BOSTON UNIVERSITY

COLLEGE OF ENGINEERING

Dissertation

TOOLS FOR INTERFACING, EXTRACTING, AND ANALYSING NEURAL SIGNALS USING WIDE-FIELD FLUORESCENCE IMAGING AND OPTOGENETICS IN AWAKE BEHAVING MICE

by

MARK E. BUCKLIN

B.S., Columbia University, 2009
M.S., Boston University, 2016

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

2019

Approved by

First Reader

Xue Han, Ph.D.
Associate Professor of Biomedical Engineering

Second Reader

David Boas, Ph.D.
Professor of Biomedical Engineering

Third Reader

Ian Davison, Ph.D.
Assistant Professor of Biology

Fourth Reader

Jerome C. Mertz, Ph.D.
Professor of Department of Biomedical Engineering

Fifth Reader

Kamal Sen, Ph.D.
Associate Professor of Biomedical Engineering

for pickle

Acknowledgements

The support and patience I have received from my committee has gone far beyond what should be expected of anyone. I can't thank you enough.

- Xue Han, Ph.D.
- David Boas, Ph.D.
- Kamal Sen Ph.D.
- Jerome Mertz, Ph.D.
- Ian Davison, Ph.D.
- Vickery Trinkaus-Randall, Ph.D.
- Steven Borkan, M.D.

TOOLS FOR INTERFACING, EXTRACTING, AND ANALYSING NEURAL SIGNALS USING WIDE-FIELD FLUORESCENCE IMAGING AND OPTOGENETICS IN AWAKE BEHAVING MICE

MARK E. BUCKLIN

Boston University, College of Engineering, 2019

Major Professor: Xue Han, Ph.D.
Associate Professor of Biomedical Engineering

ABSTRACT

Imaging of multiple cells has rapidly multiplied the rate of data acquisition as well as our knowledge of the complex dynamics within the mammalian brain. The process of data acquisition has been dramatically enhanced with highly affordable, sensitive image sensors enable high-throughput detection of neural activity in intact animals. Genetically encoded calcium sensors deliver a substantial boost in signal strength and in combination with equally critical advances in the size, speed, and sensitivity of image sensors available in scientific cameras enables high-throughput detection of neural activity in behaving animals using traditional wide-field fluorescence microscopy. However, the tremendous increase in data flow presents challenges to processing, analysis, and storage of captured video, and prompts a reexamination of traditional routines used to process data in neuroscience and now demand improvements in both our hardware and software applications for processing, analyzing, and storing captured video. This project demonstrates the ease with which a dependable and affordable wide-field fluorescence imaging system can be assembled and

integrated with behavior control and monitoring system such as found in a typical neuroscience laboratory.

An Open-source MATLAB toolbox is employed to efficiently analyze and visualize large imaging data sets in a manner that is both interactive and fully automated. This software package provides a library of image pre-processing routines optimized for batch-processing of continuous functional fluorescence video, and additionally automates a fast unsupervised ROI detection and signal extraction routine. Further, an extension of this toolbox that uses GPU programming to process streaming video, enabling the identification, segmentation and extraction of neural activity signals on-line is described in which specific algorithms improve signal specificity and image quality at the singe cell level in a behaving animal. This project describes the strategic ingredients for transforming a large bulk flow of raw continuous video into proportionally informative images and knowledge.

Preface

I have structured this document to roughly coincide with a chronological account of 6 years spent in a neuro-oriented biomedical engineering lab. My role in the lab was centered around exploratory device design and development, mostly targeting application in neuroscience research, with intended users being neuroscientist colleagues. One of the lab's most remarkable assets is the breadth and diversity of its constituents in terms of their skills and experience, both within and between the engineering/development and the science/medical sides of the lab. All efforts stood to benefit from the close proximity to skilled colleagues, most notably for the complementary guide and provide roles that assisted the development process of new devices and the experiments they were intended for.

My initial experience in optoelectronic device development was as an undergrad at Columbia University where I was advised by Elizabeth Hillman, and developed a device that combined thermography and near-infrared spectroscopy in a portable and inexpensive device intended to provide early detection of adverse neoplastic changes through at-home daily monitoring, particularly targeting use by patients with high-risk for breast cancer. I then went to the Das Lab where I developed macroscopic imaging systems used for intrinsic imaging in the visual cortex of awake primates. As a MD/PhD student, I attempt to maintain a potential to adapt the end-products of each development for clinical applicability. The story presented here is rather unusual in that success precedes failure. The volume of tangible presentable results is greatest toward the beginning stages of the work described here. This unusual inversion is what make this story worth hearing, however. Thank you for taking the

time to read this. I hope that at least the technical information provided herein, if not the procedural insight, is valuable in your current or future endeavors.

Contents

List of Tables

List of Figures

List of source codes

List of Abbreviations

CAD	.....	Computer-Aided Design
DOG	.....	Difference Of Gaussian (distributions)
FWHM	.....	Full-Width at Half Maximum
LGN	.....	Lateral Geniculate Nucleus
PDF	.....	Probability Distribution Function
$\mathbb{R}^2$	.....	the Real plane

Chapter 1

Introduction: Background and Literature Review

1.1 Optical Imaging of Neural Activity

Optical techniques for observing neural activity have recently advanced owing to both an evolution of digital imaging technology, and the development of engineered proteins that act as fluorescent indicators of neural activity. Image sensors, like those found in scientific-CMOS (sCMOS) cameras are larger, faster, and more sensitive than prior scientific grade cameras. Meanwhile, the latest generation of Genetically Encoded Calcium Indicators (GECIs), collectively called GCaMP6, report fluctuations in neural activation with extremely high fidelity. This combination of developments enables neuroscientists to open a wider channel to the brain than previously possible using conventional epifluorescence microscopy techniques that enable simultaneous recording from hundreds to thousands of neurons. Expanding the fraction of the observable neurons in an interconnected network could improve understanding of neural coding and provide insight into mechanistic properties of neural disease. Additionally, feeding a large set of neural response information to a machine learning algorithm in a neuroprosthetic application may provide improved predictive performance even when the exact mechanism of prediction is difficult to discern. However, several major challenges currently antagonize the potential benefits of these new technologies:

1. The increased size of raw data from a single imaging session can easily overwhelm the computational resources typically used to process similar but smaller sets of data.
2. The accumulation of raw data on disk over multiple imaging sessions quickly exceeds the data-storage capacity of most lab-scale servers, forcing researchers to halt data collection to process and delete, potentially creating a "nightmare scenario".
3. The experimental design and data analysis procedures familiar to neuroscientists for network activity data for 5 to 10 cells produce highly biased spurious results in the absence of numerous stimulus-response repetitions, i.e., trials. The number of repeated trials sufficient to produce an accurate description of the neural response to any stimulus is on the order of 2N, where N is the number of neurons being measured.

In the chapters that follow I provide background on the general procedure for offline video processing. I also discuss some of the issues that limit execution of these procedures on a large dataset, and the variety of approaches that I and others have attempted to address this issue. I then introduce the streaming approach that is capable of directly processing video during acquisition and extracting signals, thereby saving relevant signals only while also discarding or compressing the raw video. This approach relies on GPU programming and therefore I also provide background on the application of graphics cards for computationally demanding tasks. Using a graphics card for programming in the MATLAB environment is also discussed.

Capturing wide-field fluorescence images at high spatial and temporal resolution enables us to measure functional dynamic changes in multiple cells within a large interconnected network. Extracting a measure for each cell in a way that preserves spatial and temporal continuity with uniform/unbiased sampling of the observed sig-

nal is achievable but several factors complicate procedures intended to accomplish this task. One class of computer-vision procedure commonly applied to this task is image-segmentation (cell-segmentation in histology applications), a procedure that attempts to represent distinct objects in an image by association of each image pixel with one of any number of abstract objects or with the background. A variety of algorithms exist for efficiently performing this operation on single images. Most methods can be extended to operate in a 3rd dimension, applied to stacks of image frames to enable tracking cells at multiple depths, or equivalently over time.

However, motion induced by physiologic changes and animal movement necessitates the correct alignment of all frames in the sequence. Moreover, the massive fluctuations in signal intensity from individual and spatially overlapping cells often breeds unstable solutions for alignment that radically complicate cell identification routines by disrupting temporal continuity. Implementing a reliable procedure for identifying and tracking the same cells in each frame throughout the sequence thus becomes non-trivial.

1.2 Procedures for Calcium Imaging

The general goal of processing image data from functional fluorescence imaging experiments is to restructure raw image data in a way that maps pixels in each image frame to distinct individual cells or subcellular components, called 'Regions-Of-Interest' (ROI). Pixel-intensity values from mapped pixels are often reduced by combination to single dimensional 'trace' time-series. These traces indicate the fluorescence intensity of an individual neuron over time, and the collection approximates the distinct activity of all individual neurons in the microscope's field of view. However, this task is made difficult by motion of the brain throughout the experiment and by the apparent overlap of cells in the single image plane due to limitations of the camera's

2-dimensional perspective. These issues can be partially mitigated with a few image pre-processing steps. Most importantly is the alignment of images to correct for motion. These options are described in the Methods & Approaches section below. Most software packages specifically geared toward functional imaging implement either of two basic classes of pixel->cell mapping algorithms. One approach is to use image-segmentation routines for computer vision that seeks to combine adjacent pixels into distinct spatially segregated regions representing objects in the image.

The other common approach is to perform an eigenvalue decomposition on the covariance matrix from a stack of image frames (also called spectral decomposition, or Principal Component Analysis, PCA), resulting in an assembly of basis vectors that define the weighting coefficients for each pixel. Multiplying the basis-vectors (i.e., "components") with all frames produces a one-dimensional trace for each component. The linear combination is similar to the weighted image-segmentation method in that it assigns fractional coefficients to pixels. However, the procedure for computing the covariance matrix employed by PCA operates on as many pixels as exist in the image, multiplying each with every other pixel that creates a problem with np2 complexity, where p is the number of pixels in the image. I mention these issues inherent to PCA not because this project addresses them but because this project was initiated following substantial difficulty attempting to use PCA-based cell sorting methods with large datasets.

1.3 Computer Software Environments for Image Processing

The widespread usage of MATLAB in neuroscience communities lends potential for greater usability and easier adaptation to software developed in this environment. While software development environments focused on "ease-of-use" traditionally presume crippling sacrifices to computational performance, this assumption is now less

accurate.

Standard programs include ImageJ, the built-in routines in MATLAB's Image Processing Toolbox, Sci-Kits Image for Python, and a remarkable diversity of miscellaneous applications. MATLAB is a commercial software development platform that is geared toward fast production and the prototyping of data processing routines in a high-level programming language. It implements several core libraries (LINPACK, BLAS, etc.) that make multi-threaded operations on matrix type data highly efficient. While MATLAB has traditionally been considered the standard across neuroscience research labs, it is well recognized that its performance was lackluster for "vectorized" routines as compared to applications developed using lower-level languages like FORTRAN, C, and C++. Nevertheless, it remained in common use, and recent releases have added features that can drastically mitigate its poor performance issues, particularly through the development of a "Just-In-Time" compiler that automatically optimizes the deployment of computation accelerator resources for standard MATLAB functions. This feature enables code that performs repeated operations using for-loops or while-loops nearly as fast as equivalent code written in C. Additionally, code can be compiled into executable format using the Matlab Compiler toolbox, or used to generate equivalent C or C++ code using Matlab Coder.

