); %Steve Add
cd('/Users/nicthibodeaux/Desktop')
if exist(SubjInit) == 7
    rmdir(SubjInit,'s')
    rehash
end
mkdir(SubjInit)
cd(SubjInit)

NanInd = isnan(datatmp(:,1));
x = find(NanInd == 1);
nanbeg2 = [];
nanend2 = [];
for i =2:length(x)
    if x(i) - x(i-1) > 2
        nanbeg = x(i-1)+1;
        nanbeg2 = [nanbeg2;nanbeg];
        nanend = x(i)-1;
        nanend2 = [nanend2;nanend];
    end
end
tmp = [nanbeg2,nanend2];
CalibInd = 1;
for i = 1:length(nanbeg2)
    tmp = datatmp(nanbeg2(i):nanend2(i));
    if length(tmp) == 500
        EndMark(CalibInd) = nanend2(i);
        CalibInd = CalibInd+1;
    end
end

data{1} = datatmp(1:EndMark(72)+1);
data{2} = datatmp(EndMark(72)+2:EndMark(144)+1);
data{3} = datatmp(EndMark(144)+2:EndMark(216)+1);

NumCal = 2;
for loop = 1:3
    appendix = [{'FAR'} {'NEAR'} {'SAC'}];
    NanInd = isnan(data{loop});
    x = find(NanInd == 1);
    nanbeg2 = [];
    nanend2 = [];
    for i = 2:length(x)
        if x(i) - x(i-1) > 2
            nanbeg = x(i-1)+1;
            nanbeg2 = [nanbeg2;nanbeg];
            nanend = x(i)-1;
            nanend2 = [nanend2;nanend];
        end
    end

    tmp = [nanbeg2,nanend2];
    CalibInd = 1;
    ExpInd = 1;
    data2calib = [];
    data2exp = [];
    for i = 1:length(nanbeg2)
        tmp = data{loop}(nanbeg2(i):nanend2(i));
        if length(tmp) == 500
            data2calib(CalibInd,:) = tmp;
            CalibInd = CalibInd+1;
        else
            tmp(end+1:2500) = 0;
            data2exp(ExpInd,:) = tmp;
            ExpInd = ExpInd+1;
        end
    end
```

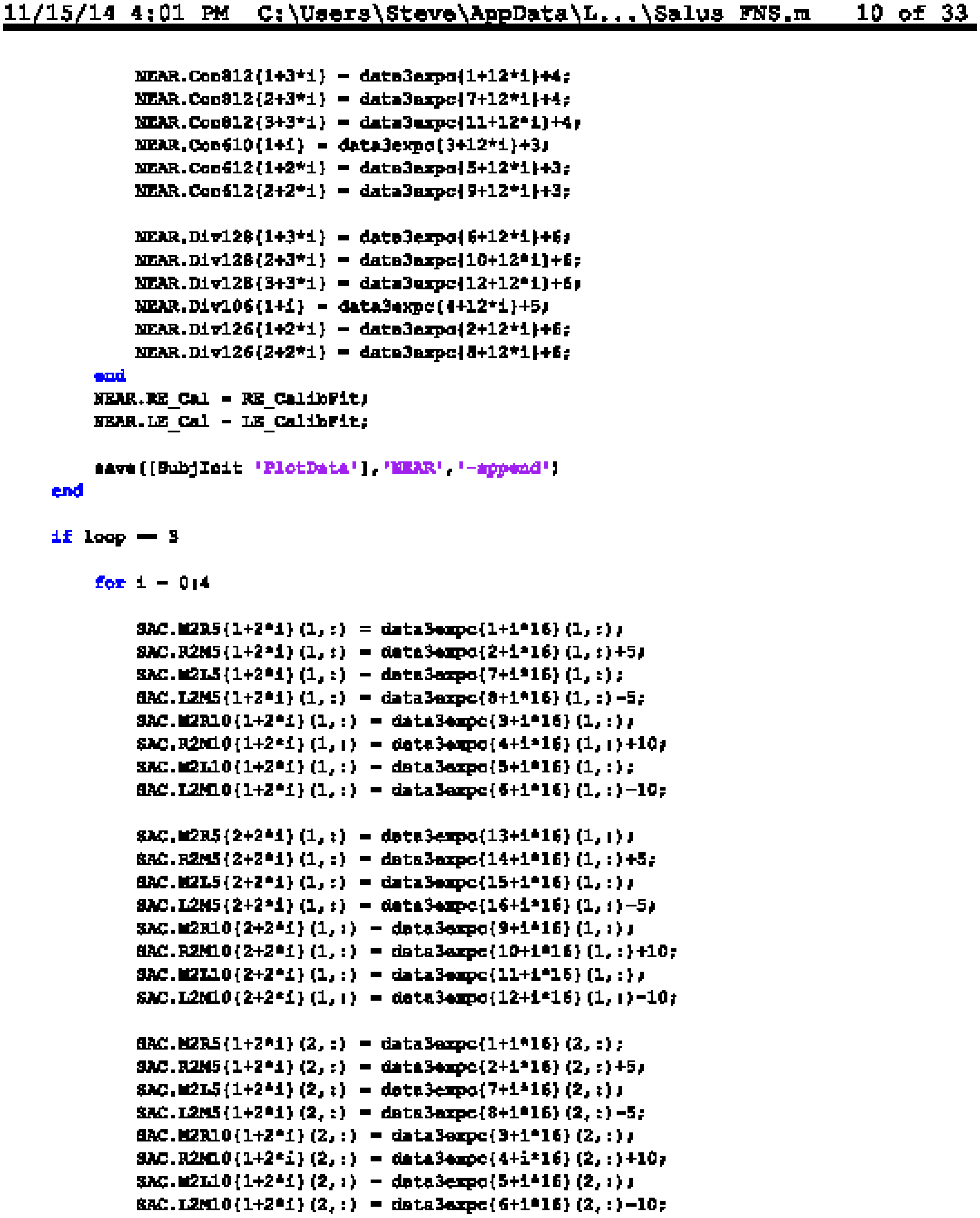
```
Data2calibSize = size(data2calib); j=1;
for i = 1:6:Data2calibSize
    data3calib{j} = data2calib(i:i+5,:);
    j=j+1;
end

Data2expSize = size(data2exp); j=1;
for i = 1:6:Data2expSize(1)
    data3exp{j} = data2exp(i:i+5,:);
    j=j+1;
end

for i=1:NumCal
    if loop == 3
    RECalib(1,i) = data3calib{6+((i-1)*6)}(1,:);
    RECalib(2,i) = data3calib{5+((i-1)*6)}(1,:);
    RECalib(3,i) = data3calib{4+((i-1)*6)}(1,:);

    LECalib(1,i) = data3calib{1+((i-1)*6)}(2,:);
    LECalib(2,i) = data3calib{2+((i-1)*6)}(2,:);
    LECalib(3,i) = data3calib{3+((i-1)*6)}(2,:);
    else
    RECalib(1,i) = data3calib{4+((i-1)*6)}(1,:);
    RECalib(2,i) = data3calib{5+((i-1)*6)}(1,:);
    RECalib(3,i) = data3calib{6+((i-1)*6)}(1,:);

    LECalib(1,i) = data3calib{1+((i-1)*6)}(2,:);
    LECalib(2,i) = data3calib{2+((i-1)*6)}(2,:);
    LECalib(3,i) = data3calib{3+((i-1)*6)}(2,:);
    end
end

LEpt1 = zeros(1,NumCal)+1; LEpt2 = zeros(1,NumCal)+500;
REpt1 = zeros(1,NumCal)+1; REpt2 = zeros(1,NumCal)+500;

if loop == 1
    CalibTargetDegrees = [1 3 5 ; 1 3 5];
end

if loop == 2
    CalibTargetDegrees = [4 5 6 ; 4 5 6];
end

if loop == 3
    CalibTargetDegrees = [10 0 -10 ; 10 0 -10];
end

for Cal=1:2
```

```
        ReselectCalibMean = 0;
        hFig=figure(Cal+2*(loop-1));
        screen_size = get(0, 'ScreenSize');
        set(hFig,'Position',[0 0 screen_size(3) screen_size(4)]);
        while ReselectCalibMean ~= 3
            CalibSP1 = subplot(2,2,1);
            plot([(LECalib{1,Cal});(LECalib{2,Cal});...
                (LECalib{3,Cal})]');
            title('Left Eye  Horizontal')
            ylim([-5 5])
            ylabel('Volts')
            xlabel('Samples')

            CalibSP2 = subplot(2,2,2);
            plot([(RECalib{1,Cal});(RECalib{2,Cal});...
                (RECalib{3,Cal})]');
            title('Right Eye Horizontal')
            ylim([-5 5])
            ylabel('Volts')
            xlabel('Samples')

            subplot(2,2,3)
            cla
            LE_MeanCalibVolts(Cal,:) = [mean(LECalib{1,Cal}(LEpt1(Cal)...
                :LEpt2(Cal))),mean(LECalib{2,Cal}(LEpt1(Cal)...
                :LEpt2(Cal))),mean(LECalib{3,Cal}(LEpt1(Cal)...
                :LEpt2(Cal)))];

            [LE_R,LE_PVAL] = corrcoef(LE_MeanCalibVolts(Cal,:),CalibTargetDegrees↙
(Cal,:));
            LE_CorrCoef(Cal,:) = LE_R(1,2);
            LE_CalibFit(Cal,:) = polyfit(LE_MeanCalibVolts(Cal,:),CalibTargetDegrees↙
(Cal,:),1);
            f = polyval(LE_CalibFit(Cal,:),LE_MeanCalibVolts(Cal,:));

            plot(LE_MeanCalibVolts(Cal,:),CalibTargetDegrees(Cal,:));
            hold on
            plot(LE_MeanCalibVolts(Cal,:),f,'r');
            R = mat2str(LE_CorrCoef(Cal));
            m = mat2str(LE_CalibFit(Cal,1));
            b = mat2str(LE_CalibFit(Cal,2));
            title({['R^2 = ',R(1:5),'        y = ',...
                m(1:4),'x + ',b(1:4)]})
            ylabel('Degrees')
            xlabel('Volts')

            subplot(2,2,4)
            cla
            RE_MeanCalibVolts(Cal,:) = [mean(RECalib{1,Cal}(REpt1(Cal):...
                REpt2(Cal))),mean(RECalib{2,Cal}(REpt1(Cal):...
                REpt2(Cal))),mean(RECalib{3,Cal}(REpt1(Cal):...
                REpt2(Cal))]);
            [RE_R,RE_PVAL] = corrcoef(RE_MeanCalibVolts(Cal,:),CalibTargetDegrees↙
(Cal,:));
            RE_CorrCoef(Cal,:) = RE_R(1,2);
            RE_CalibFit(Cal,:) = polyfit(RE_MeanCalibVolts(Cal,:),CalibTargetDegrees↙
(Cal,:),1);
            f = polyval(RE_CalibFit(Cal,:),RE_MeanCalibVolts(Cal,:));

            plot(RE_MeanCalibVolts(Cal,:),CalibTargetDegrees(Cal,:));
            hold on
            plot(RE_MeanCalibVolts(Cal,:),f,'r');
            R = mat2str(RE_CorrCoef(Cal));
            m = mat2str(RE_CalibFit(Cal,1));
            b = mat2str(RE_CalibFit(Cal,2));
            title({['R^2 = ',R(1:5),'        y = ',...
                 m(1:4),'x + ',b(1:4)]})
            ylabel('Degrees')
            xlabel('Volts')



            ReselectCalibMean = menu('Reselect data to calculate calibration mean','Left↙
Eye Horiz','Right Eye Horiz','Done');

            CalibOption = 'Type';
            if strcmp(CalibOption,'Click') == 1
                if ReselectCalibMean == 1
                    gcf(CalibSP1)
                    [TmpX,TmpY] = ginput(2);
                    LEpt1(Cal) = round(TmpX(1));
                    LEpt2(Cal) = round(TmpX(2));
                end

                if ReselectCalibMean == 2
                    gcf(CalibSP2)
                    [TmpX,TmpY] = ginput(2);
                    REpt1(Cal) = round(TmpX(1));
                    REpt2(Cal) = round(TmpX(2));
                end
            end

            if strcmp(CalibOption,'Type') == 1
                if ReselectCalibMean == 1
                    TmpX = inputdlg('Input new range for calibration','New Calibration↙
Range');

                    TmpX = str2num(TmpX{:});
                    LEpt1(Cal) = TmpX(1);
                    LEpt2(Cal) = TmpX(2);
                end

                if ReselectCalibMean == 2
```

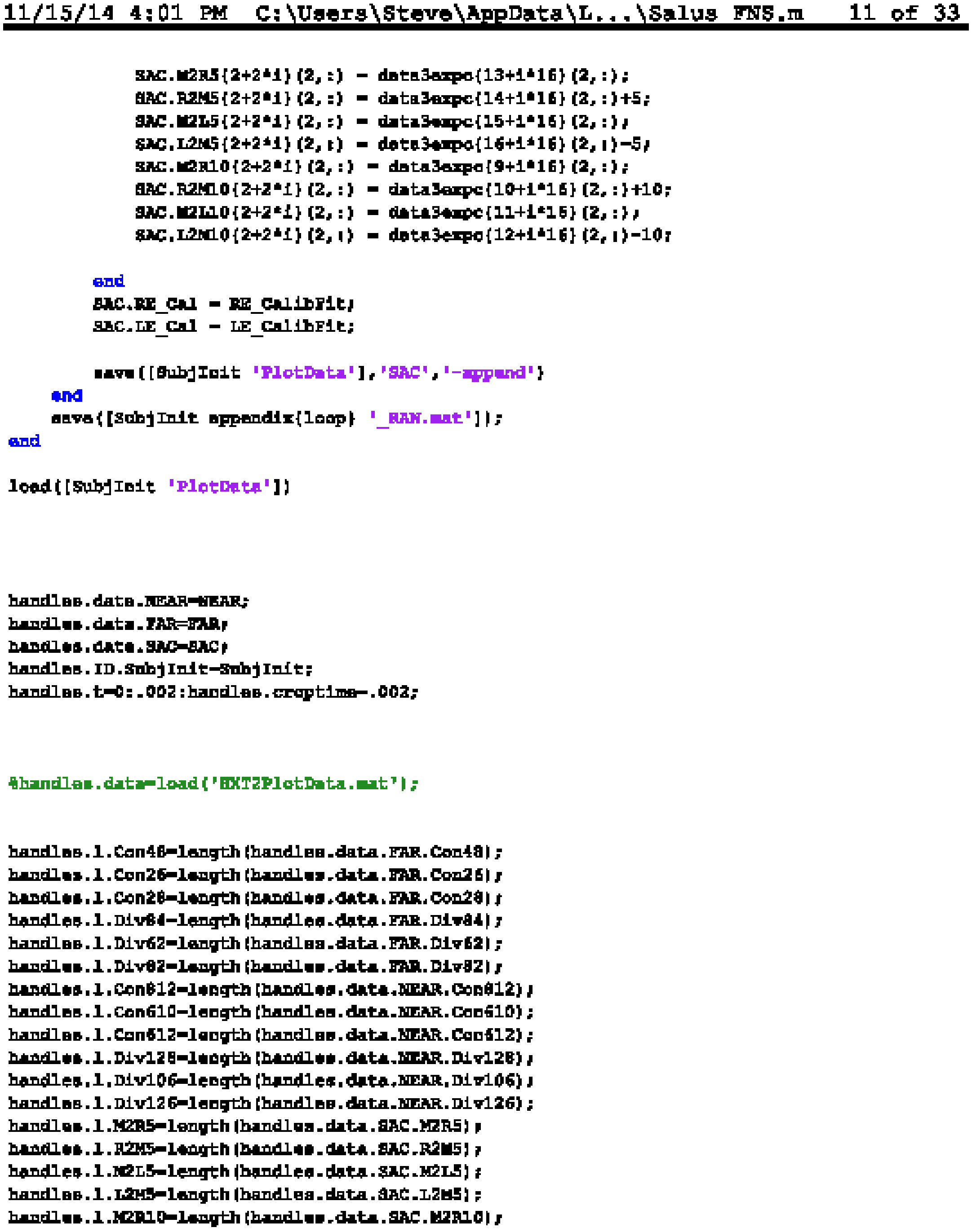
```
                    TmpX = inputdlg('Input new range for calibration','New Calibration↙
Range');
                    TmpX = str2num(TmpX{:});
                    REpt1(Cal) = TmpX(1);
                    REpt2(Cal) = TmpX(2);
                end
            end
        end
    end

    sprintf('Cal 1: Gain is %1.3f, Offset is %1.3f, R^2 is %1.3f\n\nCal 2: Gain is %1.3f, ↙
Offset is %1.3f, R^2 is %1.3f',...
        RE_CalibFit(1,1),RE_CalibFit(1,2),RE_CorrCoef(1),RE_CalibFit(2,1),RE_CalibFit↙
(2,2),RE_CorrCoef(2))

    % Cal is chosen based upon criteria that R value is greater than .95
    if abs(RE_CorrCoef(1)) >= .95 && abs(RE_CorrCoef(2)) >= .95
        RE_Calib = (RE_CalibFit(1,:)+RE_CalibFit(2,:))/2;
        LE_Calib = (LE_CalibFit(1,:)+LE_CalibFit(2,:))/2;
    elseif abs(RE_CorrCoef(1)) >= .95
        RE_Calib = RE_CalibFit(1,:);
        LE_Calib = LE_CalibFit(1,:);
    elseif abs(RE_CorrCoef(2)) >= .95
        RE_Calib = RE_CalibFit(2,:);
        LE_Calib = LE_CalibFit(2,:);
    else

    end

    save([SubjInit appendix{loop} '_RAN.mat']);

end

appendix = [{'FAR'} {'NEAR'} {'SAC'}];

for loop = 1:3
    load([SubjInit appendix{loop} '_RAN.mat']);
    [B,A] = butter(handles.filter.order,handles.filter.fc/250);
    for i = 1:length(data3exp)
        %Crop Data
        if handles.enable.cropdata==1
            tmp1=data3exp{i}(1,1:500*handles.croptime);
            tmp2=data3exp{i}(2,1:500*handles.croptime);
            data3exp{i}=[];
            data3exp{i}=[];
            data3exp{i}(1,:)=tmp1;
            data3exp{i}(2,:)=tmp2;
        end

        %Blink Remover
        if handles.enable.blink==1
```



```
            data3exp{i}(1,:)=BlinkRemover(data3exp{i}(1,:));
            data3exp{i}(2,:)=BlinkRemover( data3exp{i}(2,:));



        end

        % Filter and apply gain
        if handles.enable.filter==1
            tmp1 = filtfilt(B,A,data3exp{i}(1,:)*RE_Calib(1,1));
            tmp2 = filtfilt(B,A,data3exp{i}(2,:)*LE_Calib(1,1));
        end

        % DC offset to 0
        if handles.enable.DC==1
            data3expc{i}(1,:) = tmp1-tmp1(1);
            data3expc{i}(2,:) = tmp2-tmp2(1);
        end
    end

    if loop == 1

        for i = 0:5
            % Initial values are added here as data is split into
            % appropriate structures.
            FAR.Con48{1+3*i} = data3expc{1+12*i}+2;
            FAR.Con48{2+3*i} = data3expc{7+12*i}+2;
            FAR.Con48{3+3*i} = data3expc{11+12*i}+2;
            FAR.Con26{1+i} = data3expc{3+12*i}+1;
            FAR.Con28{1+2*i} = data3expc{5+12*i}+1;
            FAR.Con28{2+2*i} = data3expc{9+12*i}+1;