1.4 Computational Resources for Processing Large Data Sets

Routines for extracting the activity in each cell from a collection of raw imaging data rely on simultaneous access to many pixels separated over space and time (and consequently, are separated on a disk). For long recording sessions however, the size of the collection of stored image data dramatically grows. This substantial increase in data size easily exceeds the capacity of system memory in the typical workstation computer available to most researchers. Thus, performing the necessary processing

performance enhancing routines using standard programs is often unfeasible.

Another popular approach to this challenge is the migration of processing routines to a cluster-based system. In this way, image data can be distributed across many interconnected computer nodes capable of performing all locally restricted image processing procedures in parallel and then passing data to other nodes in the cluster for tasks that rely on comparisons made across time. Access to clusters capable of performing in this way has been historically restricted to researchers in universities or other large organization, and the diversity of cluster types is sizeable, with clusters often having very particular configuration requirements for efficiently implementing data processing jobs. These issues pose difficulty to the use and shared development of software libraries for image processing routines, although the growth of "cloud computing" services such as Amazon's EC2 and the Google Compute Engine, as well as collaborative computing facilities such as the Massachusetts Green High-Performance Computing Center minimize several of these processing issues. Additionally, efforts to produce a standardized interface for accessing and distributing data and for managing computing resources across diverse computing environments have seen appreciable success. Apache's release of the open-source cluster computing framework, Hadoop, and a companion data-processing engine called Spark, have encouraged a massive growth in collaborative development projects, and consequently increased the availability of robust shared libraries for data processing in a variety of applications. The Spark API can be accessed using the open-source programming Python or other languages including Java, Scala, or R. The Thunder library, a Spark package released by the Freeman lab and developed in collaboration with a number of other groups at Janelia Farm and elsewhere is specifically geared for image processing of neural imaging data.

Many applications will find that the recent improvements in accessibility and

standardization make cluster computing an attractive and worthwhile option for processing large sets of reusable data. However, this strategy imposes harsh limitations for a neuroscientist engaged in a project that is continuously generating new data, as the time required to transfer entire imaging data sets across the internet may be prohibitive. Unfortunately, storage capacity on the cloud is also quite finite. The required capacity to store the accumulated output from continuous high throughput devices such as image sensors. This rate imbalance is a central motivating issue in this project and is discussed in detail below.

The generation of sCMOS cameras available at the start of this work capture full-frame resolution video at either 30 fps or 100 fps depending on the data interface between camera and computer (USB3.0 or CameraLink). At 16-bits per pixel and 2048x2048 pixels, the maximum data rate for the USB3.0 camera is 240 MB/s. Imaging sessions typically last 30-minutes or less. Pixels are typically binned down 2x2, and frame rate is often reduced to work within the constraints our laboratory workstations impose on processing speed and storage. However, the effect of doubling resolution on processing time when using the graphics card is virtually negligible. Identifying ROIs online and extracting the traces of neural activity allows us to discard acquired images and instead, only store the relevant pixels for later analysis.

1.4.1 Graphics Processing Units for Video Processing

Graphics Processing Units were traditionally developed for the consumer gaming market. They are optimized for the process that involves translating a continuous stream of information into a two-dimensional image format for transfer to a computer monitor. In the context of gaming, the stream of information received by a GPU describes the state of objects in a dynamic virtual environment and is typically produced by a video game engine. These processors are highly optimized for this task. However,

they are equally efficient at performing the same procedure type in reverse, reducing a stream of images to structured streams of information about dynamic objects in the image. These features render them popular for video processing and computer vision applications.

All GPU architectures consists of a hierarchy of parallel processing elements. NVIDIA's CUDA architecture refers to the lowest level processing element as "CUDA Cores" and the highest level as "Symmetric Multiprocessors." Typically, data is distributed across cores and multiprocessors by specifying a layout in C-code using different terminology, "threads" and "blocks." Blocks are then termed to be organized in a "grid." Adapting traditional image processing or computer vision algorithms to quickly run on a GPU involves efficiently distributing threads and ideally minimizes communication between blocks.

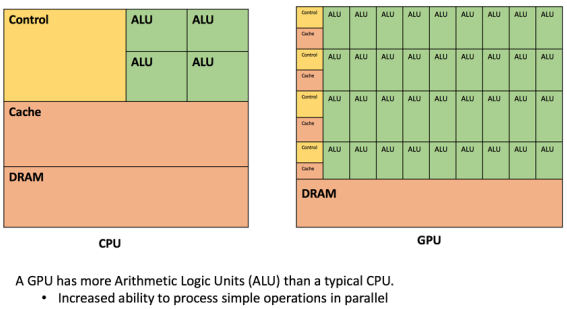


Figure 1·1: Comparison of processor architectures: CPU (left) and GPU (right) differ tremendously in the number of ALUs packed on a chip [`https://www.datascience.com/blog/cpu-gpu-machine-learning`]

MATLAB makes processing data using the GPU seemingly trivial by overloading a large number of built in functions. Performance varies however. Writing a kernel-type subfunction is often the fastest way to implement a routine written as if it operates on single (scalar) element only that can be called on all pixels at once or employs all pixel-subscripts used by the function to retrieve the pixel value at a given subscript. The kernel-type function is compiled into a CUDA kernel the first time it's called,

then repeated calls directly contact the kernel with minimal overhead. Calls typically use the arrayfun() function.

Data transfers between system memory and graphics memory is often a major bottle-neck. Therefore, this operation is best performed only once. However, once data is available to the GPU, many complex operations can be performed to extract information from the image without exceeding the processing-time limit imposed by the frame-rate of the camera sending the images.

In total, this project employs advances in both software and hardware that facilitate rapid accurate image analysis of living organisms with the ultimate goals of simplifying the acquisition and analysis of neural activity indicators in both normal and pathological states.

Chapter 2

Neural Interfaces: Fabrication, programming, and assembly

This chapter describes several projects that were started early during my graduate studies. Each project is similar in that they are outside the realm of optical imaging of neural activity, which is the focus of the rest of this dissertation. Nevertheless, they are included here because the issues they bring up will later inform the approach I take in the work described in later chapters. The projects described in the following sections are also tied together by a common goal: to enable research in the neurosciences with translation potential for clinical applications.

2.1 Animal Tracking

2.1.1 PD mouse model:

You can induce a quantifiable PD-like state in mice with a unilateral injection of the neurotoxin 6-hydroxydopamine (6-OHDA) into the striatum, and subsequent administration of apomorphine to provoke side-biased motor deficits (**?**) Side-biased "turning" behavior is quantified autonomously on two distinct platforms, a computer-vision system that allows free movement, and a virtual-reality spherical treadmill platform that simulates free movement.

2.1.2 Metrics of Behavior

Two testing platforms are used to assess changes in behavior over time. Behavior is analyzed and quantified in real-time, and are synchronized with electrophysiology and made available as stream of events synchronized with imaging and/or electrophysiology. The quantification routine creates a signal that is representative of symptom severity. For our unilaterally lesioned mouse model of PD the most readily observable impairment is the inability to walk straight; mice would turn in circles contralateral to the lesion when given intraperitoneal apomorphine.

2.1.3 Behavior Box

I built an experiment apparatus for mice to enable a study being run by Jia-Min Zhuo. The goal of the study was to elucidate the role of adult-born neurons on mouse behavior, specifically their performance in discrimination tasks. We called the apparatus the "Behavior Box" and modeled it after a commercially available but grossly over-priced box that itself came from other labs (see (**?**)).

The chamber was constructed with black plastic walls, extruded aluminum framing, and a perforated metal mesh floor 1 cm above a plastic waste tray. A 10-inch infrared touchscreen (ITouch Systems) was mounted over a 10-inch LCD monitor forming one wall of the chamber. An opaque mask with seven windows was placed over the screen to limit where the mouse could touch. A water pump with infrared detector was located at the other end of the chamber to provide reward for the water-deprived mice in the study. A white LED strip encircled the chamber from the top, and multiple speakers positioned outside to deliver sound cues. A web camera was fixed above the chamber to record and monitor mouse activity. My contribution to this project was the program that facilitated interaction between all the system components. This program controlled and recorded experiment progress. I developed the

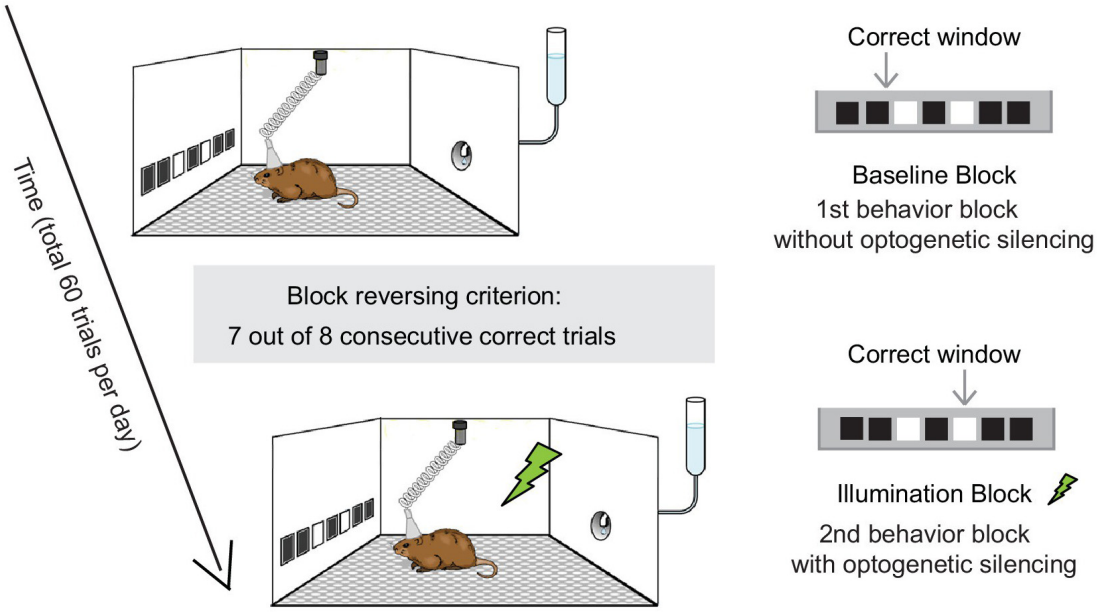


Figure 2·1: behavior-box schematic

program in MATLAB, and the main components of its function are described below. This system would eventually be used for a study investigating the development of adult-born neurons in the hippocampus ((**?**)).

2.1.4 IR Touchscreen

The IR touchscreen provided a robust measure of the location of any contact with the animal's paws or nose. The screen was more reliable than either *resistive* or *capacitive* touchscreens, which are much more common in devices like POS systems and mobile phones respectively.

The scientists users interact with the device through command-line manipulation of a "BehaviorBox" object in the MATLAB REPL. This approach provides access to features for easily customizable control of physical components including the infrared touchscreen and LCD display along with speakers, water-ports, lights, essentially anything that can be controlled electronically. The approach also enables suggestions

and autocomplete options for users new to the environment.