            FAR.Div84{1+3*i} = data3expc{6+12*i}+4;
            FAR.Div84{2+3*i} = data3expc{10+12*i}+4;
            FAR.Div84{3+3*i} = data3expc{12+12*i}+4;
            FAR.Div62{1+i} = data3expc{4+12*i}+3;
            FAR.Div82{1+2*i} = data3expc{2+12*i}+4;
            FAR.Div82{2+2*i} = data3expc{8+12*i}+4;
        end
        FAR.RE_Cal = RE_CalibFit;
        FAR.LE_Cal = LE_CalibFit;

        save([SubjInit 'PlotData'],'FAR')
        close(1:6)
    end

    if loop == 2

        for i = 0:5
```

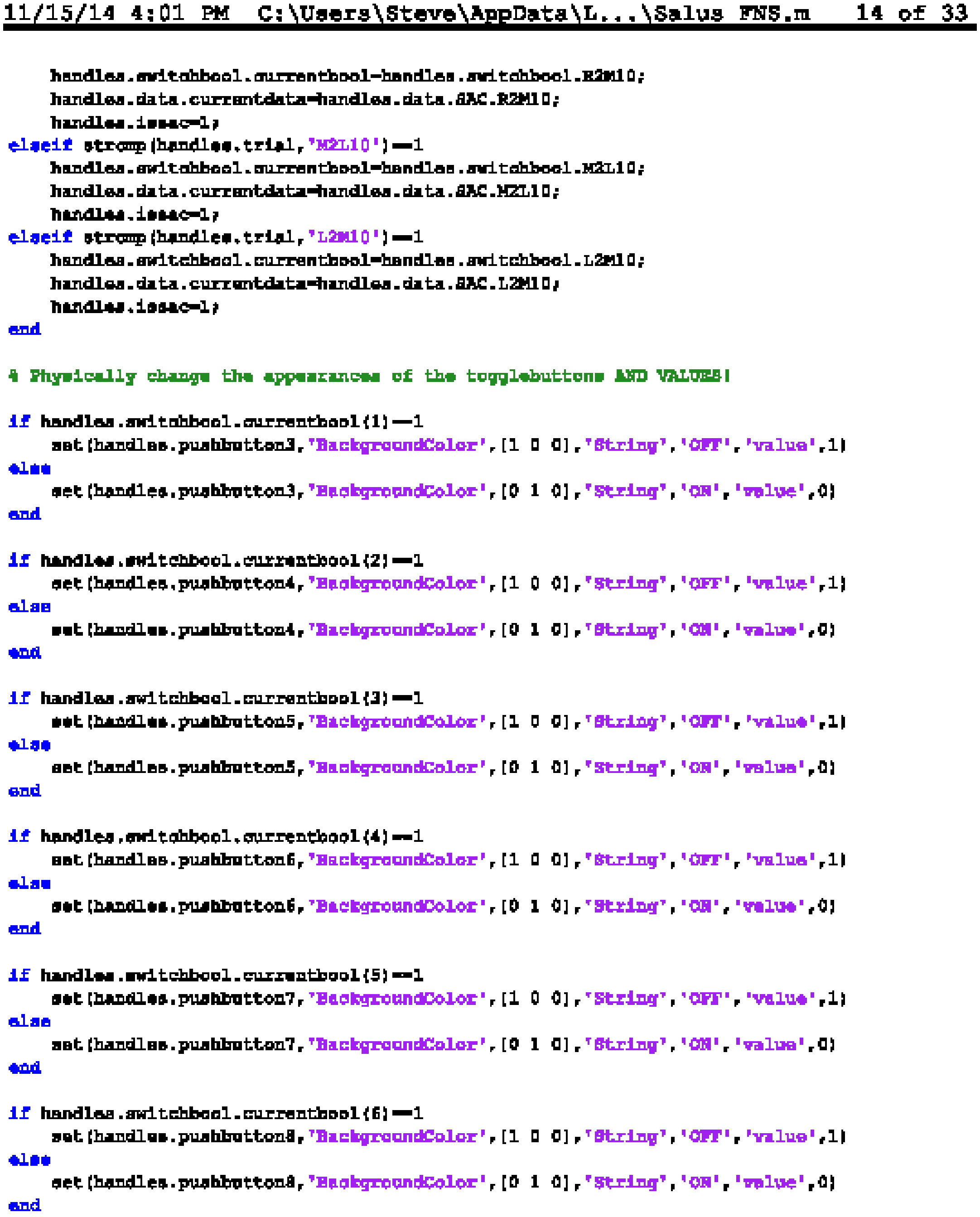


```
            NEAR.Con812{1+3*i} = data3expc{1+12*i}+4;
            NEAR.Con812{2+3*i} = data3expc{7+12*i}+4;
            NEAR.Con812{3+3*i} = data3expc{11+12*i}+4;
            NEAR.Con610{1+i} = data3expc{3+12*i}+3;
            NEAR.Con612{1+2*i} = data3expc{5+12*i}+3;
            NEAR.Con612{2+2*i} = data3expc{9+12*i}+3;

            NEAR.Div128{1+3*i} = data3expc{6+12*i}+6;
            NEAR.Div128{2+3*i} = data3expc{10+12*i}+6;
            NEAR.Div128{3+3*i} = data3expc{12+12*i}+6;
            NEAR.Div106{1+i} = data3expc{4+12*i}+5;
            NEAR.Div126{1+2*i} = data3expc{2+12*i}+6;
            NEAR.Div126{2+2*i} = data3expc{8+12*i}+6;
        end
        NEAR.RE_Cal = RE_CalibFit;
        NEAR.LE_Cal = LE_CalibFit;

        save([SubjInit 'PlotData'],'NEAR','-append')
    end

    if loop == 3

        for i = 0:4

            SAC.M2R5{1+2*i}(1,:) = data3expc{1+i*16}(1,:);
            SAC.R2M5{1+2*i}(1,:) = data3expc{2+i*16}(1,:)+5;
            SAC.M2L5{1+2*i}(1,:) = data3expc{7+i*16}(1,:);
            SAC.L2M5{1+2*i}(1,:) = data3expc{8+i*16}(1,:)-5;
            SAC.M2R10{1+2*i}(1,:) = data3expc{3+i*16}(1,:);
            SAC.R2M10{1+2*i}(1,:) = data3expc{4+i*16}(1,:)+10;
            SAC.M2L10{1+2*i}(1,:) = data3expc{5+i*16}(1,:);
            SAC.L2M10{1+2*i}(1,:) = data3expc{6+i*16}(1,:)-10;

            SAC.M2R5{2+2*i}(1,:) = data3expc{13+i*16}(1,:);
            SAC.R2M5{2+2*i}(1,:) = data3expc{14+i*16}(1,:)+5;
            SAC.M2L5{2+2*i}(1,:) = data3expc{15+i*16}(1,:);
            SAC.L2M5{2+2*i}(1,:) = data3expc{16+i*16}(1,:)-5;
            SAC.M2R10{2+2*i}(1,:) = data3expc{9+i*16}(1,:);
            SAC.R2M10{2+2*i}(1,:) = data3expc{10+i*16}(1,:)+10;
            SAC.M2L10{2+2*i}(1,:) = data3expc{11+i*16}(1,:);
            SAC.L2M10{2+2*i}(1,:) = data3expc{12+i*16}(1,:)-10;

            SAC.M2R5{1+2*i}(2,:) = data3expc{1+i*16}(2,:);
            SAC.R2M5{1+2*i}(2,:) = data3expc{2+i*16}(2,:)+5;
            SAC.M2L5{1+2*i}(2,:) = data3expc{7+i*16}(2,:);
            SAC.L2M5{1+2*i}(2,:) = data3expc{8+i*16}(2,:)-5;
            SAC.M2R10{1+2*i}(2,:) = data3expc{3+i*16}(2,:);
            SAC.R2M10{1+2*i}(2,:) = data3expc{4+i*16}(2,:)+10;
            SAC.M2L10{1+2*i}(2,:) = data3expc{5+i*16}(2,:);
            SAC.L2M10{1+2*i}(2,:) = data3expc{6+i*16}(2,:)-10;
```



```
            SAC.M2R5{2+2*i}(2,:) = data3expc{13+i*16}(2,:);
            SAC.R2M5{2+2*i}(2,:) = data3expc{14+i*16}(2,:)+5;
            SAC.M2L5{2+2*i}(2,:) = data3expc{15+i*16}(2,:);
            SAC.L2M5{2+2*i}(2,:) = data3expc{16+i*16}(2,:)-5;
            SAC.M2R10{2+2*i}(2,:) = data3expc{9+i*16}(2,:);
            SAC.R2M10{2+2*i}(2,:) = data3expc{10+i*16}(2,:)+10;
            SAC.M2L10{2+2*i}(2,:) = data3expc{11+i*16}(2,:);
            SAC.L2M10{2+2*i}(2,:) = data3expc{12+i*16}(2,:)-10;

        end
        SAC.RE_Cal = RE_CalibFit;
        SAC.LE_Cal = LE_CalibFit;

        save([SubjInit 'PlotData'],'SAC','-append')
    end
    save([SubjInit appendix{loop} '_RAW.mat']);
end

load([SubjInit 'PlotData'])




handles.data.NEAR=NEAR;
handles.data.FAR=FAR;
handles.data.SAC=SAC;
handles.ID.SubjInit=SubjInit;
handles.t=0:.002:handles.croptime-.002;



%handles.data=load('EXT2PlotData.mat');


handles.l.Con48=length(handles.data.FAR.Con48);
handles.l.Con26=length(handles.data.FAR.Con26);
handles.l.Con28=length(handles.data.FAR.Con28);
handles.l.Div84=length(handles.data.FAR.Div84);
handles.l.Div62=length(handles.data.FAR.Div62);
handles.l.Div82=length(handles.data.FAR.Div82);
handles.l.Con812=length(handles.data.NEAR.Con812);
handles.l.Con610=length(handles.data.NEAR.Con610);
handles.l.Con612=length(handles.data.NEAR.Con612);
handles.l.Div128=length(handles.data.NEAR.Div128);
handles.l.Div106=length(handles.data.NEAR.Div106);
handles.l.Div126=length(handles.data.NEAR.Div126);
handles.l.M2R5=length(handles.data.SAC.M2R5);
handles.l.R2M5=length(handles.data.SAC.R2M5);
handles.l.M2L5=length(handles.data.SAC.M2L5);
handles.l.L2M5=length(handles.data.SAC.L2M5);
handles.l.M2R10=length(handles.data.SAC.M2R10);
handles.l.R2M10=length(handles.data.SAC.R2M10);
handles.l.M2L10=length(handles.data.SAC.M2L10);
handles.l.L2M10=length(handles.data.SAC.L2M10);



set(hObject,'backgroundcolor','g');
set(handles.popupmenu1,'enable','on');
guidata(hObject, handles);




% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% BUTTON1!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);
FNS_AxesPlot(handles);

guidata(hObject, handles);



% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)

contents = cellstr(get(hObject,'String'));
handles.trial=contents{get(hObject,'Value')};

if strcmp(handles.trial,'Con48')==1
    handles.switchbool.currentbool=handles.switchbool.Con48;
    handles.data.currentdata=handles.data.FAR.Con48;
    handles.issac=0;
elseif strcmp(handles.trial,'Con26')==1
    handles.switchbool.currentbool=handles.switchbool.Con26;
    handles.data.currentdata=handles.data.FAR.Con26;
    handles.issac=0;
elseif strcmp(handles.trial,'Con28')==1
    handles.switchbool.currentbool=handles.switchbool.Con28;
    handles.data.currentdata=handles.data.FAR.Con28;
    handles.issac=0;
elseif strcmp(handles.trial,'Div84')==1
    handles.switchbool.currentbool=handles.switchbool.Div84;
    handles.data.currentdata=handles.data.FAR.Div84;
    handles.issac=0;
elseif strcmp(handles.trial,'Div62')==1
    handles.switchbool.currentbool=handles.switchbool.Div62;
    handles.data.currentdata=handles.data.FAR.Div62;
    handles.issac=0;
elseif strcmp(handles.trial,'Div82')==1
```

```
    handles.switchbool.currentbool=handles.switchbool.Div82;
    handles.data.currentdata=handles.data.FAR.Div82;
    handles.issac=0;


elseif strcmp(handles.trial,'Con812')==1
    handles.switchbool.currentbool=handles.switchbool.Con812;
    handles.data.currentdata=handles.data.NEAR.Con812;
    handles.issac=0;
elseif strcmp(handles.trial,'Con610')==1
    handles.switchbool.currentbool=handles.switchbool.Con610;
    handles.data.currentdata=handles.data.NEAR.Con610;
    handles.issac=0;
elseif strcmp(handles.trial,'Con612')==1
    handles.switchbool.currentbool=handles.switchbool.Con612;
    handles.data.currentdata=handles.data.NEAR.Con612;
    handles.issac=0;
elseif strcmp(handles.trial,'Div128')==1
    handles.switchbool.currentbool=handles.switchbool.Div128;
    handles.data.currentdata=handles.data.NEAR.Div128;
    handles.issac=0;
elseif strcmp(handles.trial,'Div106')==1
    handles.switchbool.currentbool=handles.switchbool.Div106;
    handles.data.currentdata=handles.data.NEAR.Div106;
    handles.issac=0;
elseif strcmp(handles.trial,'Div126')==1
    handles.switchbool.currentbool=handles.switchbool.Div126;
    handles.data.currentdata=handles.data.NEAR.Div126;
    handles.issac=0;

elseif strcmp(handles.trial,'M2R5')==1
    handles.switchbool.currentbool=handles.switchbool.M2R5;
    handles.data.currentdata=handles.data.SAC.M2R5;
    handles.issac=1;
elseif strcmp(handles.trial,'R2M5')==1
    handles.switchbool.currentbool=handles.switchbool.R2M5;
    handles.data.currentdata=handles.data.SAC.R2M5;
    handles.issac=1;
elseif strcmp(handles.trial,'M2L5')==1
    handles.switchbool.currentbool=handles.switchbool.M2L5;
    handles.data.currentdata=handles.data.SAC.M2L5;
    handles.issac=1;
elseif strcmp(handles.trial,'L2M5')==1
    handles.switchbool.currentbool=handles.switchbool.L2M5;
    handles.data.currentdata=handles.data.SAC.L2M5;
    handles.issac=1;
elseif strcmp(handles.trial,'M2R10')==1
    handles.switchbool.currentbool=handles.switchbool.M2R10;
    handles.data.currentdata=handles.data.SAC.M2R10;
    handles.issac=1;
elseif strcmp(handles.trial,'R2M10')==1
```



```
    handles.switchbool.currentbool=handles.switchbool.R2M10;
    handles.data.currentdata=handles.data.SAC.R2M10;
    handles.issac=1;
elseif strcmp(handles.trial,'M2L10')==1
    handles.switchbool.currentbool=handles.switchbool.M2L10;
    handles.data.currentdata=handles.data.SAC.M2L10;
    handles.issac=1;
elseif strcmp(handles.trial,'L2M10')==1
    handles.switchbool.currentbool=handles.switchbool.L2M10;
    handles.data.currentdata=handles.data.SAC.L2M10;
    handles.issac=1;
end

% Physically change the appearances of the togglebuttons AND VALUES!

if handles.switchbool.currentbool(1)==1
    set(handles.pushbutton3,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton3,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(2)==1
    set(handles.pushbutton4,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton4,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(3)==1
    set(handles.pushbutton5,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton5,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(4)==1
    set(handles.pushbutton6,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton6,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(5)==1
    set(handles.pushbutton7,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton7,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(6)==1
    set(handles.pushbutton8,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton8,'BackgroundColor',[0 1 0],'String','ON','value',0)
end
```

```
if handles.switchbool.currentbool(7)==1
    set(handles.pushbutton9,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton9,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(8)==1
    set(handles.pushbutton10,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton10,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(9)==1
    set(handles.pushbutton11,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton11,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(10)==1
    set(handles.pushbutton12,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton12,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(11)==1
    set(handles.pushbutton13,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton13,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(12)==1
    set(handles.pushbutton14,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton14,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(13)==1
    set(handles.pushbutton15,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton15,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(14)==1
    set(handles.pushbutton16,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton16,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(15)==1
    set(handles.pushbutton17,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
```

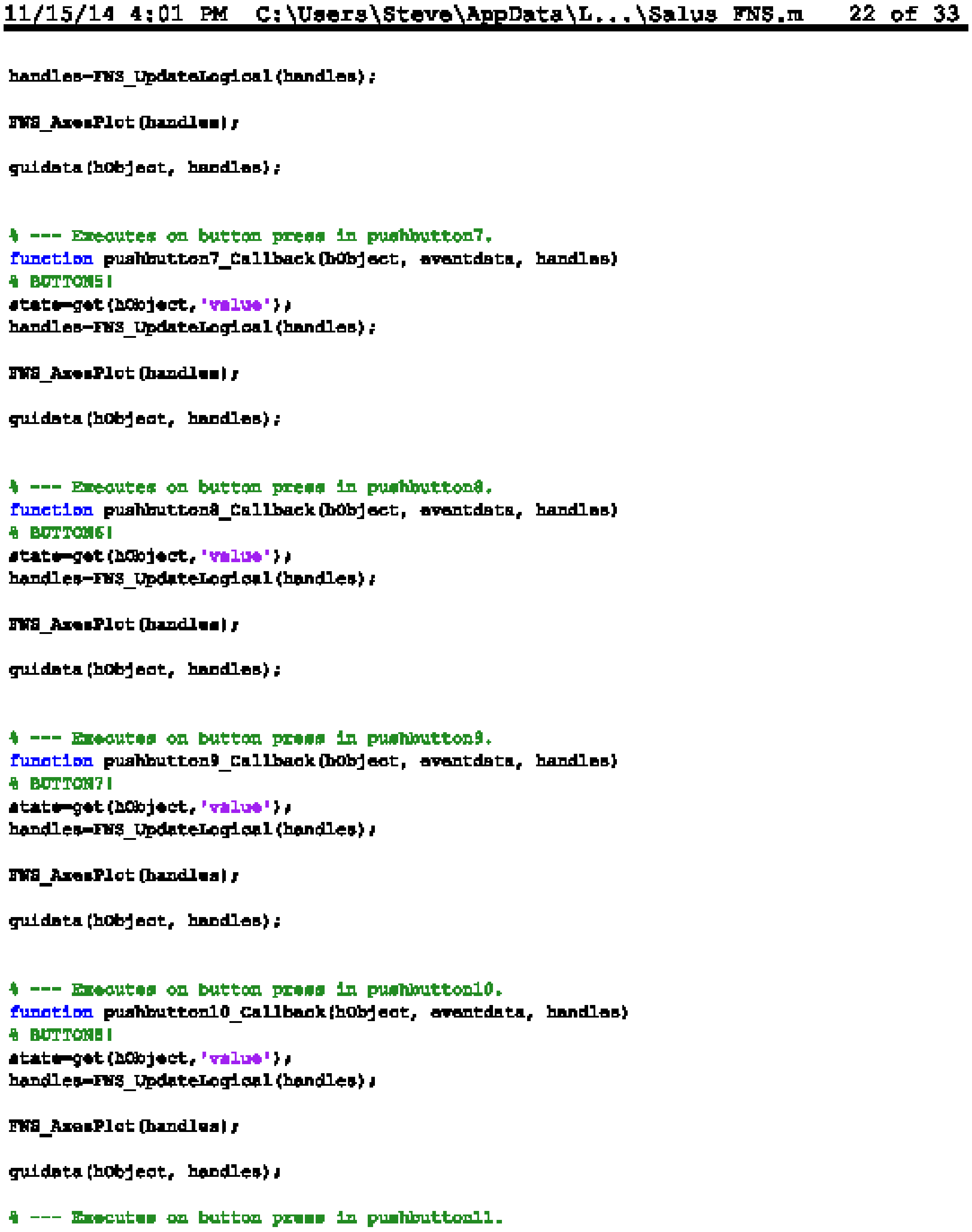
```
    set(handles.pushbutton17,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(16)==1
    set(handles.pushbutton18,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton18,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(17)==1
    set(handles.pushbutton19,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton19,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

if handles.switchbool.currentbool(18)==1
    set(handles.pushbutton20,'BackgroundColor',[1 0 0],'String','OFF','value',1)
else
    set(handles.pushbutton20,'BackgroundColor',[0 1 0],'String','ON','value',0)
end

% Update the enable boolean
tmpbool=zeros(1,18);
if strcmp(handles.trial,'Con48')==1
    tmpboolplus=[ones(1,handles.l.Con48) zeros(1,18-handles.l.Con48)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'Con26')==1
    tmpboolplus=[ones(1,handles.l.Con26) zeros(1,18-handles.l.Con26)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'Con28')==1
    tmpboolplus=[ones(1,handles.l.Con28) zeros(1,18-handles.l.Con28)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'Div84')==1
    tmpboolplus=[ones(1,handles.l.Div84) zeros(1,18-handles.l.Div84)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'Div62')==1
    tmpboolplus=[ones(1,handles.l.Div62) zeros(1,18-handles.l.Div62)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'Div82')==1
    tmpboolplus=[ones(1,handles.l.Div82) zeros(1,18-handles.l.Div82)];
    handles.enable.buttons=tmpbool+tmpboolplus;


elseif strcmp(handles.trial,'Con812')==1
    tmpboolplus=[ones(1,handles.l.Con812) zeros(1,18-handles.l.Con812)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'Con610')==1
    tmpboolplus=[ones(1,handles.l.Con610) zeros(1,18-handles.l.Con610)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'Con612')==1
    tmpboolplus=[ones(1,handles.l.Con612) zeros(1,18-handles.l.Con612)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'Div128')==1
    tmpboolplus=[ones(1,handles.l.Div128) zeros(1,18-handles.l.Div128)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'Div106')==1
    tmpboolplus=[ones(1,handles.l.Div106) zeros(1,18-handles.l.Div106)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'Div126')==1
    tmpboolplus=[ones(1,handles.l.Div126) zeros(1,18-handles.l.Div126)];
    handles.enable.buttons=tmpbool+tmpboolplus;

elseif strcmp(handles.trial,'M2R5')==1
    tmpboolplus=[ones(1,handles.l.M2R5) zeros(1,18-handles.l.M2R5)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'R2M5')==1
    tmpboolplus=[ones(1,handles.l.R2M5) zeros(1,18-handles.l.R2M5)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'M2L5')==1
    tmpboolplus=[ones(1,handles.l.M2L5) zeros(1,18-handles.l.M2L5)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'L2M5')==1
    tmpboolplus=[ones(1,handles.l.L2M5) zeros(1,18-handles.l.L2M5)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'M2R10')==1
    tmpboolplus=[ones(1,handles.l.M2R10) zeros(1,18-handles.l.M2R10)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'R2M10')==1
    tmpboolplus=[ones(1,handles.l.R2M10) zeros(1,18-handles.l.R2M10)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'M2L10')==1
    tmpboolplus=[ones(1,handles.l.M2L10) zeros(1,18-handles.l.M2L10)];
    handles.enable.buttons=tmpbool+tmpboolplus;
elseif strcmp(handles.trial,'L2M10')==1
    tmpboolplus=[ones(1,handles.l.L2M10) zeros(1,18-handles.l.L2M10)];
    handles.enable.buttons=tmpbool+tmpboolplus;
end


% change the enable state according to handles.enable.buttons
if handles.enable.buttons(1)==1
    set(handles.pushbutton3,'enable','on')
else
    set(handles.pushbutton3,'enable','off')
end

if handles.enable.buttons(2)==1
    set(handles.pushbutton4,'enable','on')
else
    set(handles.pushbutton4,'enable','off')
end
```

```
if handles.enable.buttons(3)==1
    set(handles.pushbutton5,'enable','on')
else
    set(handles.pushbutton5,'enable','off')
end

if handles.enable.buttons(4)==1
    set(handles.pushbutton6,'enable','on')
else
    set(handles.pushbutton6,'enable','off')
end

if handles.enable.buttons(5)==1
    set(handles.pushbutton7,'enable','on')
else
    set(handles.pushbutton7,'enable','off')
end

if handles.enable.buttons(6)==1
    set(handles.pushbutton8,'enable','on')
else
    set(handles.pushbutton8,'enable','off')
end

if handles.enable.buttons(7)==1
    set(handles.pushbutton9,'enable','on')
else
    set(handles.pushbutton9,'enable','off')
end

if handles.enable.buttons(8)==1
    set(handles.pushbutton10,'enable','on')
else
    set(handles.pushbutton10,'enable','off')
end

if handles.enable.buttons(9)==1
    set(handles.pushbutton11,'enable','on')
else
    set(handles.pushbutton11,'enable','off')
end

if handles.enable.buttons(10)==1
    set(handles.pushbutton12,'enable','on')
else
    set(handles.pushbutton12,'enable','off')
end

if handles.enable.buttons(11)==1
    set(handles.pushbutton13,'enable','on')
else
```

```
    set(handles.pushbutton13,'enable','off')
end

if handles.enable.buttons(12)==1
    set(handles.pushbutton14,'enable','on')
else
    set(handles.pushbutton14,'enable','off')
end

if handles.enable.buttons(13)==1
    set(handles.pushbutton15,'enable','on')
else
    set(handles.pushbutton15,'enable','off')
end

if handles.enable.buttons(14)==1
    set(handles.pushbutton16,'enable','on')
else
    set(handles.pushbutton16,'enable','off')
end

if handles.enable.buttons(15)==1
    set(handles.pushbutton17,'enable','on')
else
    set(handles.pushbutton17,'enable','off')
end

if handles.enable.buttons(16)==1
    set(handles.pushbutton18,'enable','on')
else
    set(handles.pushbutton18,'enable','off')
end

if handles.enable.buttons(17)==1
    set(handles.pushbutton19,'enable','on')
else
    set(handles.pushbutton19,'enable','off')
end

if handles.enable.buttons(18)==1
    set(handles.pushbutton20,'enable','on')
else
    set(handles.pushbutton20,'enable','off')
end


% Update the main boolean arrays according to the selected case

if strcmp(handles.trial,'Con48')==1
    handles.switchbool.Con48=handles.switchbool.currentbool;
```

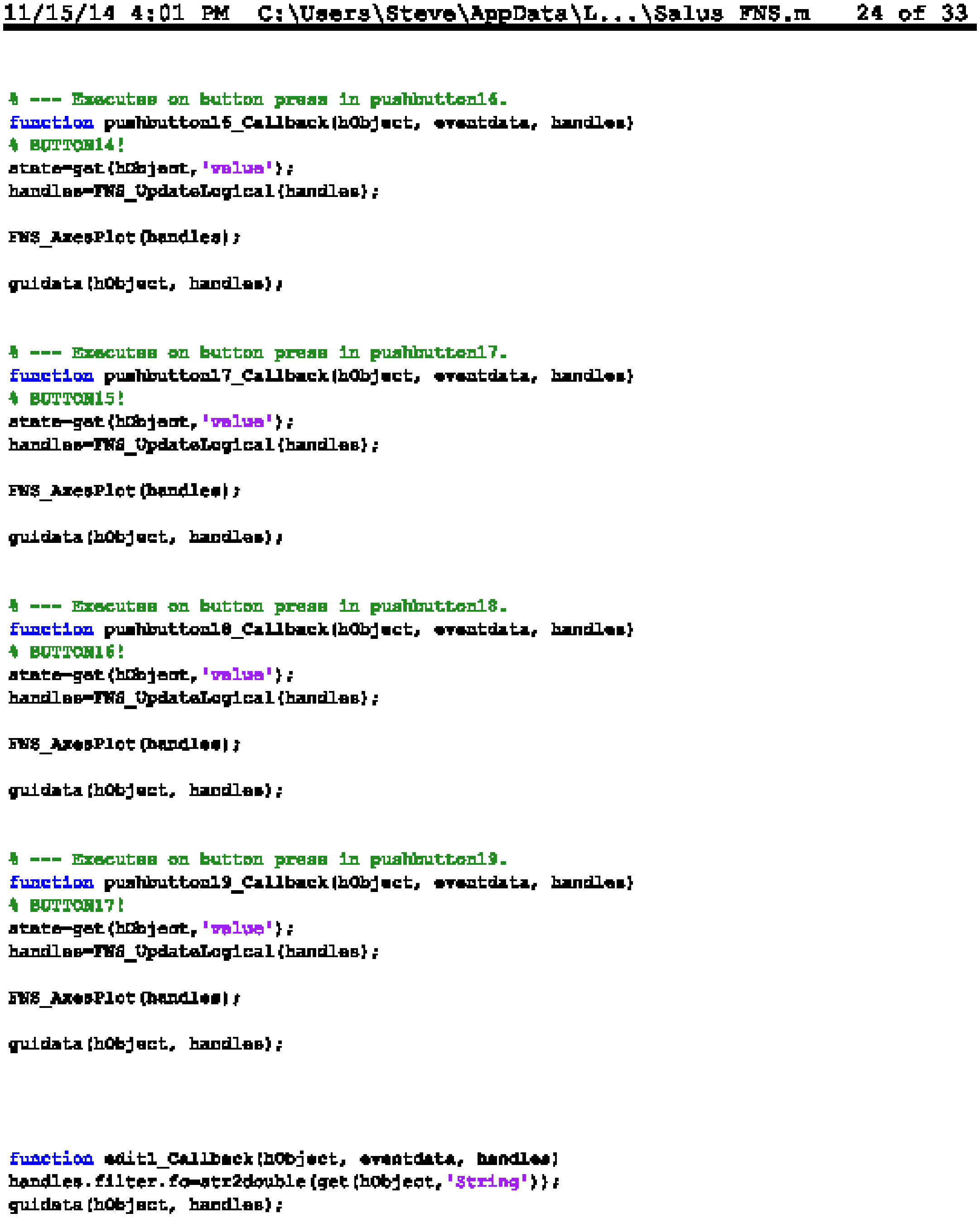
```
elseif strcmp(handles.trial,'Con26')==1
    handles.switchbool.Con26=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Con28')==1
    handles.switchbool.Con28=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Div84')==1
    handles.switchbool.Div84=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Div62')==1
    handles.switchbool.Div62=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Div82')==1
    handles.switchbool.Div82=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Con812')==1
    handles.switchbool.Con812=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Con610')==1
    handles.switchbool.Con610=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Con612')==1
    handles.switchbool.Con612=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Div128')==1
    handles.switchbool.Div128=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Div106')==1
    handles.switchbool.Div106=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Div126')==1
    handles.switchbool.Div126=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'M2R5')==1
    handles.switchbool.M2R5=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'R2M5')==1
    handles.switchbool.R2M5=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'M2L5')==1
    handles.switchbool.M2R5=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'L2M5')==1
    handles.switchbool.L2M5=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'M2R10')==1
    handles.switchbool.M2R10=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'R2M10')==1
    handles.switchbool.R2M10=handles.switchbool.currentbool;
```

```
elseif strcmp(handles.trial,'M2L10')==1
    handles.switchbool.M2L10=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'L2M10')==1
    handles.switchbool.L2M10=handles.switchbool.currentbool;
end

FNS_AxesPlot(handles);

guidata(hObject, handles);



% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% BUTTON2!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);
FNS_AxesPlot(handles);

guidata(hObject, handles);


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% BUTTON3!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);
guidata(hObject, handles);


% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% BUTTON4!
state=get(hObject,'value');
```



```
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);


% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
% BUTTON5!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);


% --- Executes on button press in pushbutton8.
function pushbutton8_Callback(hObject, eventdata, handles)
% BUTTON6!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);


% --- Executes on button press in pushbutton9.
function pushbutton9_Callback(hObject, eventdata, handles)
% BUTTON7!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);


% --- Executes on button press in pushbutton10.
function pushbutton10_Callback(hObject, eventdata, handles)
% BUTTON8!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);

% --- Executes on button press in pushbutton11.
```

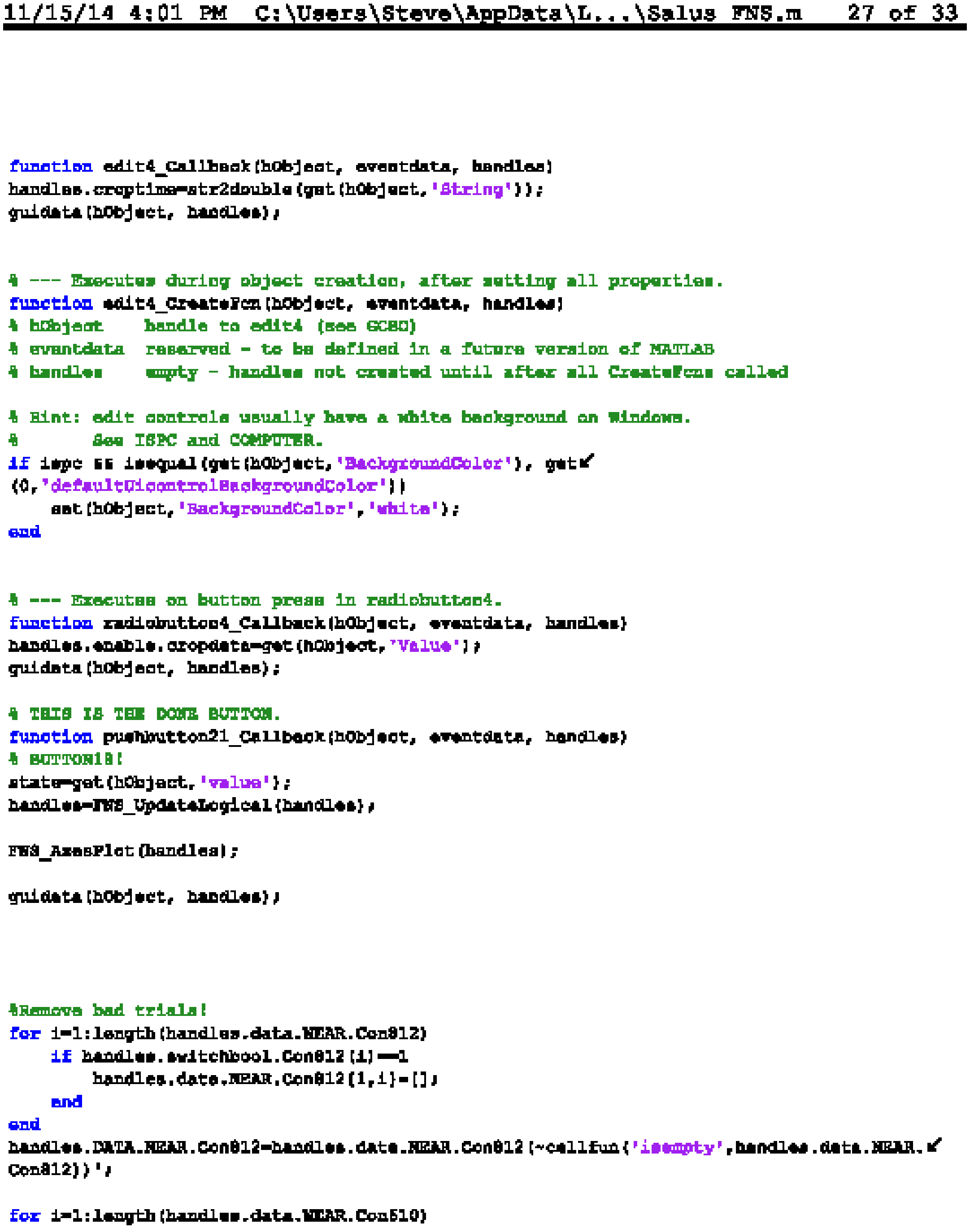
```
function pushbutton11_Callback(hObject, eventdata, handles)
% BUTTON9!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);

% --- Executes on button press in pushbutton12.
function pushbutton12_Callback(hObject, eventdata, handles)
% BUTTON10!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);


% --- Executes on button press in pushbutton13.
function pushbutton13_Callback(hObject, eventdata, handles)
% BUTTON11!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);


% --- Executes on button press in pushbutton14.
function pushbutton14_Callback(hObject, eventdata, handles)
% BUTTON12!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);


% --- Executes on button press in pushbutton15.
function pushbutton15_Callback(hObject, eventdata, handles)
% BUTTON13!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);
```



11/15/14 4:01 PM C:\Users\Steve\AppData\L...\Salus FNS.m 24 of 33

```
% --- Executes on button press in pushbutton16.
function pushbutton16_Callback(hObject, eventdata, handles)
% BUTTON14!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);


% --- Executes on button press in pushbutton17.
function pushbutton17_Callback(hObject, eventdata, handles)
% BUTTON15!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);


% --- Executes on button press in pushbutton18.
function pushbutton18_Callback(hObject, eventdata, handles)
% BUTTON16!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);


% --- Executes on button press in pushbutton19.
function pushbutton19_Callback(hObject, eventdata, handles)
% BUTTON17!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);



function edit1_Callback(hObject, eventdata, handles)
handles.filter.fo=str2double(get(hObject,'String'));
guidata(hObject, handles);
```

```
% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton22.
function pushbutton22_Callback(hObject, eventdata, handles)
handles
assignin('base','handles',handles);

% hObject    handle to pushbutton22 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on selection change in popupmenu3.
function popupmenu3_Callback(hObject, eventdata, handles)
contents = cellstr(get(hObject,'String'));
handles.currenttrial=contents{get(hObject,'Value')};
handles.switchbool.currentbool=handles.switchbool.(handles.currenttrial);
guidata(hObject, handles);



% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu3 contents as cell↙
array
%        contents{get(hObject,'Value')} returns selected item from popupmenu3


% --- Executes during object creation, after setting all properties.
function popupmenu3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes when selected object is changed in uipanel1.
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
% hObject    handle to the selected object in uipanel1
% eventdata  structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if none was selected
%   NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in radiobutton1.
function radiobutton1_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton1 (see GCBO)
handles.enable.DC=get(hObject,'Value');
guidata(hObject, handles);


% --- Executes on button press in radiobutton2.
function radiobutton2_Callback(hObject, eventdata, handles)
handles.enable.filter=get(hObject,'Value');
guidata(hObject, handles);


% --- Executes on button press in radiobutton3.
function radiobutton3_Callback(hObject, eventdata, handles)
handles.enable.blink=get(hObject,'Value');
guidata(hObject, handles);



function edit9_Callback(hObject, eventdata, handles)
handles.filter.order=str2double(get(hObject,'String'));
guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```