2.1.5 FrameSynx Toolbox

The FrameSynx toolbox for MATLAB was built to synchronize continuous image acquisition with experiments conducted in the neuroscience laboratory setting. While the experiments are conducted in separate software (and potentially on a different computer), FrameSynx listens for messages to start/stop the experiment, start a trial, etc. and responds accordingly by controlling one or multiple cameras and illumination devices, and synchronizing this information with the data acquired. The major contribution to the "Behavior Box" package, and also to later image processing packages is the procedure for definition and storage and of experimental data files, which will be touched on briefly in chapter 3.

2.1.6 Using Computer Vision to track Position and Orientation

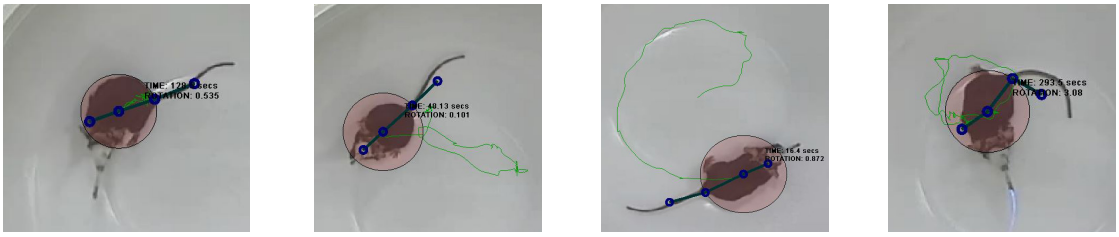
2.1.7 Mouse in a Bowl

A webcam-based motion tracking box constructed to analyze the movement of our unilaterally lesioned PD mouse model. Video is recorded at 15 frames per second and processed on-line or off-line using a function written in MATLAB. Briefly, this function converts each frame to a black and white image (logical matrix), uses morphological filtering functions to isolate the mouse (remove mouse excrement) and identify its body (remove the tail), then finds the center of mass in cartesian coordinates (maximum center of projection on x- and y-axes), and the rostral-caudal orientation measured in degrees off the x-axis. Orientation is determined by the index of maximum of a radon transform of the binary image. Processing is accomplished at a rate of 15-16 fps, using a single core, or 64 fps using parallel processing on a quad-core processor with multi-threading enabled. The advantage of this apparatus

over the virtual-reality system is that it allows free movement of an untrained mouse, with real-time movement metrics at nearly the same rate as the spherical treadmill. This apparatus would go on to be used to investigate the oscillatory dynamics of cholinergig interneurons in the striatum and their association with parkinson's-like motor deficits [(?)].

(a) (b) (c)

Figure 2·2: Automated animal Tracking for "Mouse in a bowl" type experiments (a) Raw frame of video being tracked; (b) area of detected mouse; (c) overlay of 3 consecutive frames of mouse between each



(a) (b) (c) (d)

Figure 2·3: Automated animal Tracking for "Mouse in a bowl" type experiments: (a-d) video overlay showing tracked points

2.1.8 Spherical Treadmill

A virtual reality system was assembled, adopting methods from the Harvey lab lab (?). This system allows placement of a head-restrained mouse on an 8-inch diameter polystyrene foam ball supported by a cushion of compressed air, surrounded by a

toroidal projection screen. Ball rotation is tracked with two optical computer mice placed orthogonal to each other. Movement vectors are fed into a virtual-reality engine that updates the image projected onto a toroidal screen surrounding the ball, simulating movement through any arbitrary virtual world. Movement vectors are recorded as an arbitrarily scaled translation in the mouse-relative X and Y axes and rotation around the Z axis, at approximately 30 ms intervals. This behavioral apparatus has the advantage of allowing trivial measurement of the mouse's movement ability while the mouse is head-fixed. The disadvantage is the time and potential confounds involved with training individual mice to use the system.

(a) (b) (c)

Figure 2·4: Spherical treadmill (a) Mouse running on a treadmill equpipped with virtual reality; (b) water port; (c) application of the water delivery system

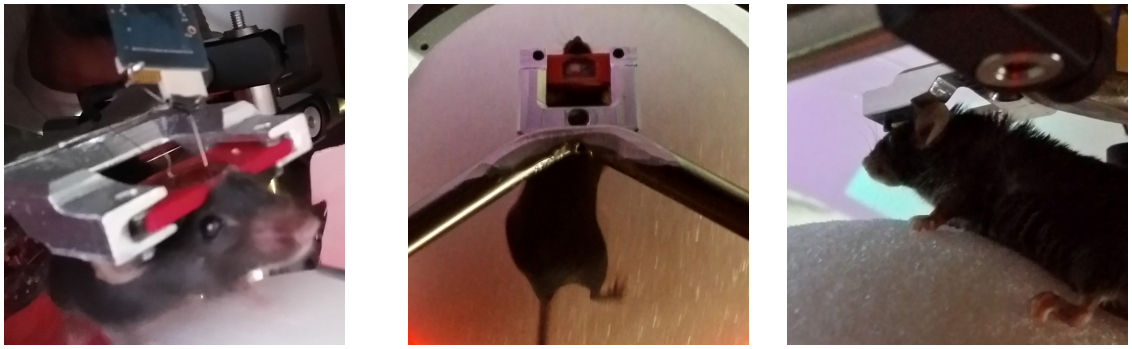
2.1.9 Headplate Holder

2.1.10 Motion Sensors

Motion sensing was implemented using a linux computer and standard mice at first, and later using precision laser navigation sensors for "gaming" mice and custom firmware written to work with any arduino-compatible microcontroller.

2.1.11 Generic USB Computer Mouse with Minimal Linux

Run "mouse_relay.py" on any computer running linux to send xy-data from 2 USB optical computer mice to another computer over an RS-232 serial-port connection.



(a) (b) (c)

Figure 2·5: Headplate holder (a) front view; (b) top view; (c) bottom view

The receiving computer (in this implementation) uses MATLAB to read the values and translate the xy-values from 2 mice on the surface of a sphere into 3 values corresponding to rotation of that sphere around 3 orthogonal axes (XYZ) with their origin at the sphere's center.

RECEIVING FUNCTIONS: The MATLAB class that receives the serial input (xy-values from both mice) is called "VrMovementInterface"

The MATLAB function that translates the double-stream of xy-values from the sphere's surface into rotation around its center is called "moveBucklin.m" and is located in the VIRMEN "movements" folder.

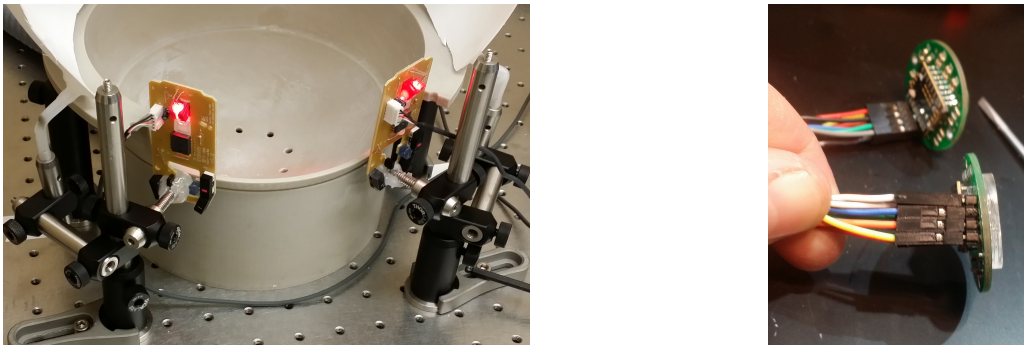
SERIAL FORMAT: XY-Values are transmitted in 'packets' using an ascii formatted string terminated by a newline. Each packet contains the Sensor Number (s) that the reading is coming from, followed by the X-Value (dx), then the Y-Value (dy). The python code looks like the following:

A single reading is received at the other end of the serial connection looking something like the following:

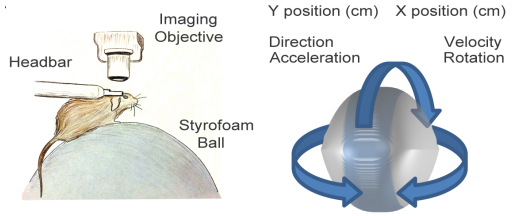
s1x34y-3

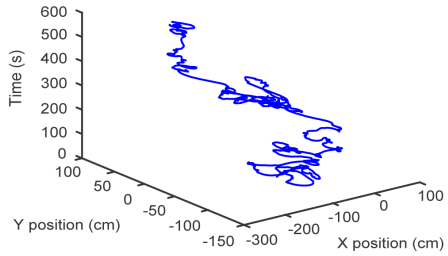
2.1.12 Navigation Sensor Chip with Arduino

The system was later improved. I wrote an Arduino compatible library that functions as a driver for for the ADNS performance precision gaming sensor. This driver passes [dx,dy] measurements from two ADNS-9800 laser mouse sensors (placed 45-degrees apart on surface of Styrofoam ball). The library has been put to use in multiple systems and enjoys collaborative devleopment from a small number of other researchers (?)



(a) (b)





(c)

Figure 2·6: Motion sensors (a) mention sensors installed; (b) motion sensors; (c) tracking mouse movement

2.2 Microscopes

This section describes the background in microscopy in the neurosciences, and also how it relates to imaging in healthcare and electrophysiology in neuroscience. It will also describe the basic elements necessary for the construction of a microscope in a laboratory where calcium imaging in an animal is available. It will also refer to later sections which cover the design and construction of mechanical elements for animal handling and optical access (i.e. the headplate and a chronic optical window).

2.2.1 Background: Brain Imaging and Microscopy in Neuroscience

Optical imaging has traditionally involved wide-field imaging or two photon imaging, each with their own distinctive advantages and disadvantages. In recent years, two photon microscopy has been a preeminent choice for imaging in tissue, because of its high spatial resolution, and tissue penetrating features. Two photon calcium imaging has been broadly applied to individual cells or subcellular components of neurons including spines and axons.

Because two photon microscopy uses a scanning mechanism, the signal to noise ratio is influenced by the time spent imaging each point, and the spatial resolution is determined by the number of points scanned to obtain each image. As a result, the size of the imaging field is inversely correlated with the overall temporal resolution while maintaining a relatively high signal-to-noise ratio, thus, two photon calcium imaging is often performed on a small area or on a sparse network of cells, when dynamic responses with high temporal fidelity is necessary.

Wide-field imaging has been used in various forms for several decades and was first used to characterize the functional architecture and hemodynamic responses in brain tissue. However, this technique has seen a renaissance recently due to its simple instrumentation, relatively inexpensive cost, and the improvements in neural

signal indicators. Optical imaging and two photon microscopy have traditionally been performed in head-fixed preparations, but recent advances have also made it possible to perform wide-field calcium imaging in freely moving animals, through miniaturized and wearable microendoscope systems

While wide-field imaging lacks the spatial resolution to resolve fine subcellular structure or the penetrating properties available with two-photon, it is possible to obtain clear neurites and somatic features, including spike detection

Because a single photon microscope does not rely on scanning features, it can be used to sample a larger field of view without sacrificing sampling rates. Additionally, recording sessions may be less sensitive to fluorophore bleaching than other techniques, which makes it possible to perform sustained illumination and subsequent imaging for an extended period of time - a desired feature for analyzing neural networks during some behavior paradigms (e.g., repeated trial learning paradigms). Thus, wide-field imaging offers an advantage if the objective is to simultaneously recording hundreds of neurons in the brain of a living and behaving animal with high temporal fidelity.

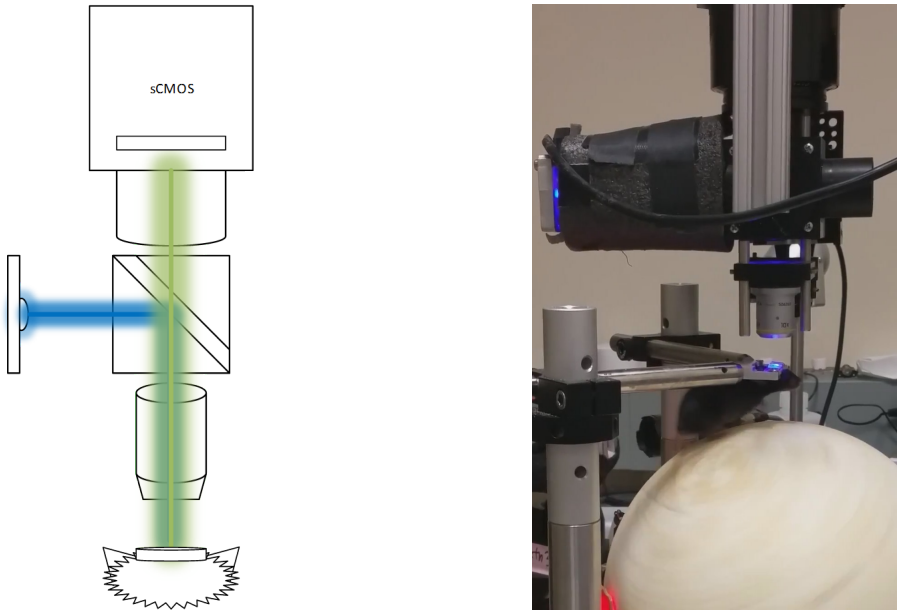
2.2.2 Cameras for Widefield Microscopy

Traditional widefield microscope or macroscope builds incorporate 'scientific grade' cameras. Compared to cameras built for other markets (e.g. consumer, industrial, studio, etc.) , these cameras are often well tested and certified to offer low or well-characterised noise at moderate speeds, and a linear photo-response profile. Unlike consumer or studio cameras which are invaribaly configured for RGB color, they are preferably configured with 'monochrome' sensors - essentially identical to the analagous color sensor, without the bayer filter. Of much greater importance, one must consider the unique connectivity and control interface that scientific cameras

come with. Standards exist, but are typically unique to this segment of the industry, with poorly defined specifications for translation to other electronic communication and connection interface standards, such as those used in studio and broadcast video, or those used with consumer cameras. The trait that is the most worthy of consideration, however, is the cost. See @discussion-cost-consumer for details.

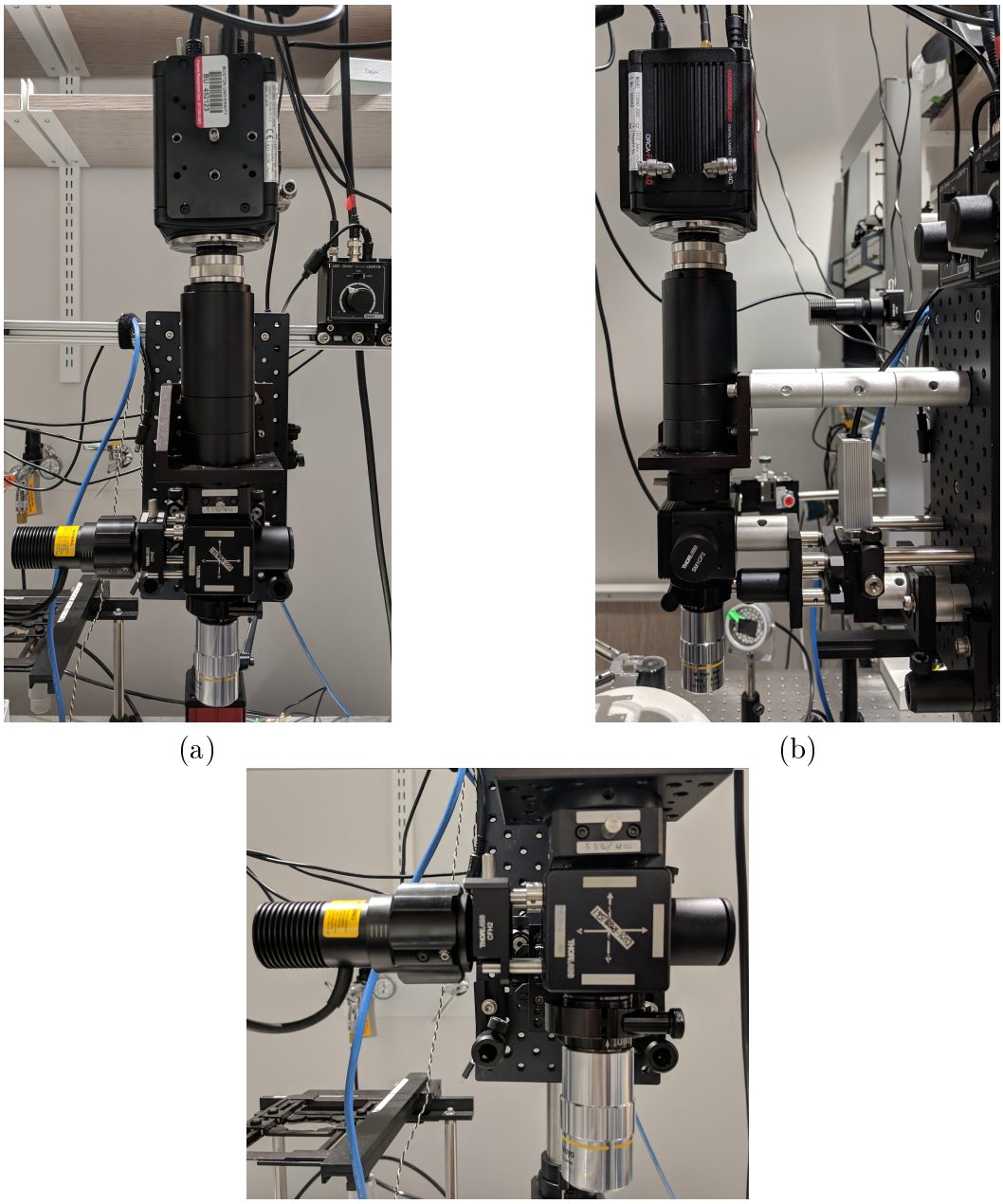
The in-vivo instrinsic-signal or fluorescent-dye imaging camera of 1 decade ago had a 0.5“-1” monochrome CCD sensor with 0.1-1 MegaPixels, a large well-depth, and moderately low noise at speeds around 30 to 60 fps. Connection was often LVDS, with custom electrial connectors unique to each camera. A particularly popular and long-running model was the Dalsa 1M30, followed by the 1M60 in later years (**?**).

2.2.3 Microscope Construction



(a) (b)

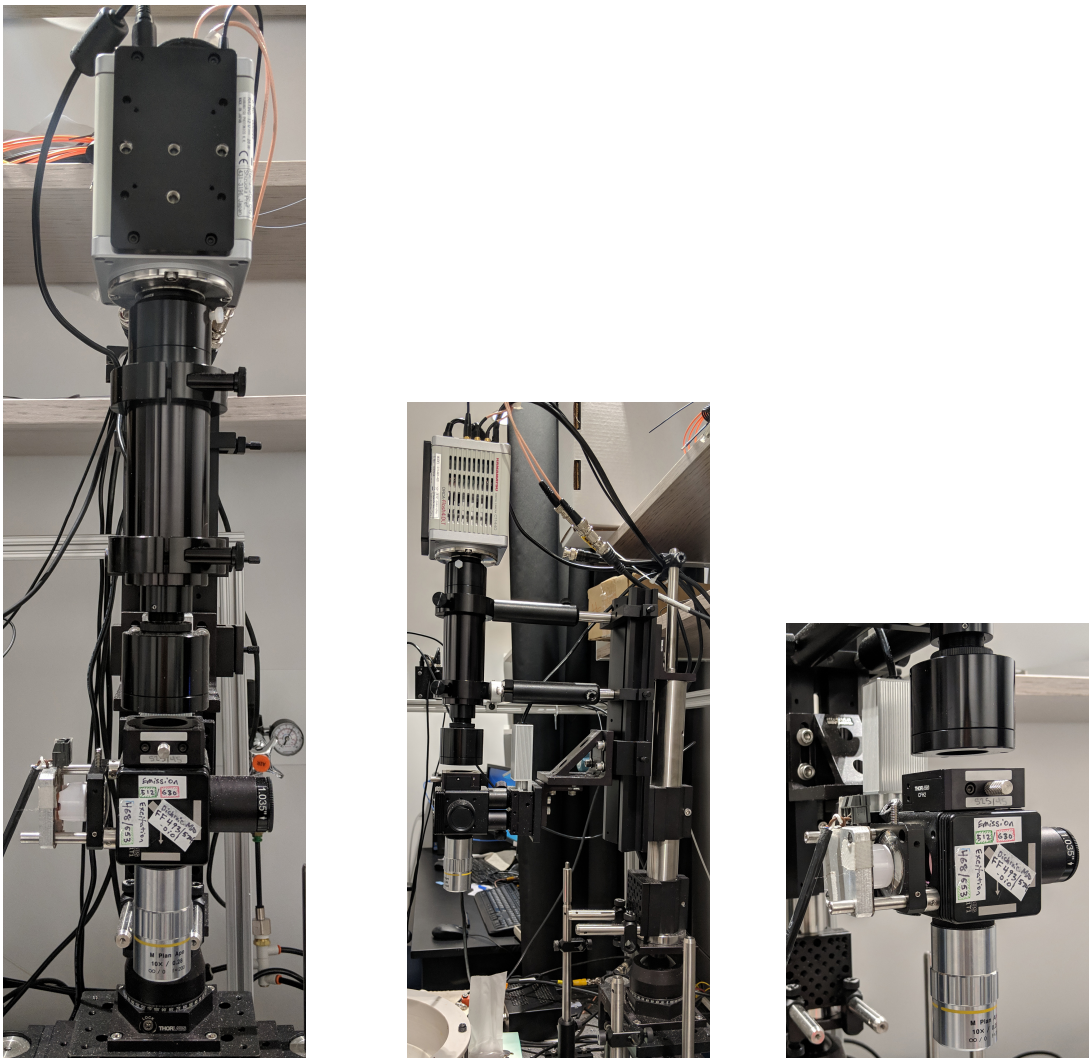
Figure 2·7: Basic configuration for a widefield epifluorescence microscope for in-vivo imaging. This first configuration used a phase contrast lens borrowed from an inverted microscope (not recommended). (a) Schematic showing relation of microscope and mouse on spherical treadmill; (b) Setup 1: the LED used for extending to the left (black covering to block light)



(a) (b)

(c)

Figure 2·8: Microscope Setup 2 (a) front; (b) close up; (c) side



(a) front **(b)** close up **(c)** side

Figure 2·9: Widefield fluorescence microscope reconfigured Setup 3. Multiple iterations are shown, with later iterations offering improved compatibility with usage of off-the-shelf components.