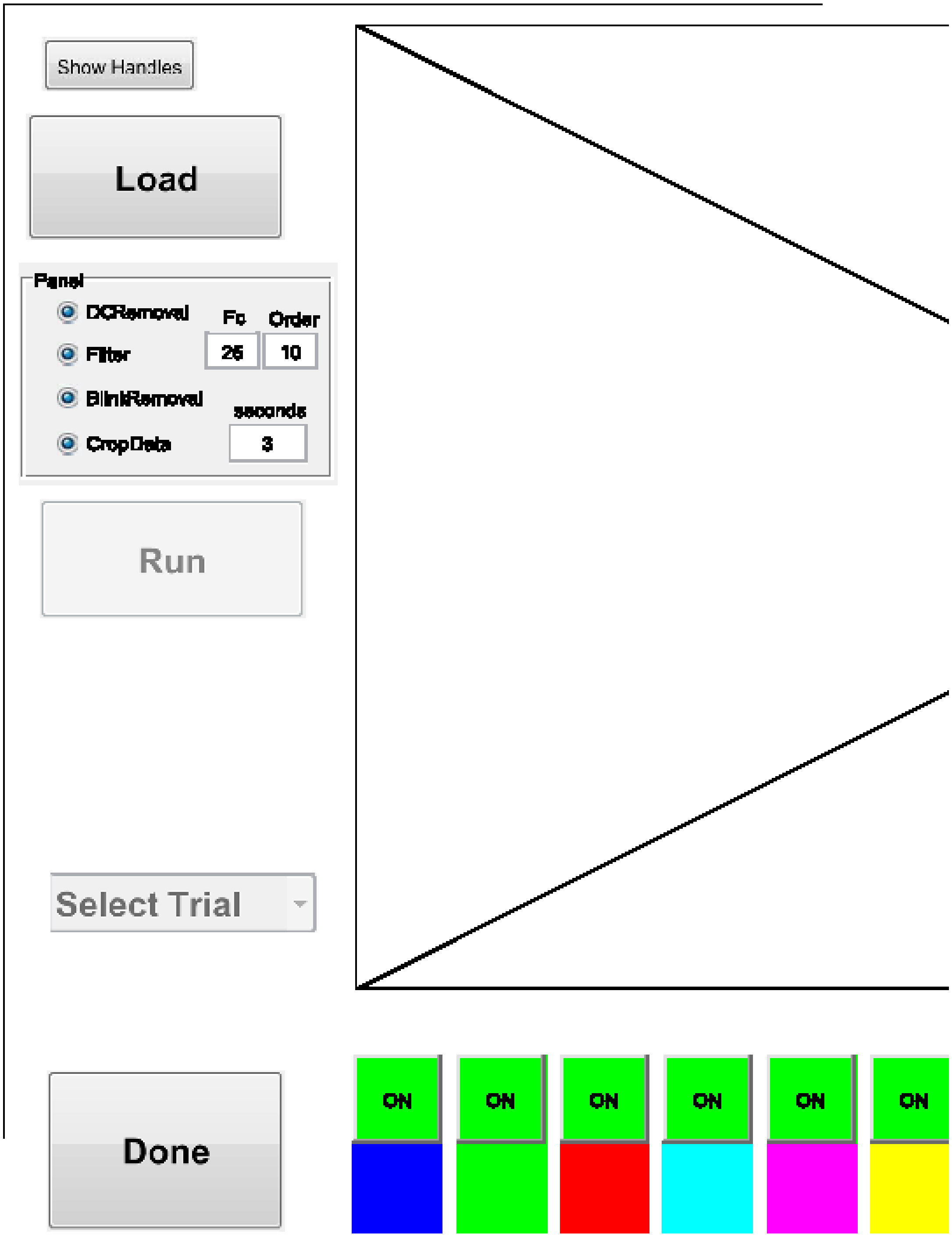
```
function edit4_Callback(hObject, eventdata, handles)
handles.croptime=str2double(get(hObject,'String'));
guidata(hObject, handles);


% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in radiobutton4.
function radiobutton4_Callback(hObject, eventdata, handles)
handles.enable.cropdata=get(hObject,'Value');
guidata(hObject, handles);

% THIS IS THE DONE BUTTON.
function pushbutton21_Callback(hObject, eventdata, handles)
% BUTTON18!
state=get(hObject,'value');
handles=FNS_UpdateLogical(handles);

FNS_AxesPlot(handles);

guidata(hObject, handles);




%Remove bad trials!
for i=1:length(handles.data.NEAR.Con812)
    if handles.switchbool.Con812(i)==1
        handles.data.NEAR.Con812{1,i}=[];
    end
end
handles.DATA.NEAR.Con812=handles.data.NEAR.Con812(~cellfun('isempty',handles.data.NEAR.↙
Con812))';

for i=1:length(handles.data.NEAR.Con510)
```

```
    if handles.switchbool.Con610(i)==1
        handles.data.NEAR.Con610{1,i}=[];
    end
end
handles.DATA.NEAR.Con610=handles.data.NEAR.Con610(~cellfun('isempty',handles.data.NEAR.↙
Con610))';

for i=1:length(handles.data.NEAR.Con612)
    if handles.switchbool.Con612(i)==1
        handles.data.NEAR.Con612{1,i}=[];
    end
end
handles.DATA.NEAR.Con612=handles.data.NEAR.Con612(~cellfun('isempty',handles.data.NEAR.↙
Con612))';

for i=1:length(handles.data.NEAR.Div128)
    if handles.switchbool.Div128(i)==1
        handles.data.NEAR.Div128{1,i}=[];
    end
end
handles.DATA.NEAR.Div128=handles.data.NEAR.Div128(~cellfun('isempty',handles.data.NEAR.↙
Div128))';

for i=1:length(handles.data.NEAR.Div106)
    if handles.switchbool.Div106(i)==1
        handles.data.NEAR.Div106{1,i}=[];
    end
end
handles.DATA.NEAR.Div106=handles.data.NEAR.Div106(~cellfun('isempty',handles.data.NEAR.↙
Div106))';

for i=1:length(handles.data.NEAR.Div126)
    if handles.switchbool.Div126(i)==1
        handles.data.NEAR.Div126{1,i}=[];
    end
end
handles.DATA.NEAR.Div126=handles.data.NEAR.Div126(~cellfun('isempty',handles.data.NEAR.↙
Div126))';

% Far
for i=1:length(handles.data.FAR.Con48)
    if handles.switchbool.Con48(i)==1
        handles.data.FAR.Con48{1,i}=[];
    end
end
handles.DATA.FAR.Con48=handles.data.FAR.Con48(~cellfun('isempty',handles.data.FAR.↙
Con48))';

for i=1:length(handles.data.FAR.Con26)
    if handles.switchbool.Con26(i)==1
        handles.data.FAR.Con26{1,i}=[];
    end
end
handles.DATA.FAR.Con26=handles.data.FAR.Con26(~cellfun('isempty',handles.data.FAR. ↙
Con26))';

for i=1:length(handles.data.FAR.Con28)
    if handles.switchbool.Con28(i)==1
        handles.data.FAR.Con28{1,i}=[];
    end
end
handles.DATA.FAR.Con28=handles.data.FAR.Con28(~cellfun('isempty',handles.data.FAR. ↙
Con28))';

for i=1:length(handles.data.FAR.Div84)
    if handles.switchbool.Div84(i)==1
        handles.data.FAR.Div84{1,i}=[];
    end
end
handles.DATA.FAR.Div84=handles.data.FAR.Div84(~cellfun('isempty',handles.data.FAR. ↙
Div84))';

for i=1:length(handles.data.FAR.Div62)
    if handles.switchbool.Div62(i)==1
        handles.data.FAR.Div62{1,i}=[];
    end
end
handles.DATA.FAR.Div62=handles.data.FAR.Div62(~cellfun('isempty',handles.data.FAR. ↙
Div62))';

for i=1:length(handles.data.FAR.Div82)
    if handles.switchbool.Div82(i)==1
        handles.data.FAR.Div82{1,i}=[];
    end
end
handles.DATA.FAR.Div82=handles.data.FAR.Div82(~cellfun('isempty',handles.data.FAR. ↙
Div82))';

%SAC
for i=1:length(handles.data.SAC.M2R5)
    if handles.switchbool.M2R5(i)==1
        handles.data.SAC.M2R5{1,i}=[];
    end
end
handles.DATA.SAC.M2R5=handles.data.SAC.M2R5(~cellfun('isempty',handles.data.SAC.M2R5))';

for i=1:length(handles.data.SAC.R2M5)
    if handles.switchbool.R2M5(i)==1
        handles.data.SAC.R2M5{1,i}=[];
    end
end
handles.DATA.SAC.R2M5=handles.data.SAC.R2M5(~cellfun('isempty',handles.data.SAC.R2M5))';
```

```
for i=1:length(handles.data.SAC.M2L5)
    if handles.switchbool.M2L5(i)==1
        handles.data.SAC.M2L5{1,i}=[];
    end
end
handles.DATA.SAC.M2L5=handles.data.SAC.M2L5(~cellfun('isempty',handles.data.SAC.M2L5))';

for i=1:length(handles.data.SAC.L2M5)
    if handles.switchbool.L2M5(i)==1
        handles.data.SAC.L2M5{1,i}=[];
    end
end
handles.DATA.SAC.L2M5=handles.data.SAC.L2M5(~cellfun('isempty',handles.data.SAC.L2M5))';

for i=1:length(handles.data.SAC.M2R10)
    if handles.switchbool.M2R10(i)==1
        handles.data.SAC.M2R10{1,i}=[];
    end
end
handles.DATA.SAC.M2R10=handles.data.SAC.M2R10(~cellfun('isempty',handles.data.SAC. ↙
M2R10))';

for i=1:length(handles.data.SAC.R2M10)
    if handles.switchbool.R2M10(i)==1
        handles.data.SAC.R2M10{1,i}=[];
    end
end
handles.DATA.SAC.R2M10=handles.data.SAC.R2M10(~cellfun('isempty',handles.data.SAC. ↙
R2M10))';

for i=1:length(handles.data.SAC.M2L10)
    if handles.switchbool.M2L10(i)==1
        handles.data.SAC.M2L10{1,i}=[];
    end
end
handles.DATA.SAC.M2L10=handles.data.SAC.M2L10(~cellfun('isempty',handles.data.SAC. ↙
M2L10))';

for i=1:length(handles.data.SAC.L2M10)
    if handles.switchbool.L2M10(i)==1
        handles.data.SAC.L2M10{1,i}=[];
    end
end
handles.DATA.SAC.L2M10=handles.data.SAC.L2M10(~cellfun('isempty',handles.data.SAC. ↙
L2M10))';

handles.data.currentdata=[];
assignin('base','DATA',handles.DATA);

handles=SalusPlots(handles.DATA.FAR.Con48,1,    'Conv 04 to 08',[0 5],[0 10],[-1 1],50,0, ↙
handles);
handles=SalusPlots(handles.DATA.FAR.Con26,2,    'Conv 02 to 06',[0 5],[0 10],[-1 1],50,0,↙
handles);
handles=SalusPlots(handles.DATA.FAR.Con28,3,    'Conv 02 to 08',[0 5],[0 10],[-1 1],50,0,↙
handles);
handles=SalusPlots(handles.DATA.FAR.Div84,4,    'Div  08 to 04',[0 5],[0 10],[-1 1],50,0,↙
handles);
handles=SalusPlots(handles.DATA.FAR.Div62,5,    'Div  06 to 02',[0 5],[0 10],[-1 1],50,0,↙
handles);
handles=SalusPlots(handles.DATA.FAR.Div82,6,    'Div  08 to 02',[0 5],[0 10],[-1 1],50,0,↙
handles);

handles=SalusPlots(handles.DATA.NEAR.Con812,7, 'Conv 08 to 12',[0 10],[5 15],[-1 1],50,0,↙
handles);
handles=SalusPlots(handles.DATA.NEAR.Con610,8, 'Conv 06 to 10',[0 10],[5 15],[-1 1],50,0,↙
handles);
handles=SalusPlots(handles.DATA.NEAR.Con612,9, 'Conv 06 to 12',[0 10],[5 15],[-1 1],50,0,↙
handles);
handles=SalusPlots(handles.DATA.NEAR.Div128,10,'Div  12 to 08',[0 10],[5 15],[-1 1],50,0,↙
handles);
handles=SalusPlots(handles.DATA.NEAR.Div106,11,'Div  10 to 06',[0 10],[5 15],[-1 1],50,0,↙
handles);
handles=SalusPlots(handles.DATA.NEAR.Div126,12,'Div  12 to 06',[0 10],[5 15],[-1 1],50,0,↙
handles);

handles=SalusPlots(handles.DATA.SAC.M2R5,13,    'Mid   to Right 05',[-2 7],[-1 1],[-2 7],↙
200,1,handles);
handles=SalusPlots(handles.DATA.SAC.R2M5,14,    'Right to Mid   05',[-2 7],[-1 1],[-2 7],↙
200,1,handles);
handles=SalusPlots(handles.DATA.SAC.M2L5,15,    'Mid   to Left  05',[-7 2],[-1 1],[-7 2],↙
200,1,handles);
handles=SalusPlots(handles.DATA.SAC.L2M5,16,    'Left  to Mid   05',[-7 2],[-1 1],[-7 2],↙
200,1,handles);
handles=SalusPlots(handles.DATA.SAC.M2R10,17,   'Mid   to Right 10',[-2 12],[-1 1],[-2↙
12],200,1,handles);
handles=SalusPlots(handles.DATA.SAC.R2M10,18,   'Right to Mid   10',[-2 12],[-1 1],[-2↙
12],200,1,handles);
handles=SalusPlots(handles.DATA.SAC.M2L10,19,   'Mid   to Left  10',[-12 2],[-1 1],[-12↙
2],200,1,handles);
handles=SalusPlots(handles.DATA.SAC.L2M10,20,   'Left  to Mid   10',[-12 2],[-1 1],[-12↙
2],200,1,handles);

assignin('base','DATA',handles.DATA);

save(['FinalData' handles.ID.SubjInit])

if exist([handles.ID.SubjInit 'Plots']) == 7
    disp('WARNING: A Directory of Plots with conflicting name exists. Please rename↙
Directory.')
else
    mkdir('eps')
```

```
cd('eps')
hgexport(1,'Conv4-8')
hgexport(2,'Conv2-6')
hgexport(3,'Conv2-8')
hgexport(4,'Div8-4')
hgexport(5,'Div6-2')
hgexport(6,'Div8-2')

hgexport(7,'Conv8-12')
hgexport(8,'Conv6-10')
hgexport(9,'Conv6-12')
hgexport(10,'Div12-8')
hgexport(11,'Div10-6')
hgexport(12,'Div12-6')

hgexport(13,'M2R5')
hgexport(14,'R2M5')
hgexport(15,'M2L5')
hgexport(16,'L2M5')
hgexport(17,'M2R10')
hgexport(18,'R2M10')
hgexport(19,'M2L10')
hgexport(20,'L2M10')

cd ..
mkdir('jpeg')
cd('jpeg')

print(1,'-djpeg','-r300','Conv4-8')
print(2,'-djpeg','-r300','Conv2-6')
print(3,'-djpeg','-r300','Conv2-8')
print(4,'-djpeg','-r300','Div8-4')
print(5,'-djpeg','-r300','Div6-2')
print(6,'-djpeg','-r300','Div8-2')

print(7,'-djpeg','-r300','Conv8-12')
print(8,'-djpeg','-r300','Conv6-10')
print(9,'-djpeg','-r300','Conv6-12')
print(10,'-djpeg','-r300','Div12-8')
print(11,'-djpeg','-r300','Div10-6')
print(12,'-djpeg','-r300','Div12-6')

print(13,'-djpeg','-r300','M2R5')
print(14,'-djpeg','-r300','R2M5')
print(15,'-djpeg','-r300','M2L5')
print(16,'-djpeg','-r300','L2M5')
print(17,'-djpeg','-r300','M2R10')
print(18,'-djpeg','-r300','R2M10')
print(19,'-djpeg','-r300','M2L10')
print(20,'-djpeg','-r300','L2M10')
cd ..
```

```
    cd ..
    close(1:20)
end

guidata(hObject, handles);
```