Chapter 3

Neural Signals: Computational considerations, interpretation and usage

3.1 Image Processing

The entire procedure for processing images and extracting cell signals can be performed in substantially less time than most commonly available tools using the approach described in Aim 1, particularly the methods for restricting the spatial extent of pixel-association operations, and distributing operations across parallel processing cores using a Single Program Multiple Data (SPMD) archetype. However, the total time still exceeds that of the acquisition session. Inefficiency arises from the overhead involved with distributing data and passing information between separate parallel processes. Graphics cards, however execute in what's called Single Instruction Multiple Data (SIMD) fashion, to distribute computation across the thousands of processing cores.

The processing components are implemented using the MATLAB System-Object framework, which allows for slightly faster performance through internal optimizations having to do with memory allocation. Most system objects, each representing one step in the serial processing and signal-extraction procedure, also have companion functions that implement the computation-heavy components of each algorithm using a pre-compiled CUDA kernel.

3.1.1 Benchmarking & General Performance

Built-in MATLAB functions that execute on the GPU can be profiled with benchmarking functions like *gputimeit()*, or with the *tic/toc* functions. When execution isn't fast enough, they need to be replaced with custom functions. The custom functions typically achieve the speed up necessary by enabling the operation to carried out on several frames at once. This reduces the over-head costs inposed for each function call by spreading it over several frames. This solution is not ideal, as it increases the latency of solutions, however does not preclude implementation in real-time system if the procedures are adapted to run on a real-time hybrid system-on-module like NVIDIA's Tegra X1, which should involve minimal effort once a standard set of successful procedures is realized. The current implementation tests the processing time of each stage of the process to ensure that the sum is less than the acquisition time for each frame dictated by the inverse of the frame-rate (30-50 milliseconds).

3.1.2 Buffered Operations

Combining frames for each operation can result in near linear speedup. For example, for the phase-correlation step required for motion correction, the FFT and IFFT are called on 16 image-frames at once, and the time take to accomplish is approximately the same as if the operation were called on 1 frame. This essentially leads to a 16x speedup, though the latency is also increased slightly. The best size to use is difficult to pre-determine, and typically must be measured for varying size 'chunks' using the benchmarking functions indicated above. The system objects manage the details necessary to allow buffered chunks of video to be passed to each stage without introducing artifacts at the temporal edges between chunks.

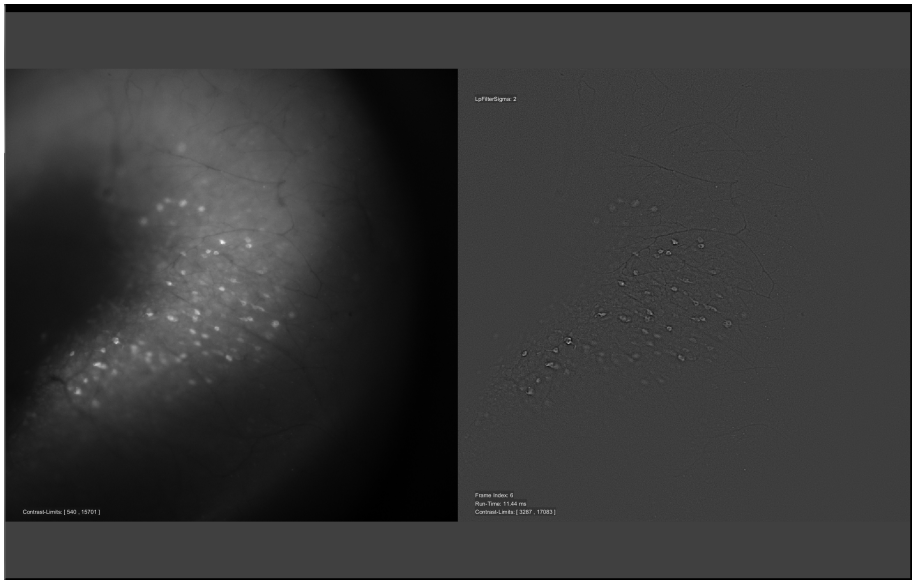


Figure 3·1: Interactive parameter adjustment for homomorphic filter operation (local contrast enhancement)

3.1.3 Image Pre-Processing & Motion Correction

Pre-processing is implemented as with the offline procedure, with a few changes. Images are aligned in chunks, and they are aligned sequentially to two templates. One template is the most recent stable frame from the preceding chunk. The other is a recursively temporal-low-pass filtered image that mitigates slow drifts. Aligning to the first template is usually more stable as the brightness of cells in the recent image will be more similar to those in the current chunk than will be the brightness of cells in the slow-moving average.

The displacement of each frame is found to sub-pixel precision, then used with a custom bicubic resampling kernel that replaces any pixels at the edges with images from the moving average.

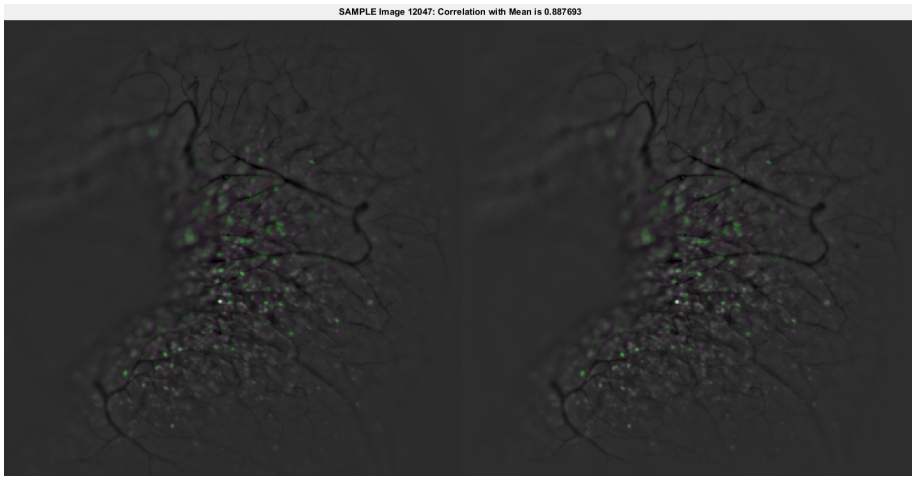


Figure 3·2: Motion compensated; comparison of uncompensated and compensated frames overlayed with mean image

3.1.4 Sequential Statistics

A number of statistics for each pixel are updated online and can be used for normalization and segmentation procedures later in the process. These include the minimum and maximum pixel intensity, and the first four central moments, which are easily converted to the mean, variance, skewness, and kurtosis. The formulas for making these calculations are given below, and are performed in a highly efficient manner as data are kept local to each processing core, and repeat computations are minimized.

Furthermore, the value used to update each central moment at each point in time can be used as a measure of change in the distribution of each pixel caused by the current pixel intensity, as explained next.

Non-Stationarity & Differential Moments

Stationary refers to the property of a signal such that its statistics do not vary over time, i.e. its distribution is stable. Neural signals tend to specifically *not* have this property, in contrast to other measurable components such as those contributed by physiologic noise (heart-rate, respirations, etc.) . Thus, by analyzing the evolution

of statistical measures calculated for each pixel as frames are added in sequence gives a highly sensitive indicator of neural activity. This is done using a routine analogous to that for updating central moments given above, except the values returned are not only the updated moment, but also the updating component – essentially the partial derivative with respect to time. This is illustrated below, including the normalization functions which convert the partial-moment values to their variance, skewness, and kurtosis analogues:

These functions run on images representing the image intensity, and also on images taken from sequential differences indicating the temporal derivative of image intensity. The combination of outputs from these operations indicate both when image intensities are significantly high relative to past distribution, and also when intensities are changing significantly faster than learned from their past distribution.

This type of function is also easily trasnalted to stencil functions on for computation on the GPU.

3.1.5 Surface Classification: Peaks, Edges, Curvature

Edge-finding methods are employed for establishing boundaries between cells, and first and second-order gradients are used to compute local measures of curvature from an eigenvalue decomposition of the local Hessian matrix. I won't go into detail, as the utility of these procedure in the most recent implementation has been lost, but nevertheless, the operation is optimized and ready to be plugged back in when further development calls for better accuracy informing cell-segmentation, or when a faster or more accurate motion-correction algorithm is called for.

3.1.6 Online Cell Segmentation & Tracking

Cells are segmented by first running sequential statistics on the properties of identifiable regions on a pixel-wise basis. That is, as regions are identified in a method similar to that used offline in Aim 1, the region-properties are calculated (Centroid, Bounding-Box, etc.) and statistics for these properties are updated at each pixel covered by a proposed region. After sufficient evidence has gathered, Seeds are generated by finding the local peak of a seed-probability function that optimizes each pixel's proximity to a region centroid, and distance from any boundary. Regions are grown from these seed regions, and registered in a hierarchy that allows for co-labeling of cellular and sub-cellular components. Newly identified regions occur as new seeds, where as seeds overlapping with old regions are used to identify sub-regions, or to track regions over time.

3.1.7 Signal Extraction from Subcellular Compartments

I also have functions for the extraction of normalized Pointwise-Mutual-Information (nPMI), which can operate on a pixel-to-pixel basis or on a region-to-pixel basis. This operation accumulates mutually informative changes in all pixels in the maximal bounding-box (e.g. 64x64 pixels) surrounding each identified regions centroid. The weights given by this function can take on values between -1 and 1, and can be used to inform any reduction operations to follow. Additionally, spatial moments can indicate the subcellular distribution of activity across the identified region. In this context, the first spatial moment M_{00} indicates the mean signal intensity.

3.1.8 User Interface for Parameter Tuning

Some system-objects also incorporate a user interface to aid in parameter selection for tuning.

3.1.9 Image Processing: Tone mapping and Filtering

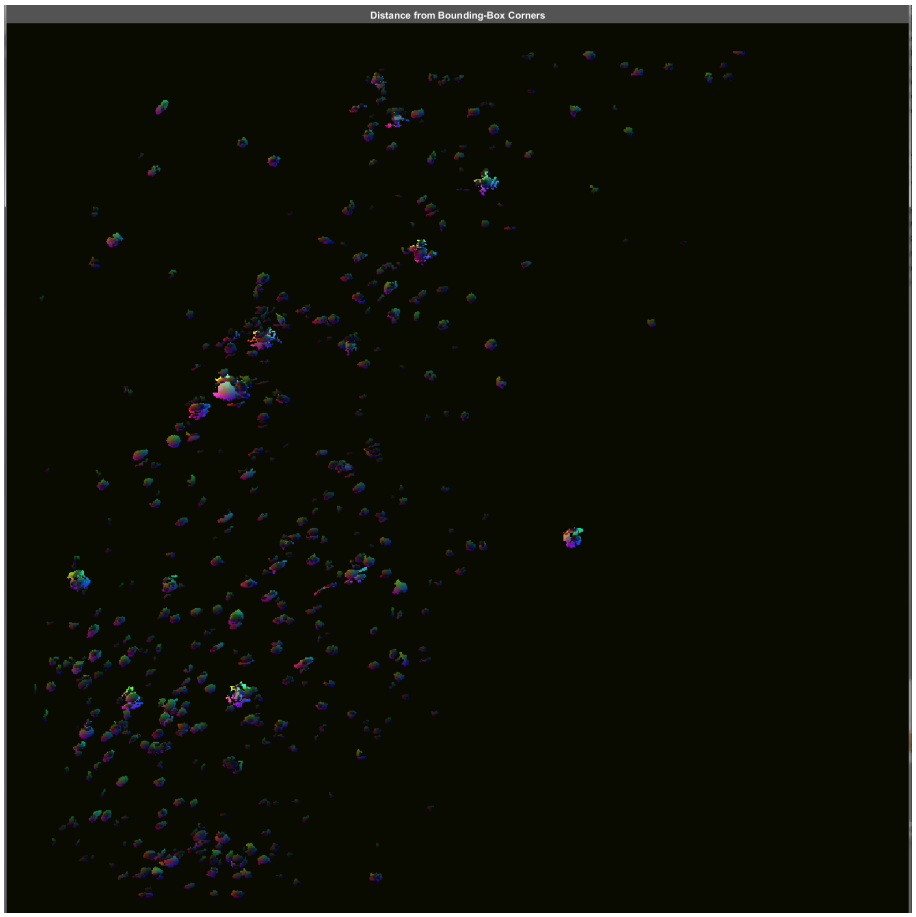


Figure 3·3: Feature generation showing complex-valued moving average of distance of each pixel to bounding box corners (post segmentation)

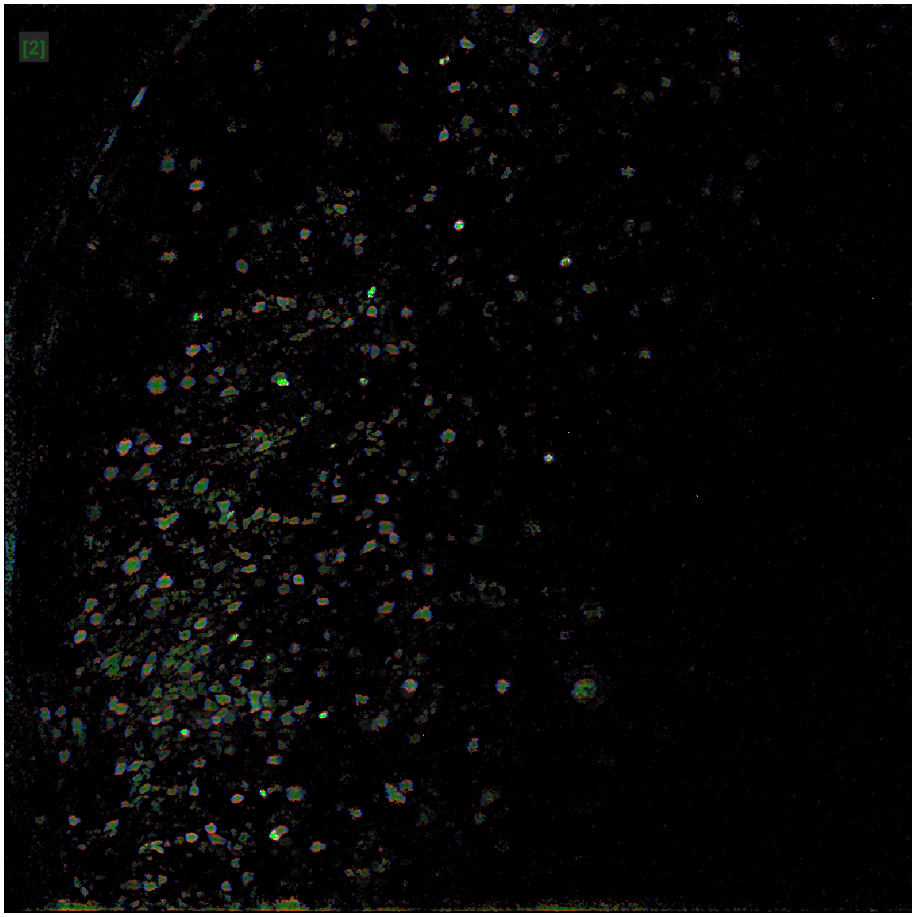


Figure 3·4: Feature generation of single motion-compensated frame using complex-valued normalized pointwise mutual information of each pixel with its local neghborhood

Listing 1 Incremental update of the statistics structure (min, max, and first 4 central moments)

```
    function      = updateStatisticStructure(            )
% updateStatisticStructure - updates a structure of cumulative sequential pixel st
%
% Syntax:
%                        >> stat = updateStatisticStructure(stat, Finput)
%




% GET/ENSURE DATA INPUT IS IN EXPECTED FORMAT (NUMERIC-ARRAY & FLOATING-POINT)
useFloat = true;
F = ignition.shared.formatVideoNumericArray(Finput, useFloat);

% GET DIMENSIONS OF INPUT
[numRows, numCols, numChannels, numFrames] = size(F);
frameDim = 4;


% INITIALIZE STAT STRUCTURE
if (nargin < 2)
        stat = struct.empty();
end
if isempty(stat)

        if isa(F, 'gpuArray')
                initZeroPixel = gpuArray.zeros(1,floatClassName);
                initZeroFrame = gpuArray.zeros(numRows, numCols, numCh
        else
                initZeroPixel = zeros(1,floatClassName);
                initZeroFrame = zeros(numRows, numCols, numChannels, f
        end
        stat.N = initZeroPixel;
        stat.Min = min(F, [], frameDim);
        stat.Max = max(F, [], frameDim);
        stat.M1 = initZeroFrame;
        stat.M2 = initZeroFrame;
        stat.M3 = initZeroFrame;
        stat.M4 = initZeroFrame;
end


% STATISTIC CONTRIBUTION COUNTS/WEIGHTS
na = stat.N;
nb = numFrames;
```

Listing 2 Incremental update of the statistics structure programmed to run as pixel-wise CUDA kernel on GPU

```
    function      = updateStatisticsGPU(       )
% UPDATESTATISTICSGPU
%
% USAGE:
%                        >> stat = updateStatisticsGPU(F);
%                        >> stat = updateStatisticsGPU(F, stat);
%
% SEE ALSO:
%                        COMPUTENONSTATIONARITYRUNGPUKERNEL, GETSTATISTICDIFFERENT
%
% Mark Bucklin

% ================================================================
% GPU COMPANION FUNCTION UNIVERSAL HEADER
% ================================================================
persistent ...
        frameDim ...
        numFrames ...
        rowSubs colSubs chanSubs ...
        numPixels
if isempty(numPixels) || (numel(F) ~= numPixels) %, defineVideoFormat, end
        [~,~,~,numFrames] = getVideoSegmentSize(F);
        [rowSubs, colSubs, chanSubs, ~] = getVideoSegmentSubscripts(F);
        [~, ~, ~, frameDim] = getVideoSegmentDimension();
        numPixels = numel(F);
end



% ================================================================
% RUN KERNEL TO UPDATE STAT INPUT OR INITIALIZE STATS & RETURN
% ================================================================
if (nargin < 2)
        stat = [];
end

if isInitialized()
        % ---------------------------------------
        % RUN UPDATE KERNEL
        % ---------------------------------------

        % FROM 2ND INPUT ARGUMENT: PRESUMED OUTPUT OF PREVIOUS CALL
        [N,Fmin,Fmax,M1,M2,M3,M4] = fromStatStruct();

        % UPDATE CENTRAL MOMENTS
```

Listing 3 Incremental update of point-wise normalized mutual information, assessed in the local neighborhood of each pixel

```
    function                = pointwiseMutualInformationRunGpuKernel(  , P% Qmin, r
% BENCHMARK: ~ 1.6 ms/frame/displacement (tested with 16 frame chunk)



% ============================================================
% PROCESS INPUT - FILL DEFAULTS
% ============================================================
[numRows, numCols, numFrames, numChannels] = size(Q);
rowSubs = int32(gpuArray.colon(1,numRows)');
colSubs = int32(gpuArray.colon(1,numCols));
frameSubs = int32(reshape(gpuArray.colon(1, numFrames), 1,1,numFrames));
chanSubs = reshape(int32(gpuArray.colon(1,numChannels)), 1,1,1,numChannels);
if nargin < 4
        radialDisplacement = [];
        if nargin < 3
                Qmin = [];
                if nargin < 2
                        P = [];
                end
        end
end
if isempty(radialDisplacement)
        radialDisplacement = 2. (0:5); %[1 2 3 4 6 8 10];
        %         radialDisplacement = 2.^(0:4);
        %         neighborDisplacement = int32([ -1 0 1 -1 1 -1 0 1 ; -1 -1 -1 0 (
end
neighborDisplacement = ...
        int32(bsxfun(@times, ...
        [ -1 0 1 -1 1 -1 0 1 ; -1 -1 -1 0 0 1 1 1]', ...
        double(reshape(radialDisplacement,1,1,[]))));
% end
% numNeighbors = size(neighborDisplacement,1);
numNeighbors = numel(neighborDisplacement)/2;
neighborSubs = int32(reshape(gpuArray.colon(1,numNeighbors), 1, 1, numNeighbors));
neighborRowSubs = int32(reshape(neighborDisplacement(:,1,:), 1, 1, numNeighbors));
neighborColSubs = int32(reshape(neighborDisplacement(:,2,:), 1, 1, numNeighbors));

% TODO:
carryOverCoeff = single(.5);

if isempty(Qmin)
        Qmin = gpuArray.zeros(numRows,numCols, 'single') + 1/4;%1/8
        %         estQmax = .5 .* single(mean(max(Q,[],1),2) + mean(max(Q,[],2),1)
        %         estQmin = .5 .* single(mean(min(Q,[],1),2) + mean(min(Q,[],2),1)
```

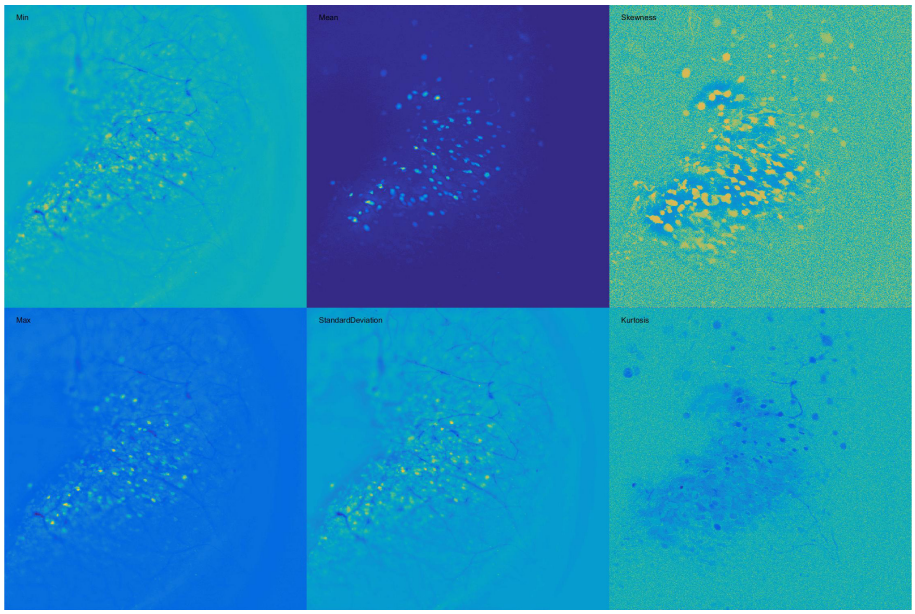


Figure 3·5: Pixel-wise statistics of 128 frames (min, max, mean, standard deviation, skewness, kurtosis)