Panel

○ DCRemoval

○ Filter

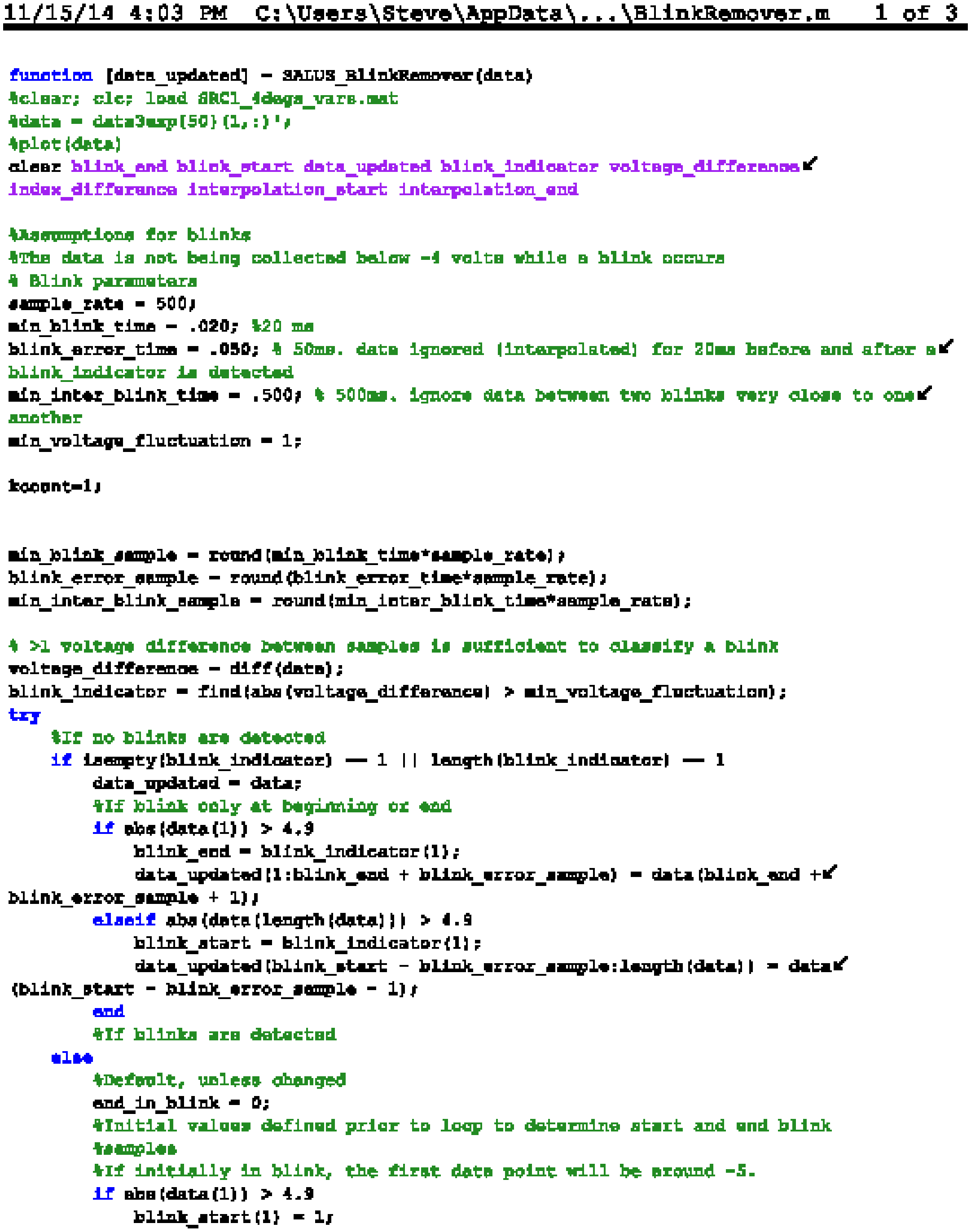
○ BlinkRemoval

○ CropData

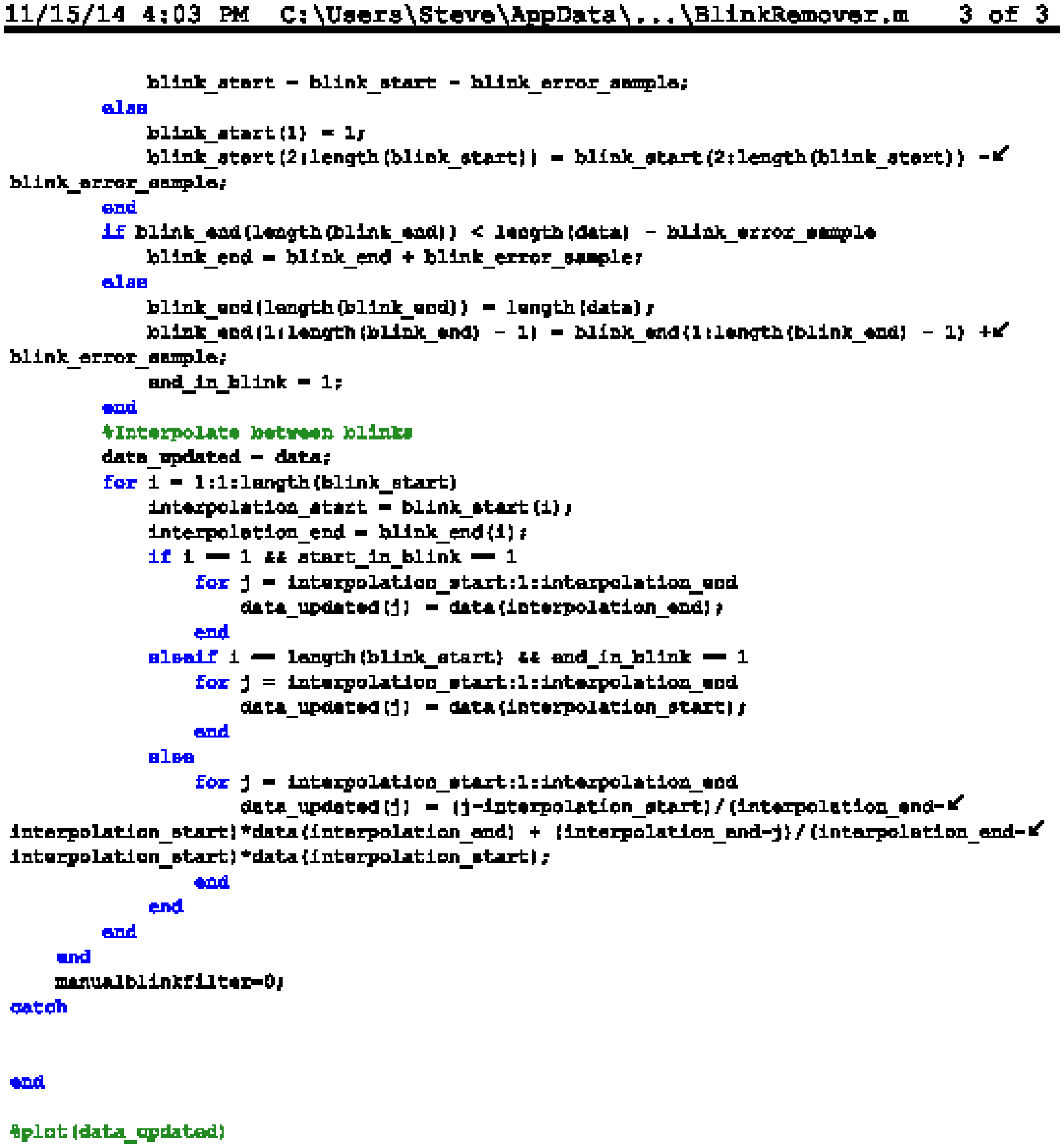
Fc: 25

Order: 10

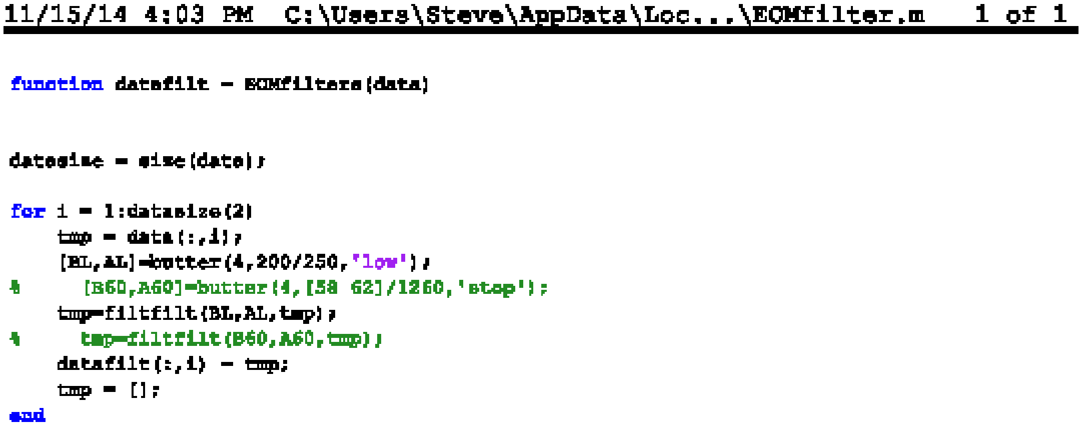
seconds: 3



```
11/15/14 4:03 PM  C:\Users\Steve\AppData\...\BlinkRemover.m    1 of 3

function [data_updated] = SALUS_BlinkRemover(data)
%clear; clc; load SRC1_4degs_vars.mat
%data = data3exp{50}(1,:)';
%plot(data)
clear blink_end blink_start data_updated blink_indicator voltage_difference↙
index_difference interpolation_start interpolation_end

%Assumptions for blinks
%The data is not being collected below -4 volts while a blink occurs
% Blink parameters
sample_rate = 500;
min_blink_time = .020; %20 ms
blink_error_time = .050; % 50ms. data ignored (interpolated) for 20ms before and after a↙
blink_indicator is detected
min_inter_blink_time = .500; % 500ms. ignore data between two blinks very close to one↙
another
min_voltage_fluctuation = 1;

kcount=1;


min_blink_sample = round(min_blink_time*sample_rate);
blink_error_sample = round(blink_error_time*sample_rate);
min_inter_blink_sample = round(min_inter_blink_time*sample_rate);

% >1 voltage difference between samples is sufficient to classify a blink
voltage_difference = diff(data);
blink_indicator = find(abs(voltage_difference) > min_voltage_fluctuation);
try
    %If no blinks are detected
    if isempty(blink_indicator) == 1 || length(blink_indicator) == 1
        data_updated = data;
        %If blink only at beginning or end
        if abs(data(1)) > 4.9
            blink_end = blink_indicator(1);
            data_updated(1:blink_end + blink_error_sample) = data(blink_end +↙
blink_error_sample + 1);
        elseif abs(data(length(data))) > 4.9
            blink_start = blink_indicator(1);
            data_updated(blink_start - blink_error_sample:length(data)) = data↙
(blink_start - blink_error_sample - 1);
        end
        %If blinks are detected
    else
        %Default, unless changed
        end_in_blink = 0;
        %Initial values defined prior to loop to determine start and end blink
        %samples
        %If initially in blink, the first data point will be around -5.
        if abs(data(1)) > 4.9
            blink_start(1) = 1;
            blink_end(1) = blink_indicator(1);
            in_blink = 0;
            blink_number = 2;
            start_in_blink = 1;
        else
            blink_start(1) = blink_indicator(1);
            in_blink = 1;
            blink_number = 1;
            start_in_blink = 0;
        end

        %Sample separation between 1V/sample fluctuations
        index_difference = diff(blink_indicator);

        %Start loop to define start and end blink samples
        for i = 1:1:length(index_difference)
            %If a blink is in progress, then end it if next fluctuation falls
            %within declared temporal range of a blink
            %Before 7/24/13 first conditional: index_difference(i) > min_blink_sample &&↙
in_blink == 1
            if in_blink == 1
                blink_end(blink_number) = blink_indicator(i+1);
                blink_number = blink_number + 1;
                in_blink = 0;
                % Redefine the end of a blink if a new fluctuation was observed
                % before the minimum inter-blink time was reached
            elseif index_difference(i) <= min_inter_blink_sample && in_blink == 0
                blink_end(blink_number-1) = blink_indicator(i+1);
                %A new blink is defined when the next fluctuation is observed
            elseif index_difference(i) > min_inter_blink_sample && in_blink == 0  && i ~=↙
length(index_difference)
                blink_start(blink_number) = blink_indicator(i+1);
                in_blink = 1;
                %If it's the final sample and the subject is defined to be in a
                %blink, then the last blink ends at the end of the recording
            elseif index_difference(i) > min_inter_blink_sample && in_blink == 0  && i ==↙
length(index_difference)
                blink_start(blink_number) = blink_indicator(i+1);
                blink_end(blink_number) = length(data);
                end_in_blink = 1;
            elseif in_blink == 1 && i == length(index_difference)
                blink_end(blink_number) = blink_indicator(i+1);
                blink_number = blink_number + 1;
                in_blink = 0;
            end
        end
        %The region of interpolation is extended beyond the voltage fluctuation
        %to account for untrustworthy data surrounding a blink
        %The following conditional statements take into account blinks at the
        %beginning or end of the recording.
        if blink_start(1) > blink_error_sample
```

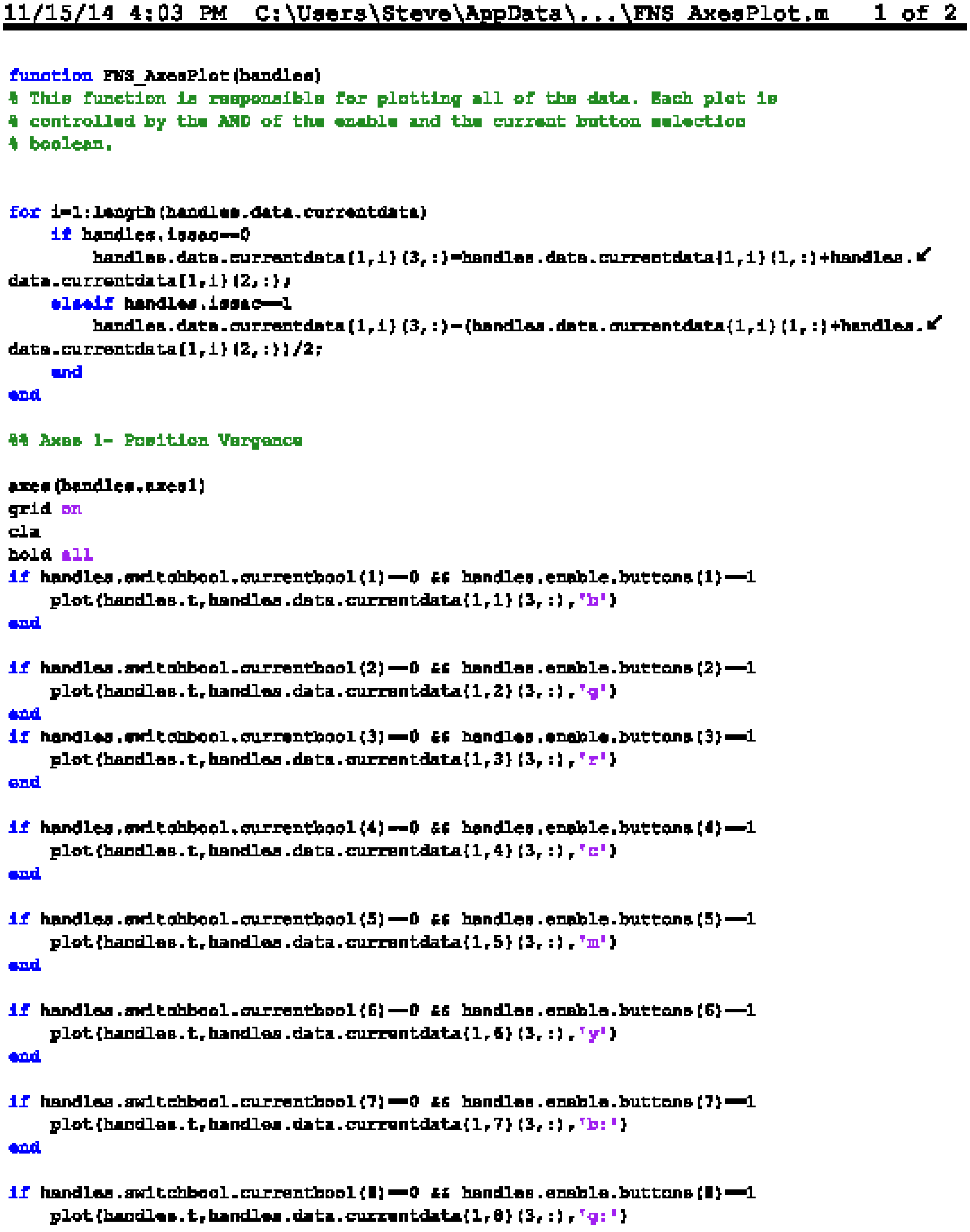


```
            blink_start = blink_start - blink_error_sample;
        else
            blink_start(1) = 1;
            blink_start(2:length(blink_start)) = blink_start(2:length(blink_start)) -↙
blink_error_sample;
        end
        if blink_end(length(blink_end)) < length(data) - blink_error_sample
            blink_end = blink_end + blink_error_sample;
        else
            blink_end(length(blink_end)) = length(data);
            blink_end(1:length(blink_end) - 1) = blink_end(1:length(blink_end) - 1) +↙
blink_error_sample;
            end_in_blink = 1;
        end
        %Interpolate between blinks
        data_updated = data;
        for i = 1:1:length(blink_start)
            interpolation_start = blink_start(i);
            interpolation_end = blink_end(i);
            if i == 1 && start_in_blink == 1
                for j = interpolation_start:1:interpolation_end
                    data_updated(j) = data(interpolation_end);
                end
            elseif i == length(blink_start) && end_in_blink == 1
                for j = interpolation_start:1:interpolation_end
                    data_updated(j) = data(interpolation_start);
                end
            else
                for j = interpolation_start:1:interpolation_end
                    data_updated(j) = (j-interpolation_start)/(interpolation_end-↙
interpolation_start)*data(interpolation_end) + (interpolation_end-j)/(interpolation_end-↙
interpolation_start)*data(interpolation_start);
                end
            end
        end
    end
    manualblinkfilter=0;
catch


end

%plot(data_updated)
```



11/15/14 4:03 PM C:\Users\Steve\AppData\Loc...\EOMfilter.m   1 of 1

```
function datafilt = EOMfilters(data)


datasize = size(data);

for i = 1:datasize(2)
     tmp = data(:,i);
     [BL,AL]=butter(4,200/250,'low');
%       [B60,A60]=butter(4,[58 62]/1260,'stop');
     tmp=filtfilt(BL,AL,tmp);
%      tmp=filtfilt(B60,A60,tmp);
     datafilt(:,i) = tmp;
     tmp = [];
end
```



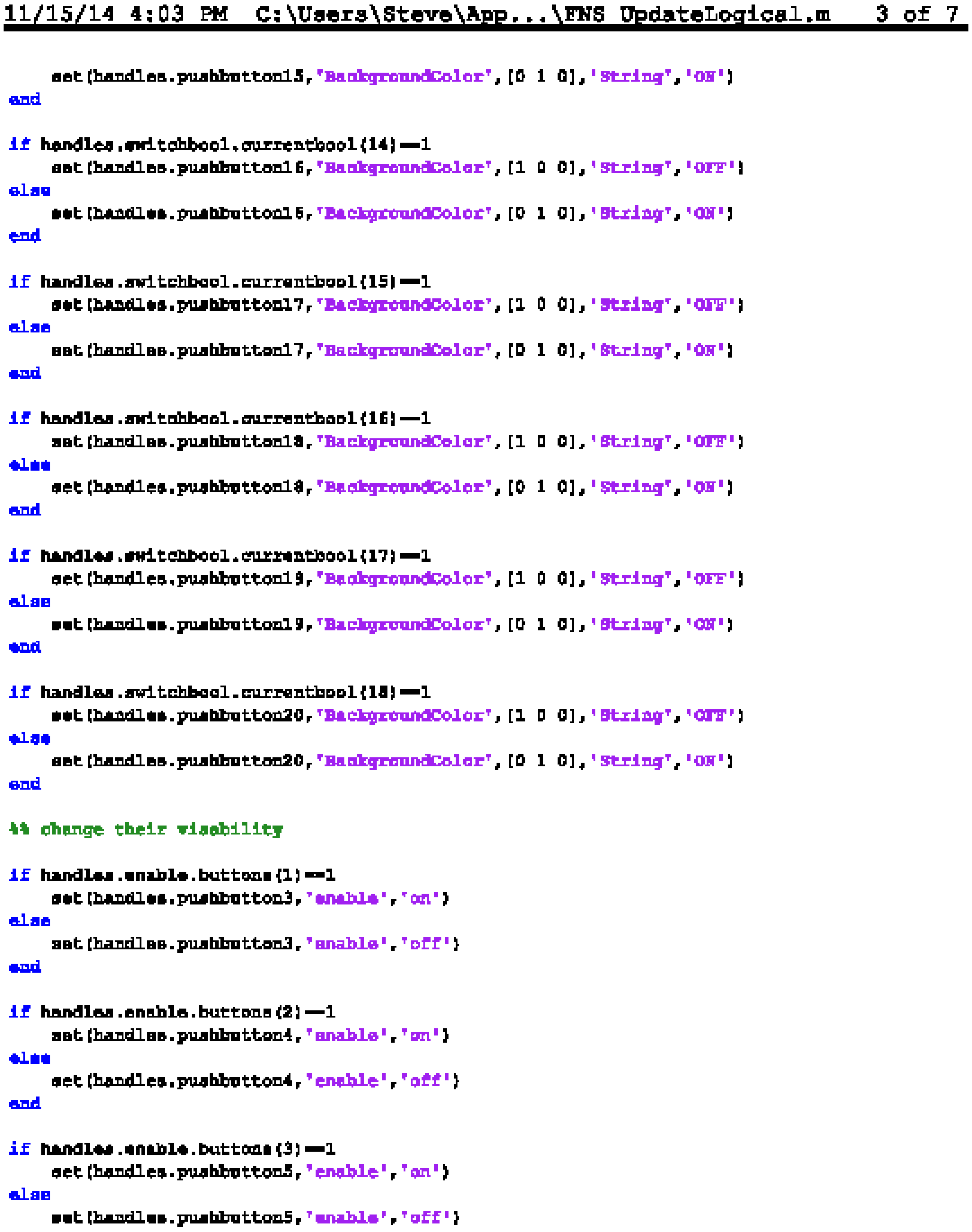
11/15/14 4:03 PM C:\Users\Steve\AppData\...\FNS AxesPlot.m   1 of 2

```
function FNS_AxesPlot(handles)
% This function is responsible for plotting all of the data. Each plot is
% controlled by the AND of the enable and the current button selection
% boolean.


for i=1:length(handles.data.currentdata)
    if handles.issac==0
        handles.data.currentdata{1,i}(3,:)=handles.data.currentdata{1,i}(1,:)+handles.↙
data.currentdata{1,i}(2,:);
    elseif handles.issac==1
        handles.data.currentdata{1,i}(3,:)=(handles.data.currentdata{1,i}(1,:)+handles.↙
data.currentdata{1,i}(2,:))/2;
    end
end

%% Axes 1- Position Vergence

axes(handles.axes1)
grid on
cla
hold all
if handles.switchbool.currentbool(1)==0 && handles.enable.buttons(1)==1
    plot(handles.t,handles.data.currentdata{1,1}(3,:),'b')
end

if handles.switchbool.currentbool(2)==0 && handles.enable.buttons(2)==1
    plot(handles.t,handles.data.currentdata{1,2}(3,:),'g')
end
if handles.switchbool.currentbool(3)==0 && handles.enable.buttons(3)==1
    plot(handles.t,handles.data.currentdata{1,3}(3,:),'r')
end

if handles.switchbool.currentbool(4)==0 && handles.enable.buttons(4)==1
    plot(handles.t,handles.data.currentdata{1,4}(3,:),'c')
end

if handles.switchbool.currentbool(5)==0 && handles.enable.buttons(5)==1
    plot(handles.t,handles.data.currentdata{1,5}(3,:),'m')
end

if handles.switchbool.currentbool(6)==0 && handles.enable.buttons(6)==1
    plot(handles.t,handles.data.currentdata{1,6}(3,:),'y')
end

if handles.switchbool.currentbool(7)==0 && handles.enable.buttons(7)==1
    plot(handles.t,handles.data.currentdata{1,7}(3,:),'b:')
end

if handles.switchbool.currentbool(8)==0 && handles.enable.buttons(8)==1
    plot(handles.t,handles.data.currentdata{1,8}(3,:),'g:')
```