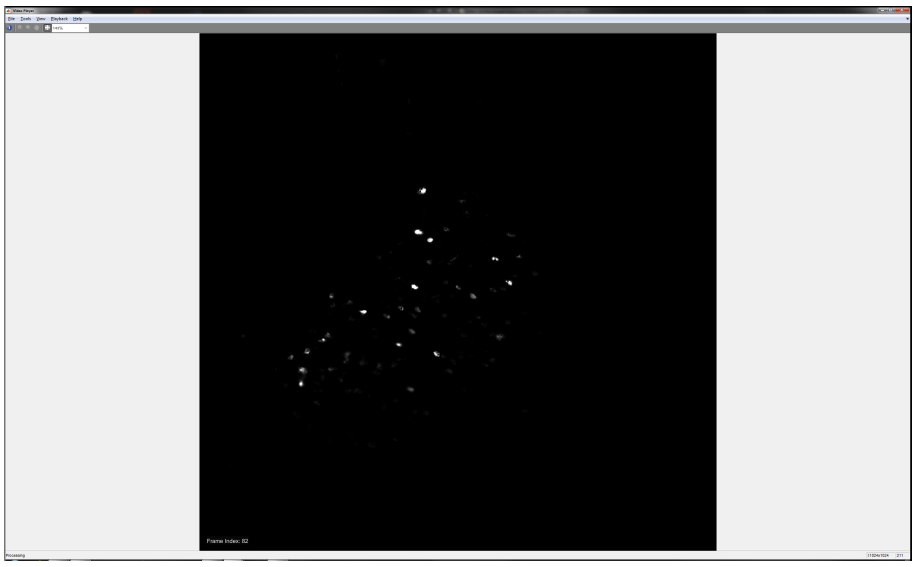


Figure 3·6: Normalized differential skewness

Chapter 4

Discussion: the last mile in computing for clinicians, engineers, and research scientists

This dissertation presents straightforward and reproducible methods for assembling laboratory equipment that can capture the behavior and neural activity of laboratory animals as well as the procedures for managing and analyzing the collected data. In some ways, the recommended procedures deviate from standard practice or the most obvious approaches. In this section, the newer approaches are compared and contrasted with current or traditional ones.

The long-term goal is to improve image quality data analysis in a finite and manageable manner that becomes (perhaps) as elegant, unique, and chaotic as the human brain itself.

4.1 Primary Goals

The function of the brain is to translate/encode sensory input into neural output, actuating an effect that promotes organism survival or the survival of offspring. It achieves this by communicating input through interconnected neurons via converging and diverging connections that comprise the neural network. One way to study the brain is by testing and observing the properties of individual neurons and the response to changing conditions at the direct connections they form with others. Another

approach is to observe a collection of neurons and measure their response to variable conditions in their external environment either by recording or stimulating variations in sensory input or measuring an organism's physical/behavioral response.

One might presume that the expansion of information provided by measuring activity from a larger number of cells in a network would simplify analysis in stimulus-response type experiments and afford insight about underlying functional mechanisms. Unfortunately, the correlation and information theoretic procedures traditionally used to make these associations suffer from a systematic bias that exponentially grows with the number responses considered for each stimulus (i.e., the number of included cells). The trial number necessary to overcome this bias becomes exponentially large although methods such as shuffling/resampling tests exist for bias correction.

A systems neuroscience experiment benefits from online feedback in one or both of two ways:

1. It informs the user regarding the current number of trials, i.e., repeated presentations of the stimulus will be sufficient to overcome limited sampling bias in an experiment attempting to learn the neural response/pattern associated with a specific stimulus. This could be done by testing pattern hypotheses online against subsets of collected data and then assessing their stability.

2. Online pattern recognition feedback maximizes the information in the response to a stimulus either by directing modification of the stimulus, or directing modification of the field-of-view.

Streaming processing addresses the issues of processing and storing pools of data from large networks on the scale that is necessary for deep learning type methods to be effective. Additionally, I demonstate a strategy in the methods section by which incorporating this online processing stream into stimulus-response-type experiments

could help correct limited sampling bias, enabling neural coding analysis in large populations of neurons [@ince_presence_2009]. This approach works when the experimental intention is to study neural coding in general, for which it's sufficient to have an arbitrary stimulus.

The earliest version of the software mentioned in Chapter **??** was published in 2016 (**?**)

4.2 State of current methods

At the start of the work described here, we found ourselves with technology providing "neural signals" that vastly exceeded our expectations and the assumptions of the tools we applied to work with it. In the past, fluctations in optical imaging data were dominated by "noise. " The form of noise depended on the process; all types of imaging, intrinsic signal, fluorescent dye, etc., had relatively small fluctuations resulting from neural activity. With new engineered molecules, like GCaMP6, and new images sensors, like those dubbed scientific CMOS, these sources of noise were comparatively small. This improved signal-to-noise ratio opens the door for new opportunities and facilitates change to traditional analytic routines. The abundance of signals available from our research animals not only makes old routines inefficient, but paradoxically, also insufficient. Such an abundance of data factors at our finger tips requires a level of discipline in study design to make the scientific method work that was previously unnecessary as the difficulty in finding signals was inherently "self regulating" and inherently limiting.

4.2.1 Signal and Noise in Neural Imaging Data

Traditional noise in neural signals can be roughly categorized as having origin in physiology or technology. The physiological noise sources include "artifacts" caused by an

animal's breathing, heart beat, or other physical movements in response to the experimentally controlled world around them. Technological noise is usually broken down into sensor noise sources: read noise and thermal noise, and noise relating to digitization. A third type of "noise" could arguably be categorized as either, as it lies at the interface of technology and biology. For example, the complex interactions of exogenous calcium-binding proteins like GCaMP with the endogenous calcium handling proteins of a neuron potentially creates noise at the technology-biology interface. By strict definition, however, only the sensor noise should termed noise, as other sources are mostly predictable and unpredictable and can be systematically neutralized or accommodated prior to data analysis. The noise level in the signals gathered by a combination of GCaMP6 and a sCMOS camera is miniscule relative to the signals indicating fluctuation in calcium concentration. The problem of visualization of these signals persists however, as the dynamic range of signal varies tremendously over space and time, and requires some treatment prior to being displayed on our currently limited computer monitors. Previously common methods, particularly intrinsic-signal imaging, provided very small signals that required "averaging over time" before any specific or reproducible response could be ascertained.

4.2.2 Correlation, Confounding Signals, and Non-linear dyanamics

4.2.3 Motion Artifact

4.3 Exponential Expansion in Data Volume

The quality of cheaply available image sensors has risen drastically and are readily available. A workable interface can be readily established and the stream of information they provide once switched on is virtually unlimited. In the stark contrast however, storage for this never-ending data stream is both finite in its capacity, and

cumulative in its consumption of available storage devices.

4.3.1 Fields sharing these challenges

Scientists often view themselves as working inside laboratory full of sensors, being "data-rich" but "space-poor". For better or worse, scientists are not alone in dealing with this inherent technologic problem. Massive investment has been poured into managing this issue for commercial purposes, and – perhaps unsettlingly – for governmental surveillance purposes. The volume of recordable traffic bouncing through choke points of the internet exceeds the capacity of any government to store for more than about 24 hours. Likewise, the massive volume of video data acquired by video surveillance systems in China requires a similar solution to one desired by scientists and physicians to resolve our data acquisition challenges.

4.3.2 Technological Opportunities to Expect

Current solutions proposed by commercial and governmental giants are not radical. They include calls for standardization in data format that could enable solutions for efficient transmission and storage to be shared by improving common tools. Common streaming formats allow compression and storage to be abstracted from each application. Databases are being developed to take advantage of heterogeneous computational architectures and distributed storage spaces. Traditional document-based or relational databases are outperformed by graph-based "triple-store" databases, time-series databases, and by databases programmed for specific architecture, including GPU-databases. These technologic developments are targeted at the bottlenecks currently restricting access to data. Early results with these approaches suggest an orders of magnitude improvement in throughput. These tools are being developed both with and without the contribution of physicians and scientists. It would be prudent how-

ever, to take advantage of new developments by orienting these tools to the specific needs of scientists and clinicians.

4.4 Incomplete synthesis of actionable knowledge

As humans we enjoy dealing in and acting on information with great *certainty*. We have phenomenal acuity for predicting the future, far surpassing that of any other species. Science, the practice of *knowing* has without question accelerated our individual and pan-species capability in this regard, and yet, science deals with surprisingly high levels of *uncertainty*. One might argue that trading, measuring, and learning to affect uncertainty is the general bread and butter of science. Another could say that, while that is true, the standards for what is a reasonable level of uncertainty are substantially below what they should be.

4.5 Clinical translation potential

Devices that rely on optogenetics to deliver stimulation to neurons inherently share the same hurdles to clinical translation. These hurdles include the requirement for gene-therapy and its associated risks. Several early trials of viral transfection of cells had adverse effects including a greatly increased risk of carcinoma. In these early studies, the DNA insertion location was uncontrolled leaving important regions of DNA tumor suppressor genes exposed to damage. New methods that improve the safety of gene therapy have been developed. Several of the more recent methods utilize adeno-associated virus (AAV) with greater control regarding the site of DNA insertion and also cause less DNA damage. These more recent methods suggest the possibility that with continuing research, methods may be developed without the inherent potential to stimulate malignancy. Working on a project that requires a

technology that does not as of yet exist represents one of the greatest educational challenges and benefits of this project. That leap of faith into a future that also does not exist requires us to depend on each other as a team of collaborators in a mutually interdependent manner. In order to succeed, we must do so together. Without each other, our therapeutics would never reach their ultimate "target audience", the patient. In this scenario, we share both successes and setbacks in the same meaningful way whether such events occur within our own labs or others located elsewhere.

4.6 Cranial Window

The two-stage cranial implant device described here was developed to enable reliable, long-term optical access and intermittent physical access to mouse neocortex. Our particular application required bilateral cortical windows compatible with wide-field imaging through a fluorescence microscope, and physical access to the underlying tissue for virus-mediated gene delivery and injection of exogenous labeled cells. Optical access is required as soon as possible post-installation and ideally, is sustainable for several months thereafter. My current designs are focused on addressing the issue common to other window designs meant for rodents, that is, progressive degradation of the optical light-path at the brain-to-window interface caused by highly scattering tissue growth. The elastomer insert is molded to fit the chamber and craniotomy site, blocking tissue growth in way that provides a reliable optical interface lasting up to one year. Additionally, the core design can be rapidly adapted to improve its performance or interface with diverse applications.