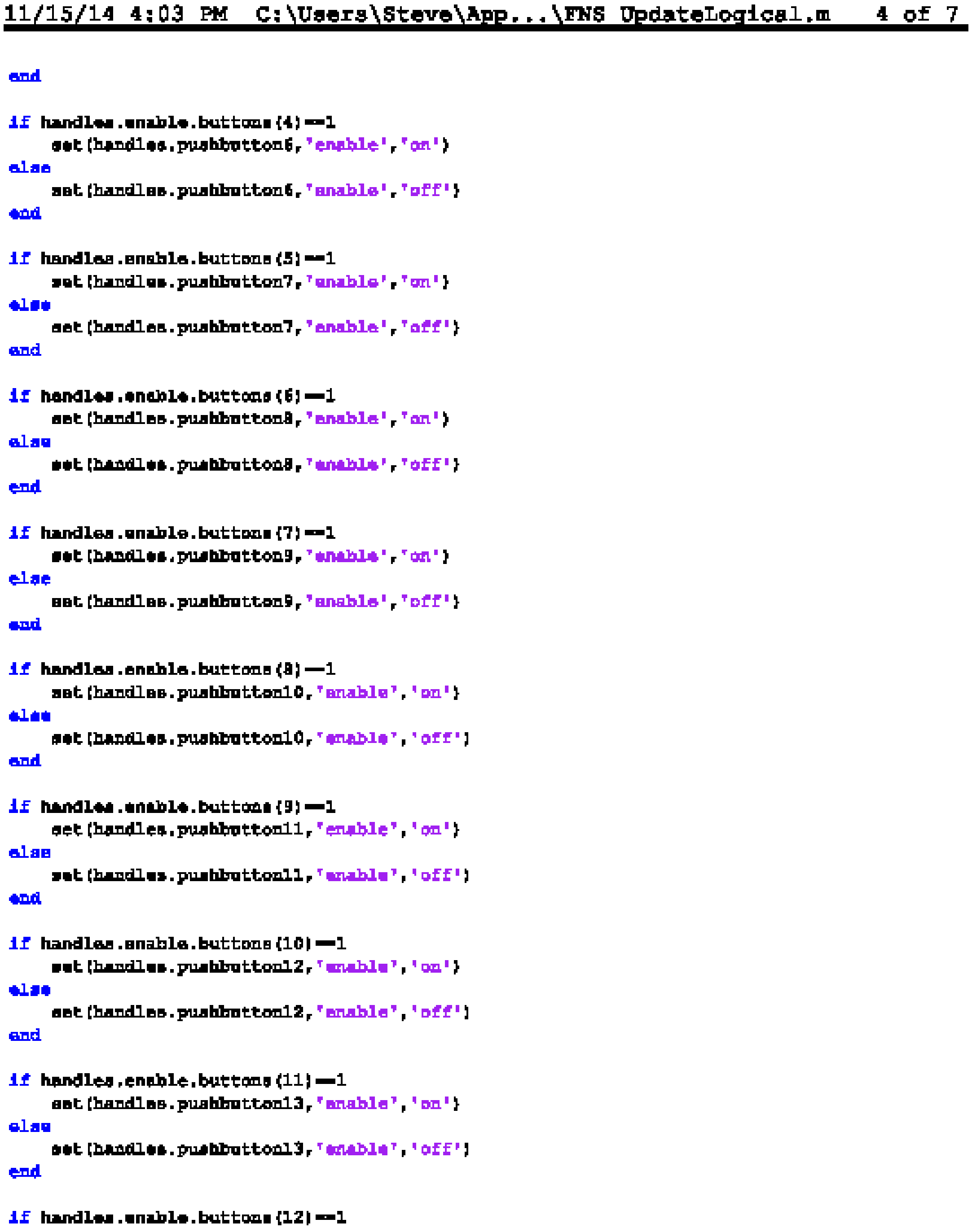
```
end

if handles.switchbool.currentbool(9)==0 && handles.enable.buttons(9)==1
    plot(handles.t,handles.data.currentdata{1,9}(3,:),'r:')
end

if handles.switchbool.currentbool(10)==0 && handles.enable.buttons(10)==1
    plot(handles.t,handles.data.currentdata{1,10}(3,:),'c:')
end

if handles.switchbool.currentbool(11)==0 && handles.enable.buttons(11)==1
    plot(handles.t,handles.data.currentdata{1,11}(3,:),'m:')
end

if handles.switchbool.currentbool(12)==0 && handles.enable.buttons(12)==1
    plot(handles.t,handles.data.currentdata{1,12}(3,:),'y:')
end

if handles.switchbool.currentbool(13)==0 && handles.enable.buttons(13)==1
    plot(handles.t,handles.data.currentdata{1,12}(3,:),'b--')
end

if handles.switchbool.currentbool(14)==0 && handles.enable.buttons(14)==1
    plot(handles.t,handles.data.currentdata{1,12}(3,:),'g--')
end

if handles.switchbool.currentbool(15)==0 && handles.enable.buttons(15)==1
    plot(handles.t,handles.data.currentdata{1,12}(3,:),'r--')
end

if handles.switchbool.currentbool(16)==0 && handles.enable.buttons(16)==1
    plot(handles.t,handles.data.currentdata{1,12}(3,:),'c--')
end

if handles.switchbool.currentbool(17)==0 && handles.enable.buttons(17)==1
    plot(handles.t,handles.data.currentdata{1,12}(3,:),'m--')
end

if handles.switchbool.currentbool(18)==0 && handles.enable.buttons(18)==1
    plot(handles.t,handles.data.currentdata{1,12}(3,:),'y--')
end

if handles.issac==1
    xlim([0 1.5]);
end
```

```
function handlesout=FNS_UpdateLogical(handles)
%This function is responsible for the control of the buttons. It will first
%read all the buttons to see their value, then update the boolean it
%belongs to.

%% Update all the values in the bools
handles.switchbool.currentbool(1)=get(handles.pushbutton3,'value');
handles.switchbool.currentbool(2)=get(handles.pushbutton4,'value');
handles.switchbool.currentbool(3)=get(handles.pushbutton5,'value');
handles.switchbool.currentbool(4)=get(handles.pushbutton6,'value');
handles.switchbool.currentbool(5)=get(handles.pushbutton7,'value');
handles.switchbool.currentbool(6)=get(handles.pushbutton8,'value');
handles.switchbool.currentbool(7)=get(handles.pushbutton9,'value');
handles.switchbool.currentbool(8)=get(handles.pushbutton10,'value');
handles.switchbool.currentbool(9)=get(handles.pushbutton11,'value');
handles.switchbool.currentbool(10)=get(handles.pushbutton12,'value');
handles.switchbool.currentbool(11)=get(handles.pushbutton13,'value');
handles.switchbool.currentbool(12)=get(handles.pushbutton14,'value');
handles.switchbool.currentbool(13)=get(handles.pushbutton15,'value');
handles.switchbool.currentbool(14)=get(handles.pushbutton16,'value');
handles.switchbool.currentbool(15)=get(handles.pushbutton17,'value');
handles.switchbool.currentbool(16)=get(handles.pushbutton18,'value');
handles.switchbool.currentbool(17)=get(handles.pushbutton19,'value');
handles.switchbool.currentbool(18)=get(handles.pushbutton20,'value');

%% Physically change the appearances of the togglebuttons

if handles.switchbool.currentbool(1)==1
    set(handles.pushbutton3,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton3,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(2)==1
    set(handles.pushbutton4,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton4,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(3)==1
    set(handles.pushbutton5,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton5,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(4)==1
    set(handles.pushbutton6,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton6,'BackgroundColor',[0 1 0],'String','ON')
end
```

```
if handles.switchbool.currentbool(5)==1
    set(handles.pushbutton7,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton7,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(6)==1
    set(handles.pushbutton8,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton8,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(7)==1
    set(handles.pushbutton9,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton9,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(8)==1
    set(handles.pushbutton10,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton10,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(9)==1
    set(handles.pushbutton11,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton11,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(10)==1
    set(handles.pushbutton12,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton12,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(11)==1
    set(handles.pushbutton13,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton13,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(12)==1
    set(handles.pushbutton14,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton14,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(13)==1
    set(handles.pushbutton15,'BackgroundColor',[1 0 0],'String','OFF')
else
```



```
    set(handles.pushbutton15,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(14)==1
    set(handles.pushbutton16,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton16,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(15)==1
    set(handles.pushbutton17,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton17,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(16)==1
    set(handles.pushbutton18,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton18,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(17)==1
    set(handles.pushbutton19,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton19,'BackgroundColor',[0 1 0],'String','ON')
end

if handles.switchbool.currentbool(18)==1
    set(handles.pushbutton20,'BackgroundColor',[1 0 0],'String','OFF')
else
    set(handles.pushbutton20,'BackgroundColor',[0 1 0],'String','ON')
end

%% change their visability

if handles.enable.buttons(1)==1
    set(handles.pushbutton3,'enable','on')
else
    set(handles.pushbutton3,'enable','off')
end

if handles.enable.buttons(2)==1
    set(handles.pushbutton4,'enable','on')
else
    set(handles.pushbutton4,'enable','off')
end

if handles.enable.buttons(3)==1
    set(handles.pushbutton5,'enable','on')
else
    set(handles.pushbutton5,'enable','off')
```



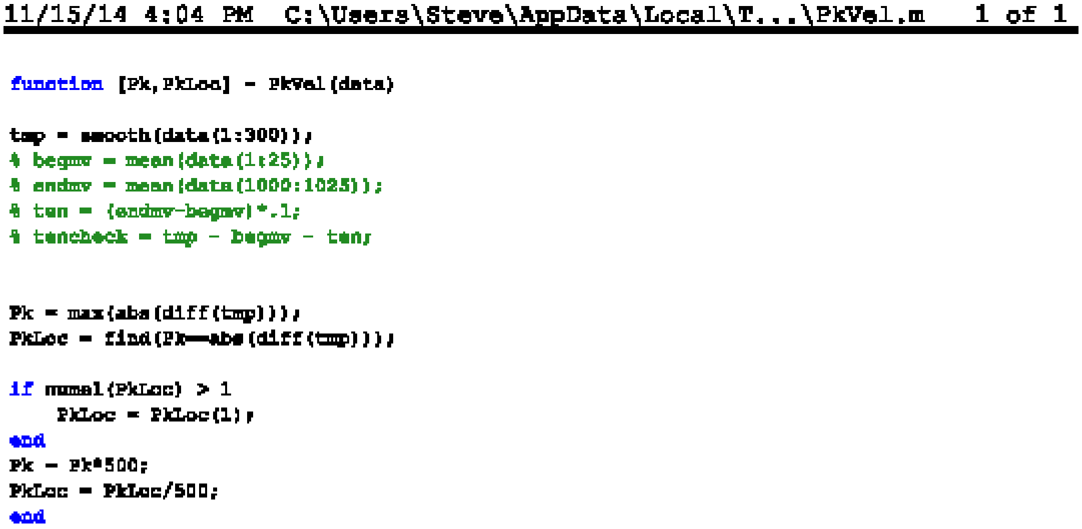
11/15/14 4:03 PM C:\Users\Steve\App...\FNS UpdateLogical.m 4 of 7

```
end

if handles.enable.buttons(4)==1
    set(handles.pushbutton6,'enable','on')
else
    set(handles.pushbutton6,'enable','off')
end

if handles.enable.buttons(5)==1
    set(handles.pushbutton7,'enable','on')
else
    set(handles.pushbutton7,'enable','off')
end

if handles.enable.buttons(6)==1
    set(handles.pushbutton8,'enable','on')
else
    set(handles.pushbutton8,'enable','off')
end

if handles.enable.buttons(7)==1
    set(handles.pushbutton9,'enable','on')
else
    set(handles.pushbutton9,'enable','off')
end

if handles.enable.buttons(8)==1
    set(handles.pushbutton10,'enable','on')
else
    set(handles.pushbutton10,'enable','off')
end

if handles.enable.buttons(9)==1
    set(handles.pushbutton11,'enable','on')
else
    set(handles.pushbutton11,'enable','off')
end

if handles.enable.buttons(10)==1
    set(handles.pushbutton12,'enable','on')
else
    set(handles.pushbutton12,'enable','off')
end

if handles.enable.buttons(11)==1
    set(handles.pushbutton13,'enable','on')
else
    set(handles.pushbutton13,'enable','off')
end

if handles.enable.buttons(12)==1
```

```
    set(handles.pushbutton14,'enable','on')
else
    set(handles.pushbutton14,'enable','off')
end

if handles.enable.buttons(13)==1
    set(handles.pushbutton15,'enable','on')
else
    set(handles.pushbutton15,'enable','off')
end

if handles.enable.buttons(14)==1
    set(handles.pushbutton16,'enable','on')
else
    set(handles.pushbutton16,'enable','off')
end

if handles.enable.buttons(15)==1
    set(handles.pushbutton17,'enable','on')
else
    set(handles.pushbutton17,'enable','off')
end

if handles.enable.buttons(16)==1
    set(handles.pushbutton18,'enable','on')
else
    set(handles.pushbutton18,'enable','off')
end

if handles.enable.buttons(17)==1
    set(handles.pushbutton19,'enable','on')
else
    set(handles.pushbutton19,'enable','off')
end

if handles.enable.buttons(18)==1
    set(handles.pushbutton20,'enable','on')
else
    set(handles.pushbutton20,'enable','off')
end


%% Update the main boolean arrays according to the selected case

if strcmp(handles.trial,'Con48')==1
    handles.switchbool.Con48=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Con26')==1
    handles.switchbool.Con26=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Con28')==1
```

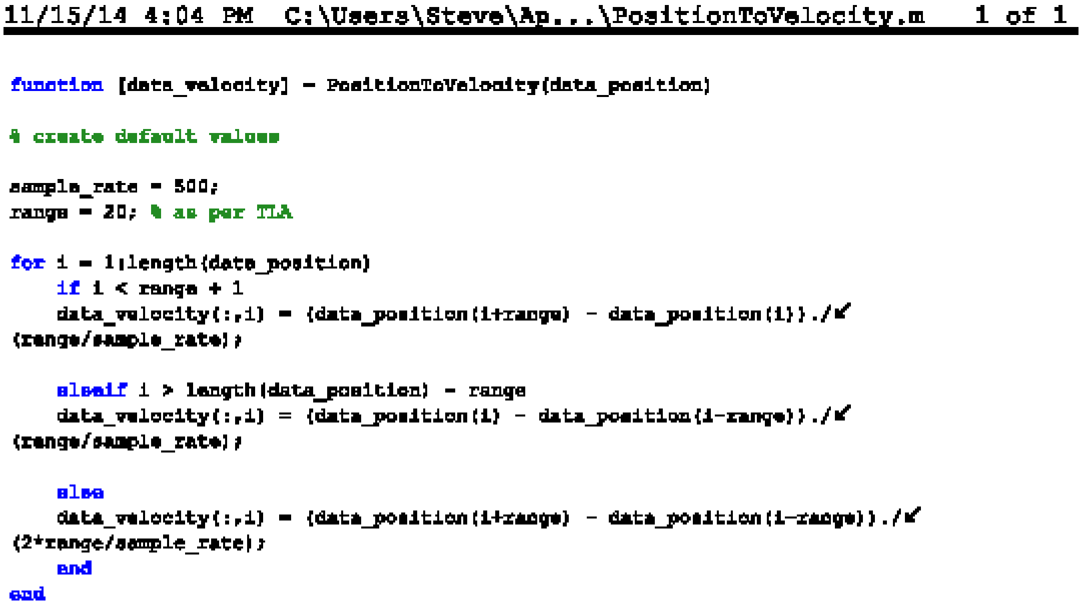
```
    handles.switchbool.Con28=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Div84')==1
    handles.switchbool.Div84=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Div62')==1
    handles.switchbool.Div62=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Div82')==1
    handles.switchbool.Div82=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Con812')==1
    handles.switchbool.Con812=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Con610')==1
    handles.switchbool.Con610=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Con612')==1
    handles.switchbool.Con612=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Div128')==1
    handles.switchbool.Div128=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Div106')==1
    handles.switchbool.Div106=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'Div126')==1
    handles.switchbool.Div126=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'M2R5')==1
    handles.switchbool.M2R5=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'R2M5')==1
    handles.switchbool.R2M5=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'M2L5')==1
    handles.switchbool.M2R5=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'L2M5')==1
    handles.switchbool.L2M5=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'M2R10')==1
    handles.switchbool.M2R10=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'R2M10')==1
    handles.switchbool.R2M10=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'M2L10')==1
    handles.switchbool.M2L10=handles.switchbool.currentbool;

elseif strcmp(handles.trial,'L2M10')==1
```

```
    handles.switchbool.L2M10=handles.switchbool.currentbool;
end



handlesout=handles;
```



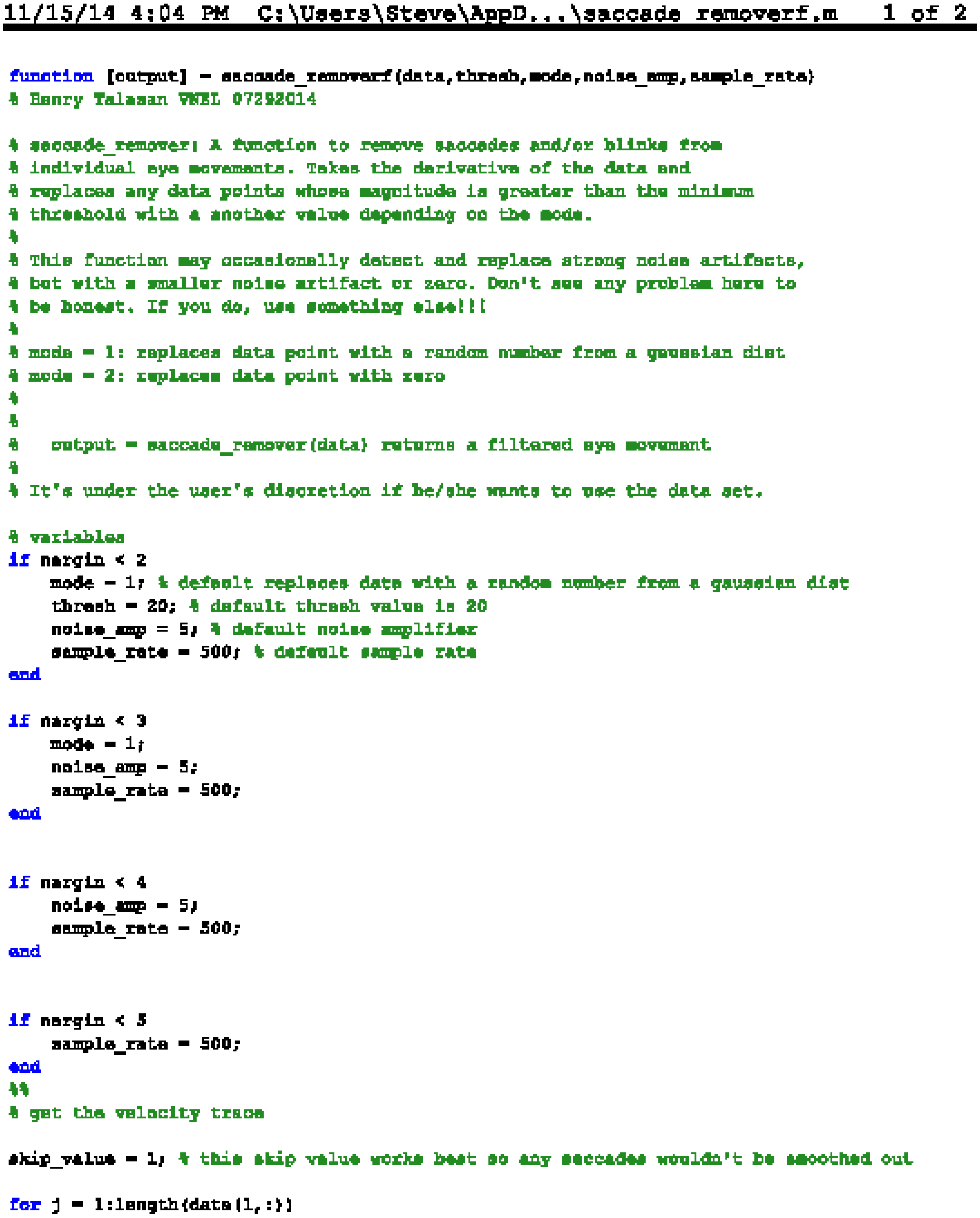
11/15/14 4:04 PM C:\Users\Steve\AppData\Local\T...\PkVel.m 1 of 1

```
function [Pk,PkLoc] = PkVel(data)

tmp = smooth(data(1:300));
% begmv = mean(data(1:25));
% endmv = mean(data(1000:1025));
% ten = (endmv-begmv)*.1;
% tencheck = tmp - begmv - ten;


Pk = max(abs(diff(tmp)));
PkLoc = find(Pk==abs(diff(tmp)));

if numel(PkLoc) > 1
    PkLoc = PkLoc(1);
end
Pk = Pk*500;
PkLoc = PkLoc/500;
end
```



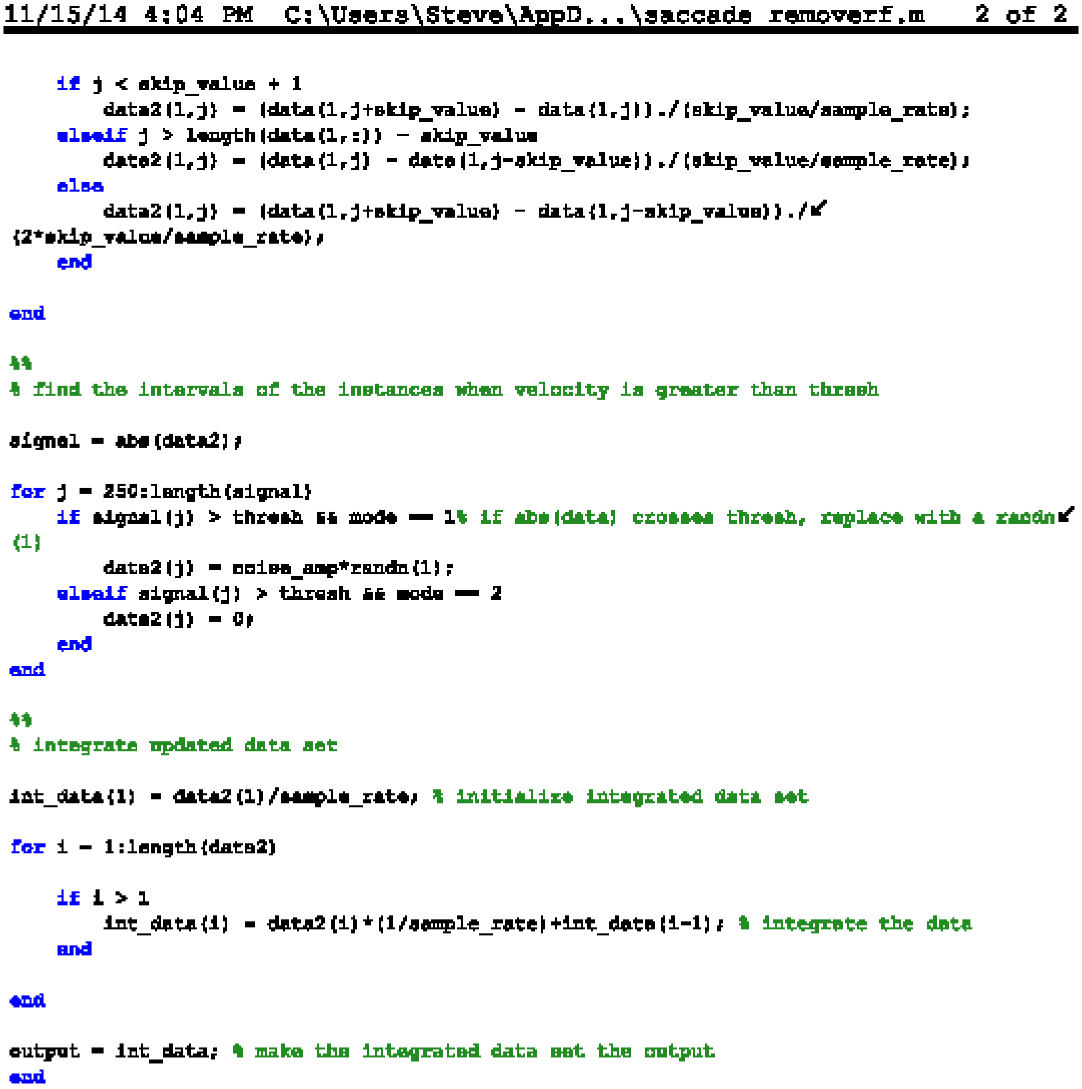
11/15/14 4:04 PM C:\Users\Steve\Ap...\PositionToVelocity.m 1 of 1

```
function [data_velocity] = PositionToVelocity(data_position)

% create default values

sample_rate = 500;
range = 20; % as per TLA

for i = 1:length(data_position)
    if i < range + 1
    data_velocity(:,i) = (data_position(i+range) - data_position(i))./↙
(range/sample_rate);

    elseif i > length(data_position) - range
    data_velocity(:,i) = (data_position(i) - data_position(i-range))./↙
(range/sample_rate);

    else
    data_velocity(:,i) = (data_position(i+range) - data_position(i-range))./↙
(2*range/sample_rate);
    end
end
```



11/15/14 4:04 PM C:\Users\Steve\AppD...\saccade removerf.m 1 of 2

```
function [output] = saccade_removerf(data,thresh,mode,noise_amp,sample_rate)
% Henry Talasan VNEL 07252014

% saccade_remover: A function to remove saccades and/or blinks from
% individual eye movements. Takes the derivative of the data and
% replaces any data points whose magnitude is greater than the minimum
% threshold with a another value depending on the mode.
%
% This function may occasionally detect and replace strong noise artifacts,
% but with a smaller noise artifact or zero. Don't see any problem here to
% be honest. If you do, use something else!!!
%
% mode = 1: replaces data point with a random number from a gaussian dist
% mode = 2: replaces data point with zero
%
%
%   output = saccade_remover(data) returns a filtered eye movement
%
% It's under the user's discretion if he/she wants to use the data set.

% variables
if nargin < 2
    mode = 1; % default replaces data with a random number from a gaussian dist
    thresh = 20; % default thresh value is 20
    noise_amp = 5; % default noise amplifier
    sample_rate = 500; % default sample rate
end

if nargin < 3
    mode = 1;
    noise_amp = 5;
    sample_rate = 500;
end


if nargin < 4
    noise_amp = 5;
    sample_rate = 500;
end


if nargin < 5
    sample_rate = 500;
end
%%
% get the velocity trace

skip_value = 1; % this skip value works best so any saccades wouldn't be smoothed out

for j = 1:length(data(1,:))
```



11/15/14 4:04 PM   C:\Users\Steve\AppD...\saccade removerf.m   2 of 2

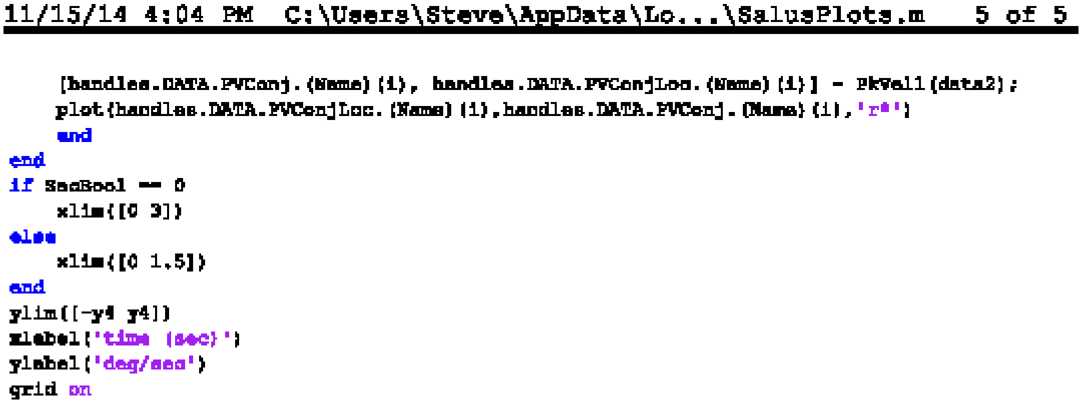
```
    if j < skip_value + 1
        data2(1,j) = (data(1,j+skip_value) - data(1,j))./(skip_value/sample_rate);
    elseif j > length(data(1,:)) - skip_value
        data2(1,j) = (data(1,j) - data(1,j-skip_value))./(skip_value/sample_rate);
    else
        data2(1,j) = (data(1,j+skip_value) - data(1,j-skip_value))./↙
(2*skip_value/sample_rate);
    end

end

%%
% find the intervals of the instances when velocity is greater than thresh

signal = abs(data2);

for j = 250:length(signal)
    if signal(j) > thresh && mode == 1% if abs(data) crosses thresh, replace with a randn↙
(1)
        data2(j) = noise_amp*randn(1);
    elseif signal(j) > thresh && mode == 2
        data2(j) = 0;
    end
end

%%
% integrate updated data set

int_data(1) = data2(1)/sample_rate; % initialize integrated data set

for i = 1:length(data2)

    if i > 1
        int_data(i) = data2(i)*(1/sample_rate)+int_data(i-1); % integrate the data
    end

end

output = int_data; % make the integrated data set the output
end
```

```
function [handles] = SalusPlots(data,fignum,movement,y1,y2,y3,y4,SacBool,handles)
if length(movement) == 13 % naming the movement in the function
    Name = movement([1:3 6:7 12:13]);
else
    Name = [movement(1) '2' movement([10 16:17])];
end
figure(fignum)
%% Left Eye
subplot(4,2,1)
AvgDat = AvgData(data); % getting the average data
for i = 1:length(data)
    t = 0.002:0.002:3.000;
    plot(t,data{i}(2,:),'.','MarkerSize',.3,'Color',[.5 .5 .5]); % plots each trial
    hold on
    plot(t,AvgDat(2,:),'LineWidth',2,'Color',[1 0 0]); % plotting the mean
end
title(sprintf('%s Degrees (LE)',movement),'FontSize',14)
if SacBool == 0
    xlim([0 3])
else
    xlim([0 1.5])
end
ylim(y1) % hard coded y bounds
xlabel('time (sec)')
ylabel('degrees')
grid on

%% Left Eye Velocity
subplot(4,2,3)
for i = 1:length(data)
    t = 0.002:0.002:3.000;
    plot(t,PositionToVelocity(data{i}(2,:)),'b.','MarkerSize',.3,'Color',[.5 .5 .5]);
    hold on
    plot(t,PositionToVelocity(AvgDat(2,:)),'b','LineWidth',2)
end
title('LE Velocity','FontSize',14)
if SacBool == 0
    xlim([0 3])
else
    xlim([0 1.5])
end
ylim([-y4 y4])
xlabel('time (sec)')
ylabel('deg/sec')
grid on

%% Right Eye
subplot(4,2,2)
for i = 1:length(data)
```

```
    t = 0.002:0.002:3.000;
    plot(t,data{i}(1,:),'b.','MarkerSize',.3,'Color',[.5 .5 .5]);
    hold on
    plot(t,AvgDat(1,:),'LineWidth',2,'Color',[1 0 0]);

end
title(sprintf('%s Degrees (RE)',movement),'FontSize',14)
if SacBool == 0
    xlim([0 3])
else
    xlim([0 1.5])
end
ylim(y1)
xlabel('time (sec)')
ylabel('degrees')
grid on

%% Right Eye Velocity
subplot(4,2,4)
for i = 1:length(data)
    t = 0.002:0.002:3.000;
    plot(t,PositionToVelocity(data{i}(1,:)),'b.','MarkerSize',.3,'Color',[.5 .5 .5]);
    hold on
    plot(t,PositionToVelocity(AvgDat(1,:)),'b','LineWidth',2)
end
title('RE Velocity','FontSize',14)
if SacBool == 0
    xlim([0 3])
else
    xlim([0 1.5])
end
ylim([-y4 y4])
xlabel('time (sec)')
ylabel('deg/sec')
grid on

%% Disconjugate
subplot(4,2,5)

for i = 1:length(data)
    t = 0.002:0.002:3.000;
    if SacBool == 0
    data2 = data{i}(1,:)+data{i}(2,:); % getting the total disconjugate movement par↙
trial
    handles.DATA.PosDisc.(Name)(i,:) = data2; % saving the movement in the mat file
    plot(t,data2,'b.','MarkerSize',.3,'Color',[.5 .5 .5]);
    hold on
    plot(t,(AvgDat(1,:)+AvgDat(2,:)),'LineWidth',2,'Color',[1 0 0]);
    else
    data2 = data{i}(1,:)-data{i}(2,:); % getting the total disconjugate movement par↙
trial
```

```
    handles.DATA.PosDisc.(Name)(i,:) = data2; % saving the movement in the mat file
    plot(t,data2,'b.','MarkerSize',.3,'Color',[.5 .5 .5]);
    hold on
    plot(t,AvgDat(1,:)-AvgDat(2,:),'LineWidth',2,'Color',[1 0 0]);
    end
end
title(sprintf('%s Degrees (Disconjugate)',movement),'FontSize',13)
if SacBool == 0
    xlim([0 3])
else
    xlim([0 1.5])
end
ylim(y2)
ylabel('degrees')
grid on


%% Disconjugate Velocity
subplot(4,2,7)
for i = 1:length(data)
    t = 0.002:0.002:3.000;
    if SacBool == 0
    data2 = PositionToVelocity(data{i}(1,:)+data{i}(2,:)); % getting the disconjugate↙
movement's velocity
    handles.DATA.VelDisc.(Name)(i,:) = data2; % saving data
    plot(t,data2,'b.','MarkerSize',.3,'Color',[.5 .5 .5]);
    hold on
    plot(t,PositionToVelocity((AvgDat(2,:)+AvgDat(1,:))),'b','LineWidth',2)
    [handles.DATA.PVDisc.(Name)(i), handles.DATA.PVDiscLoc.(Name)(i)] = PkVel1(data2); %↙
saving the Peak Information
    plot(handles.DATA.PVDiscLoc.(Name)(i),handles.DATA.PVDisc.(Name)(i),'r*')
    else
    data2 = PositionToVelocity(data{i}(1,:)-data{i}(2,:)); % getting the disconjugate↙
movement's velocity
    handles.DATA.VelDisc.(Name)(i,:) = data2; % saving data
    plot(t,data2,'b.','MarkerSize',.3,'Color',[.5 .5 .5]);
    hold on
    plot(t,PositionToVelocity(AvgDat(1,:)-AvgDat(2,:)),'b','LineWidth',2)
    [handles.DATA.PVDisc.(Name)(i), handles.DATA.PVDiscLoc.(Name)(i)] = PkVel1(data2); %↙
saving the Peak Information
    plot(handles.DATA.PVDiscLoc.(Name)(i),handles.DATA.PVDisc.(Name)(i),'r*')
    end
end
if SacBool == 0
    xlim([0 3])
else
    xlim([0 1.5])
end
ylim([-y4 y4])
xlabel('time (sec)')
ylabel('deg/sec')
```

```
grid on

%% Conjugate
subplot(4,2,6)
for i = 1:length(data)
    t = 0.002:0.002:3.000;
    if SacBool == 0
    data2 = data{i}(1,:)-data{i}(2,:); % get conjugate movement per trial
    handles.DATA.PosConj.(Name)(i,:) = data2; % saving conjugate data
    plot(t,data2,'b.','MarkerSize',.3,'Color',[.5 .5 .5]);
    hold on
    plot(t,(AvgDat(1,:)-AvgDat(2,:)),'LineWidth',2,'Color',[1 0 0]);
    else
    data2 = (data{i}(1,:)+data{i}(2,:))/2; % get conjugate movement per trial
    handles.DATA.PosConj.(Name)(i,:) = data2; % saving conjugate data
    plot(t,data2,'b.','MarkerSize',.3,'Color',[.5 .5 .5]);
    hold on
    plot(t,(AvgDat(1,:)+AvgDat(2,:))/2,'LineWidth',2,'Color',[1 0 0]);
    end
end
title(sprintf('%s Degrees (Conjugate)',movement),'FontSize',13)
if SacBool == 0
    xlim([0 3])
else
    xlim([0 1.5])
end
ylim(y3)
xlabel('time (sec)')
ylabel('degrees')
grid on

%% Conjugate Velocity
subplot(4,2,8)
for i = 1:length(data)
    t = 0.002:0.002:3.000;
    if SacBool == 0
    data2 = PositionToVelocity((data{i}(1,:)-data{i}(2,:))); % making the conj velocity↙
per trial
    handles.DATA.VelConj.(Name)(i,:) = data2; % save conjugate velocity
    plot(t,data2,'b.','MarkerSize',.3,'Color',[.5 .5 .5]);
    hold on
    plot(t,PositionToVelocity((AvgDat(2,:)-AvgDat(1,:))),'b','LineWidth',2)
    [handles.DATA.PVConj.(Name)(i), handles.DATA.PVConjLoc.(Name)(i)] = PkVel1(data2);
    plot(handles.DATA.PVConjLoc.(Name)(i),handles.DATA.PVConj.(Name)(i),'r*')
    else
    data2 = PositionToVelocity((data{i}(1,:)+data{i}(2,:))/2); % making the conj velocity↙
per trial
    handles.DATA.VelConj.(Name)(i,:) = data2; % save conjugate velocity
    plot(t,data2,'b.','MarkerSize',.3,'Color',[.5 .5 .5]);
    hold on
    plot(t,PositionToVelocity((AvgDat(2,:)+AvgDat(1,:))/2),'b','LineWidth',2)
```