4.6.1 "Biomimicry" in visual processing

This section describes how computer image and video processing relate to visual processing in the mammalian brain. The overall goal is to emphasize the advantage and

importance of biomimetic development. Neuromorphic computing with "on chip" image processing repsents an improvement over "edge computing". Event-based image sensors such as the "artificial retina"or tittto attempt to replicate physiologic environments wherein event streams are demanded that are asynchronous and threshold-based. Convolutional neural-nets and deep learning for specific tasks have secveal but also have substantial technlgoic similarities differences. Genetic programming approaches to procedure optimization will hopefully minimize latency whle maximizing sensitivity and accuracy at minimal computational cost, energy expenditure (i.e., with high metabolic efficiency) that facilitates visual stream processing amenable to feature extraction with motion estimation and compensation. The current asymmetry of learning/training time negatively impacts on the desired inference computation time. Common standards applied across projects with common themes can be facilitatied by employing rigorous adherence to non-proprietary open source conventions that includes (bt is not limited to) optical parts (lens threads), file formats, widely available software libraries in standard programming languages, and ease of file transmission that are web-based. We are well advised to borrow from related sectors with better developed solutions such as surveillance, media streaming for web/entertainment, sports, astronomy/telescopes, medical imaging and even automotive applications.

4.6.2 Critical Elements

In assessing the design presented here, we highlight a few critical elements that facilitate the maintenance of the long-term optical quality. The Methods section describes the specifics of surgical procedures for headplate installation and insert attachment. These procedures were established after testing the variable formulations in protocol. First, the design of the silicone insert must incorporate a mechanical barrier that fits along the edges of the craniotomy. To be effective, the barrier must be contin-

uous along the circumference, and extend as far as the inside surface of the skull. Achieving this tight fit without agressively impinging on the brain requires some sort of fine height adjustment capability. The silicone insert must be attached at the correct height during the installation procedure, or shortly thereafter. The insert must be slightly depressed until full contact is made across the entire window. However, pressing beyond this distance quickly exerts an untoward increase in intracranial pressure that promotes both inflammation and adverse outcomes. A mechanism for fine adjustment can be designed into the system and is in fact incorporated into the installation procedure as is done in the first design and demonstrated in the second design presented here. Of particular note, we found that administration of antibiotic and anti-inflammatory drugs in the days surrounding any major surgical procedure had a substantial impact on the viability of the optical interface. We used both corticosteroid and non-steroidal anti-inflammatory drugs. Attempts to exclude either drug caused poor outcomes for study animals. Lastly, sealing the chamber is absolutely critical for achieving viability of the optical interface as well as the animal's overall well being. Equally critical to the long-term health of the imaging chamber is the requirement to establish and maintain an air-tight seal between the chamber and the outside world. This includes a permanent seal between the chamber and skull and a reversible seal between the chamber rim and the optical insert. is Although specific to the system design, a permanent seal is absolutely essential to ensure long-term functionality. In addition to establishing and maintaining an air-tight seal, it is necessary to eliminate all air pockets within the chamber. Residual air pockets will be susceptible to bacteria growth and may disrupt normal intracranial and intermembrane pressures after installation. We employed sterile agarose fill to displace all air within the chamber prior to sealing. Dead space surrounding the silicone insert, including that temporarily filled with agarose, will fill with fluid and eventually be

overtaken by granulation tissue. This preventive process is helpful to the maintenance of a sterile chamber environment and therefore, care should be taken not to disrupt it. However, an excess of dead space will delay this process and thus should also be minimized when adapting the design. Several attempts to test variations from the described procedures indicated that all elements mentioned above are equally critical to achieving a reliable imaging window with sustained optical quality. Implementing the procedures as described or an effective alternative solution should mitigate the primary obstacle to long-term imaging in mice and other rodents by reducing the need to pre-terminate imaging experiments due to light-path disruption by tissue ingrowth. This capability will drastically reduce wasted time and resources for experiments of any duration and facilitate previously infeasible research that require longer-term observations including aging or progressive chronic neurological disorders.

4.6.3 Staging Implant Installation & Tissue Access

Configuring the implant as described to enable a staged installation of multiple parts enables surgical procedures to be easily spread across multiple days. This capability offers a number of advantages including saving time and resources, particularly during the prototype stages by allowing time to ensure each implanted animal fully recovers from the initial procedure and anesthesia. Additionally, the delay between surgeries allows the initial inflammation and immune system responses triggered by craniotomy to resolve before attempting a second intervention in tissue that is sensitive to these manipulations (e.g., viral or cell injections). Importantly, this system affords the capability to image the first tissue intervention from day 0. Similarly, designing a system installed in multiple stages enables trivial and repeatable tissue access at later time points by simply reversing the insert attachment procedure. The process may be comparable to a previously reported method of removing the entire cranial

glass window to access the tissue. With this newer system however, the methods used to detach and reattach the cranial window are relatively faster, simpler and carry less risk of tissue damage. Additionallly, the described methodd provide full cranial access without compromising the image field, an advantage not provided by a fixed access port.

4.6.4 Design Adaptation

The specific designs described in this report work well and have much to offer. The potential for fast and unrestricted adaptation is the greatest asset of the underlying system. Most users will find greater utility in adopting components of the design and fabrication process that can be readily customized to fit their exact needs. The design can also be rapidly transformed to accommodate various applications or to modify its performance in response to new technologies and demands. This rapid adaptability was a primary goal of this projec, and informed our design and engineering decisions throughout developement. Anyone with access to common laboratory equipment and moderate engineering and fabrication skills can produce a system to fit their particular needs. As an inherent aspect of any design process, the adaptation of the original design evolved over the course of prototyping and testing. In presenting two designs in this report, our intention was to demonstrate the technical feasibility of continuous development of a "future-proof" system. The original system was adapted to accomodate the continuous evolution of image sensor technology, particularly the growth in size and resolution, expanding the field of view and allowing simultaneous access to cellular interactions across multiple brain regions using wide-field imaging. We found that subtle dimensional changes, and the addition of miniscule features exert a large impact on the success of any design. We also found that adjusting features to address one aspect of functionality may have unintended effects. For

example, the inclusion of a thin skirt extending below the optical insert that was incorporated to protect against tissue growth within the image field also promotes physical conformity of the brain to the optical window interface over time. This conformity results in a flat imaging plane that is optimal for wide-field imaging and was previously unachievable.

4.6.5 Rapid fabrication

The rapid iterative process used here was made possible by using a combination of widely available rapid prototyping procedures, 3D-printing and laser-cutting. Major progress of manufacturing and its increased versatility, providing better quality, customization, lower cost and shorter production time. In an effort to compare various manufacturing technologies, we explored a number of companies and advanced with 3D metal printing. The final products provided by at i.materialise achieved our experimental goals in product design. We also developed parts in collaboration with other rapid prototyping companies including Shapeways and Sculpteo. In addition:

- Various features and functions of the silicone insert were transformed and extended to conform to new design requirements, some requiring distinctively different design approaches

- Versatility of silicone elastomer to cover a spectrum of design strategies to optimize its configuration might be beneficial

- The design principles that evolved from the initial development are robust and can be applied to new developments or refinements while preserving the successful qualities of the original implant

- CAD designs of these reported systems are open source accessible and can be modified and extended by evolving demands and technologies We, the authors,

also call for replication, adaptation, and evaluation (i.e., continued open/shared development).

4.6.6 Future improvements

The current project primarily explores the ability to mold precise and complex features using silicone elastomer to discover configurations to improve image performance using encapsulated electrodes and optical guides. These approaches replace combination optical + integrated electrode window and do not require optogenetics stimulation. More significantly, the encapsulation of carbon, metal colloidal particles or quantum dots into polymer hydrogel networks imparts exclusive thermal, sonic, optical, electrical or magnetic properties. Specifically, the polymer interface may provides a means for directly penetrating neurons to gain electrophysiological recording or facilitate drug infusions, allowing recording and/or manipulation during imaging session. In the near future, improvments in window thickness and chromatic aberration wil enhance both wider-field and 2-photon imaging, aprocess that will be enhanced by improved and lenses and embeded, integrated electronic components, such as LEDs for illumination or stimulation, or sensors. These mbedded devices will facilitate positioning, especially in combination with kinematic headplates that alow for repeatable head positioning and newer fabrication materials.

Appendix A

Appendix

Currivulum Vitae

Mark E. Bucklin

6 Front St Chelsea, MA 02150 markbucklin@gmail.com

Education

BOSTON UNIVERSITY,
School of Medicine; Division of Graduate Medical Sciences
Boston, MA
2010-Present

- Currently in 1st year of M.D./Ph.D. program
- Pursuing Ph.D. in Biomedical Engineering

COLUMBIA UNIVERSITY,
Fu Foundation School of Engineering and Applied Science
New York, NY
2005-2009

- B.S. Biomedical Engineering – Biomedical Imaging track
- 3.69 Cumulative GPA
-

MOUNT DESERT ISLAND HIGH SCHOOL
Bar Harbor, ME

2001-2005

Experience

CENTER FOR NEUROBIOLOGY AND BEHAVIOR - CUMC
Technician: Lab of Aniruddha Das, Ph.D.
New York, NY

May 2008-August 2010

- Developed image acquisition software in MATLAB and firmware in embedded C
- Built routines for data management, analysis, visualization, and access over a network
- Constructed a device to measure respiratory patterns in non-human primates
- Upgraded optical imaging system to use optical fibers, enabling simultaneous dual-wavelength intrinsic signal imaging and neural-recording

NORTHEAST HARBOR AMBULANCE SERVICE
EMT/Ambulance Attendant
Northeast Harbor, ME
2005-Present

- Lead care or assist a paramedic
- Completed Intermediate level course for higher level license
- Started IVs, collected blood, and obtained ECGs during >100 hours of clinical rotations

TEACHING AND PROJECTS ABROAD
Medical Volunteer
Nagoda Hospital, Sri Lanka
May-June 2006

- Assisted and observed local medical personnel in rounds and various procedures
- Spent time in General, OB-GYN, Pediatric, Surgical, and Post-Mortem wards

JACKSON LABORATORY
Research Intern
Bar Harbor, ME
2003-2005

- Designed and led study determining the effect of a global growth hormone deficiency on atherosclerosis in mice
- Bred and genotyped mice and performed histology to quantify arterial damage
- Used PCR genotyping to identify QTLs, and taught procedures to incoming students

HANCOCK COUNTY MEDICAL MISSION
Surgeon's Assistant
Ibarra, Ecuador
March 2005

- Directly assisted one general and one vascular surgeon in more than 50 operations
- Provided free-healthcare while gaining valuable medical experience

Skills and Interests

PROGRAMMING: MATLAB, Java, C/C++

INTERESTS: Electronics, Sailing, Traveling, Web Development

References

Aniruddha Das Ph.D., Center for Neurobiology and Behavior, CUMC – `ad2069@columbia.edu`

Michael Dennis M.D., St. Mary's Episcopal Church – `newrinkle@aol.com`

Heather Frazer, Pd.D., – `Frazer@fau.edu`