11/15/14 4:04 PM C:\Users\Steve\AppData\Lo...\SalusPlots.m   5 of 5

```
    [handles.DATA.PVConj.(Name)(i), handles.DATA.PVConjLoc.(Name)(i)] = PkVal1(data2);
    plot(handles.DATA.PVConjLoc.(Name)(i),handles.DATA.PVConj.(Name)(i),'r*')
    end
end
if SecBool == 0
    xlim([0 3])
else
    xlim([0 1.5])
end
ylim([-y4 y4])
xlabel('time (sec)')
ylabel('deg/sec')
grid on
```

C.1 Haploscope Table Components

In Part C of the appendix, functional dimensions of Haploscope table components are documented.

ings

1. Haploscope Physical Dimensions

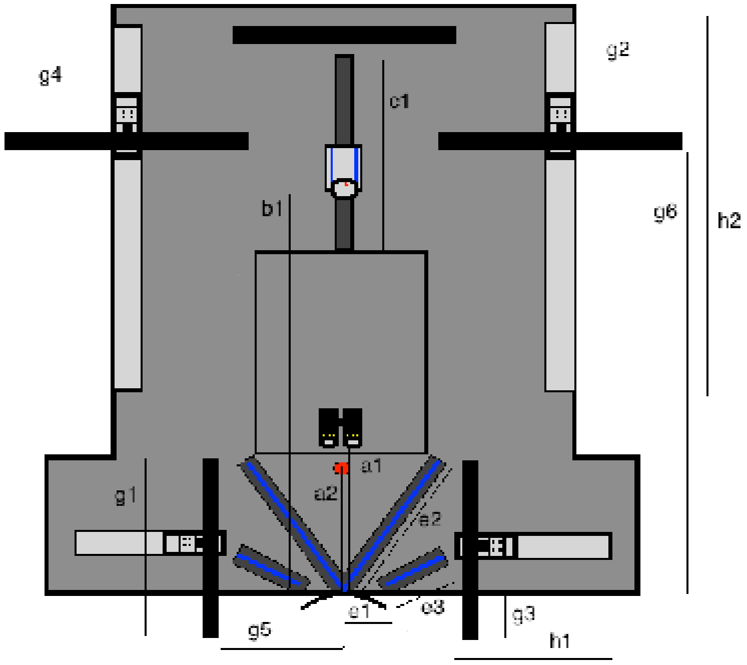


Figure 1: Major measurements of Haploscope

a1. Distance Vergence Sensor – 35 cm ± 1 cm
a2. Distance to Infrared Source – 30 cm ± 2 cm
b1. Distance to Simple Accommodation Sensor – 60 cm ± 5 cm
c1. Length of Track System for Accommodation Sensor – 40 cm ± 5 cm
e1. Distance between Mirror Holder's Instrument Table attachment Screws – 12 cm ± 0.2 cm
e2. Length of Large Reflection Mirrors – 30 cm ± 1 cm
e3. Length of Small Reflection Mirrors – 20 cm ± 5 cm
g1. Length of Close Stimuli Monitor – 51 cm ± 5 cm
g2. Length of Distant Stimuli Monitor – 56 cm ± 5 cm
g3. Length of Close Stimuli Monitor outside instrument table edge – 17 cm ± 2 cm
g4. Length of Distant Stimuli Monitor outside instrument table edge – 23 cm ± 2 cm
g5. Standard Position of Close Stimuli Monitor from tip of Large Mirror Holder – 30 cm ± 1 cm
g6. Standard Position of Distant Stimuli Monitor from table edge – 120 cm ± 1 cm
h1. Length of Close Track Rail - 40 cm ± 5 cm
h2. Length of Distant Track Rail - 120 cm ± 5 cm

2. Haploscope Components

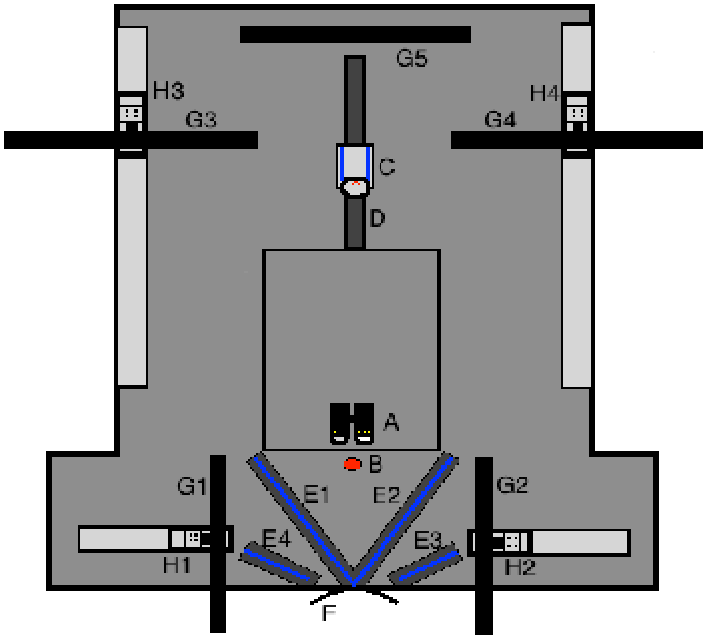


Figure 2: Major Parts of Haploscope

A. Vergence Sensor - ISCAN Eye Tracker Model ETL-400
B. Infrared Source – ISCAN Infrared Source
C. Accommodation Sensor – Power Refractor PlusOptix III
D. Simple Track System for Accommodation Sensor
E1. Large Reflection Mirror in Large Mirror Holder 1 - Stereoscopic Mirror 50/50
E2. Large Reflection Mirror in Large Mirror Holder 2 - Stereoscopic Mirror 50/50
E3. Small Reflection Mirror in Small Mirror Holder 1- Stereoscopic Mirror 50/50
E4. Small Reflection Mirror in Small Mirror Holder 2 - Stereoscopic Mirror 50/50
G1. Close Stimuli Monitor 1 - Dell No. 1703 FPS
G2. Close Stimuli Monitor 2 - Dell No. 1703 FPS
G3. Distant Stimuli Monitor 1 - Dell No. 1703 FPS
G4. Distant Stimuli Monitor 2 - Dell No. 1703 FPS
G5. Calibration Screen - Dell No. 1703 FPS
H1. Close Track System including Transit Plate Carrier 1
H2. Close Track System including Transit Plate Carrier 2
H3. Distant Track System including Transit Plate Carrier 1
H4. Distant Track System including Transit Plate Carrier 2
F. Chin Rest

1. Monitor Track Dimensions & Components (From Top)

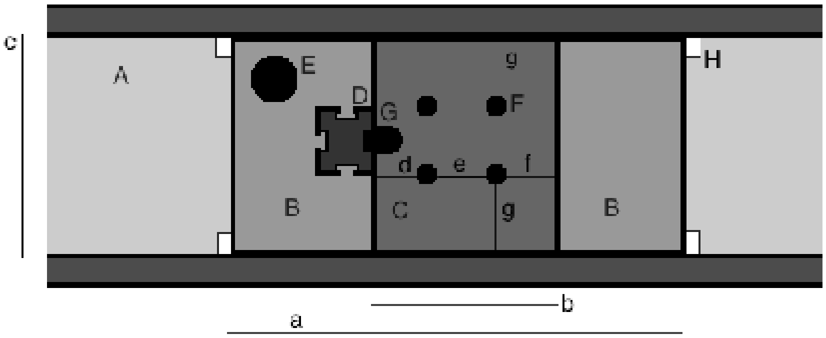
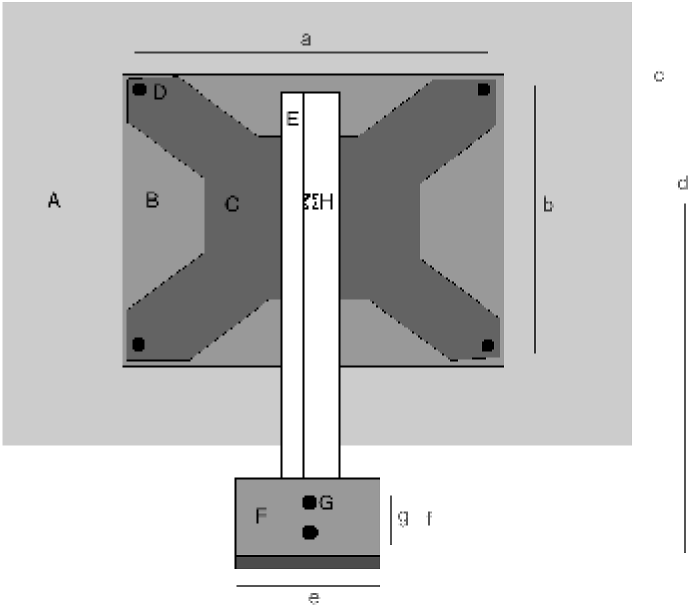


Figure 3: Major Parts of Monitor Track- Z axis view

Track Rail– ROTRONIC, ROLINE Wall Track System for LCD, Article Number 17.03.1151
B. Transit Plate Carrier – Included in ROTRONIC, ROLINE, Article Number 17.03.1151
C. Attachment Plate – Alumni L-shaped bar with square bottom
D. 80/20 Bar - 10 Series 4111
E. Attachment Securing Hub – Secures Transit Plate Carrier to Track Rail, included in ROTRONIC, ROLINE, Article Number 17.03.1151
F. Four Security Screws attaching Attachment Plate to Transit Plate Carrier
G. Two Security Screws attaching Attachment Plate to 80/20 Bar
H. Plastic Slots - Included in ROTRONIC, ROLINE, Article Number 17.03.1151
a. Length of Transit Plate Carrier – 19 cm
b. Length of Attachment Plate – 8cm
c. Width of Transit Plate Carrier and Attachment Plate – 8 cm
d. Distance - 2.5 cm
e. Distance - 3 cm
f. Distance - 2.5 cm
g. Distance – 3 cm

2. Monitor Track Dimensions & Components (From Back)



3.

Figure 4: Major Parts of Monitor Track- Y View

Stimuli Monitor
B. Monitor attachment area
C. Monitor Attachment Bracket
D. Four Monitor Bracket Attachment Screws
E. 80/20 Bar - 10 Series 4111 with
F. Attachment Plate – Alumni L-shaped bar with square bottom
G. Two Security Screws attaching Attachment Plate to 80/20 Bar
H. 80/20 Bar to Monitor Bracket Attachment Screw
a. Length between Monitor Attachment Screws – 10 cm
b. Width between Monitor Attachment Screws – 10 cm
c. Height of 80/20 Bar – 35 cm
d. Height of 80/20 Bar to Monitor Bracket Attachment Screw – 20 cm
e. Length of Attachment Plate – 8 cm
f. Height of Attachment Plate – 5 cm
g. Distance between Attachment Plate to 80/20 Bar Screws – 1.5 cm

4. Primary Mirror Dimensions

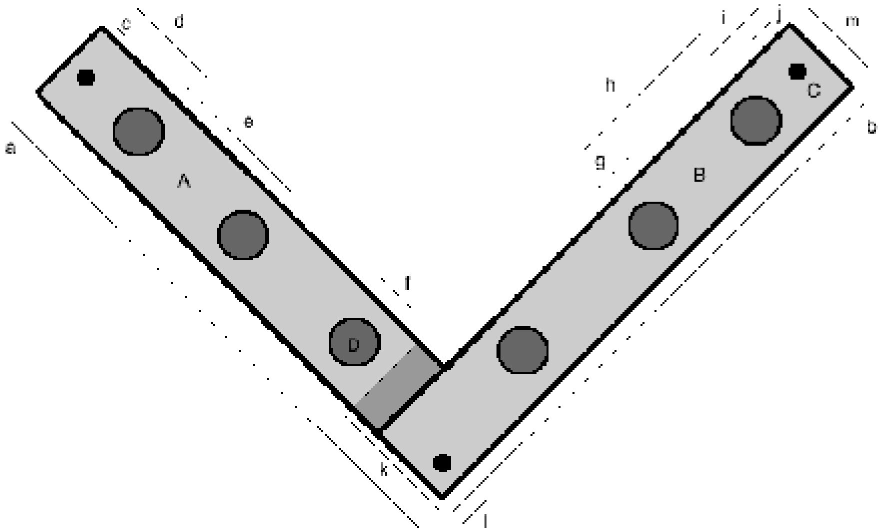


Figure 5: Primary Mirror Holder Major Dimensions

A. Large Mirror Holder 1
B. Large Mirror Holder 2
C. Three Mirror Holder to Instrument Table Attachment Screws
D. Six Holes for Mirror Attachment
a. Length of Large Mirror Holder 1 – 30 cm
b. Length of Large Mirror Holder 2 – 30 cm
c. Distance - 1 cm
d. Distance - 5 cm
e. Distance - 7.5 cm
f. Hole Radius – 2.5 cm
g. Hole Radius – 2.5 cm
h. Distance between Holes - 5 cm
i. Distance - 5 cm
j. Distance - 1 cm
k. Distance - 10 cm
l. Distance - 1 cm
m. Width of Mirror Holder 1 and 2 – 4.5 cm

5. Secondary Mirror Dimensions

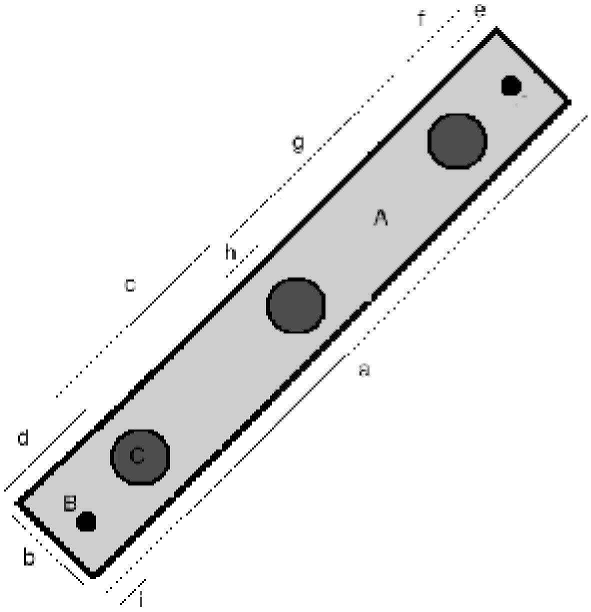


Figure 6: Major Dimensions of Secondary Mirror Holders

A. Small Mirror Holder 1 and 2
B. Two Mirror Holder to Instrument Table Attachment Screws
C. Three Holes for Mirror Attachment
a. Length of Small Mirror Holder 1 and 2 – 20 cm
b. Width of Small Mirror Holder 1 and 2 – 4.5 cm
c. Distance - 5 cm
d. Distance - 5 cm
e. Distance - 1 cm
f. Distance - 5 cm
g. Distance - 5 cm
h. Distance between Holes for Mirror - 5 cm
i. Distance - 1 cm

REFERENCES

[1] Marieb, Elaine Nicpon, and Katja Hoehn. Human Anatomy & Physiology. San Francisco: Benjamin Cummings, 2010. Print.

[2] Purves, D. A., *et al.*, *Neuroscience*, 4 ed. Sunderland, MA: Sinauer Associated, Inc. 1993.

[3] “Diagram of the retina,” retrieved April 11, 2014 from http://www.catalase.com/retina.gif

[4] Sevel, D., “The origins and insertions of the extraocular muscles: development, histologic features, and clinical significance,” *Trans Am Ophthalmol Soc,* vol.24, pp. 488-526, 1869.

[5] Collewijn, H.and Erkelens, C. J., “Binocular eye movements and the perception of depth,” *Rev Oculomot Res*, vol. 4, pp. 13-9, 1983.

[6] Alvarez, T. L., et al., “Short-term predictive changes in the dynamics of disparity vergence eye movements,” J Vis, vol. 5, pp. 640-9, 2005.

[7] Scheimn, Mitchell, Jane Gwiazda, and Tianjing Li. "Non-surgical Interventions for Convergence Insufficiency." Cochrane Eyes and Vision Group (2011): n. pag. PubMed. Web. 15 Oct. 2014.

[8] Carvelho, Tristan, Robert S. Allison, Elizabeth L. Iving, and Christopher Herriot. "Computer Gaming for Vision Therapy." *IEEE* (2008): 198-204. Print.

[9] Foss, Alexander. "I-BiT - Evaluation of a Novel Binocular Treatment System (I-BiTTM) in Children With Amblyopia." *ClinicalTrials.gov*. NIH, Oct. 2012. Web. 18 Dec. 2013. <ID:NCT01702727>.

[10] "ISCANHomeFrame." *ISCANHomeFrame*. N.p., n.d. Web. 12 Oct. 2013. <http://www.iscaninc.com/>.

[11] "Solutions." *SensoMotoric Instruments GmbH Gaze and Eye Tracking Systems Products Overview*. N.p., n.d. Web. 12 Oct. 2013. <http://www.smivision.com/en/gaze-and-eye-tracking-systems/products/overview.html>.

[12] "Announcing the EyeLink 1000 Plus - an Innovative Successor to the EyeLink 1000..." *SR Research*. N.p., n.d. Web. 12 Oct. 2013. <http://www.sr-research.com/>.

[13] "Eye Tracking Systems for Research and Analysis." *Tobii*. N.p., n.d. Web. 12 Oct. 2013. <http://www.tobii.com/en/eye-tracking-research/global/>.

[14] "Cambridge Research Systems - High-Speed Video Eye Tracker Toolbox." *Cambridge Research Systems - High-Speed Video Eye Tracker Toolbox*. N.p., n.d. Web. 12 Oct. 2013. <http://www.crsltd.com/tools-for-vision-science/eye-tracking/high-speed-video-eye-tracker-toolbox/>.

[15] Razdan, Rikki. "ISCAN IR Level Certification." Letter. 15 July 2009.

[16] Blade, Pamela J., and Rowan Candy. "Validation of the PowerRefractor for Measuring Human Infant... : Optometry & Vision Science." N.p., June 2006. Web. 28 Sept. 2014.

[17] Porcar E, Martinez-Palomera A. Prevalence of general binocular dysfunctions in a population of university students. Optom Vis Sci 1997;74:111-113.

[18] Rouse MW, Borsting E, Hyman L, Scheiman M l. Frequency of convergence insufficiency among fifth and sixth graders. The Convergence Insufficiency and Reading Study (CIRS) group. Optom Vis Sci 1999;76:643-649.

[19] Rouse MW, Hyman L, Hussein M, Solan H. Frequency of convergence insufficiency in optometry clinic settings. Convergence Insufficiency and Reading Study (CIRS) Group. Optom Vis Sci 1998;75:88-96.

[20] Rouse M, Borsting E, Mitchell GL, Scheiman M. Validity of the convergence insufficiency symptom survey: a confirmatory study. Optom Vis Sci 2009;86:357-363.

[21] Cohen Y, Segal O, Barkana Y, et al. Correlation between asthenopic symptoms and different measurements of convergence and reading comprehension and saccadic fixation eye movements. Optometry 2010;81:28-34.