A low-cost, modular, environment for imaging calcium in neurons and collecting simultaneous motor information

Introduction

Calcium imaging is a burgeoning technique used for imaging and assessing the collective activity of hundreds of neurons simultaneously. A new challenge is the development of techniques that allow for concomitant execution of different tasks and experimental paradigms along with calcium imaging. Of recent interest include examining the motor output of mice while imaging from relevant parts of the striatum (Barbera et al., 2016 and Klaus et al. 2017), and imaging the hippocampus during operant conditioning.

An ideal experimental setup requires several components. First and most importantly, it must have high temporal fidelity. Calcium imaging has strict temporal requirements due to the fact that GCaMP must first be exposed to an LED for a fixed amount of time before an image is captured. A common digital design is to set up a CMOS or other imaging device to capture a frame every time it receives a digital pulse from the device responsible for organizing and synchronizing the experiment. Therefore, substantial jitter in digital pulse delivery can cause potentially substantial frame loss. That is, if the camera has not finished with the previous imaging cycle, it could skip a frame if the next pulse signal occurs too early. Also, in order to train a mouse to respond to a conditioned stimulus, repetition of stimulus and response must occur in a highly regular temporal fashion.

Secondly, the experimental setup must be easy to manipulate or alter. Technical skillsets vary widely in the field of neuroscience, and to be adapted widely experimental designs must accommodate these widely varying backgrounds. It is infeasible and inefficient to rely on a technician every time one must subtly tweak or disturb an experimental paradigm. Ideally, the experimental setup would enable a user to quickly translate or implement an idea they have in mind, and be simple enough to encourage the user to build novel experimental designs instead of conforming to preexisting designs. Experimental setups should accelerate and not impede the pace of research and discovery.

Finally, the experimental setup should be both widely accessible and open-source. These requirements have several sub-components that go hand-in-hand. First, it should be affordable, for reasons that are obvious. Current environments and programming environments can be exceedingly expensive [GIVE EXAMPLES HERE]. The Teensy 3.2 itself costs only $19.80 (<https://www.pjrc.com/store/teensy32.html>). The most expensive experimental component that we use in our setup is the ADNS-9800 sensor, which costs only $27.50. (<https://www.tindie.com/products/jkicklighter/adns-9800-laser-motion-sensor/>). The Arduino and Teensyduino programming environments are free, with the option of leaving a donation for continued development. Wide accessibility is necessary to maximize the effect of an open-source environment. Even if money is not an object to academic audiences, the lower the cost of an item, the more readily hobbyists will adopt the product. As they do more and more, we will see the development of new open source libraries accelerate. Cost can be prohibitive without grant money; therefore, if an open-source programming or design environment existed but were expensive, this would preclude wide-spread contribution of new software libraries or hardware components to the existing system by pricing out hobbyists. For example, in our implementation of a motion-sensing calcium imaging paradigm, we utilize the ADNS-9800 sensor, which is produced by a small company (Jack Enterprises, LLC) in Cookeville, Tennessee. This sensor affords us easy and affordable access to a high-speed, high-fidelity gaming sensor. Open-source products potentially offer faster, highly parallel development by taking advantage of the global village.

Here, we introduce two specific implementations of calcium imaging experimental designs, implemented via a Teensy 3.2 microcontroller in conjunction with several simple code scripts, and thereby demonstrate the ease and usefulness of adopting such a design for future experiments.

**Methods**

**Motion tracking using the ADNS-9800**

There are a number of ways in which people have attempted to observe motor output while imaging from the striatum. In one particular technique, experimenters mount a fluorescence microscope on the head of a mouse, and allow the mouse to move freely while recording activity via video (Barbera et al. 2016) or via video in addition to an accelerometer (Klaus et al. 2017). However, resting a microscope, no matter how light, on the head of a mouse restricts its normal range of movement for the mouse, limiting its peak velocity and introducing a confound variable to the experiment. For example, bearing additional weight recruits more muscle fibers and potentially supportive architecture, which could blur distinctions between neural representations of high and low motor patterns, particularly in motion-related regions of the brain such as the striatum.

Another technique utilizes a “three-dimensional treadmill” setup, initially proposed by Dombeck et al. (2007) and utilized widely elsewhere (Aronov and Tank, 2014; Gritton et al. (2018) (in review). In this setting, the mouse is fitted with a head plate and imaging window, and is suspended atop a Styrofoam ball that is supported by compressed air (Figure 1). This type of imaging offers small image jitter primarily in-plane, which is advantageous because it can easily be corrected by standard cross-correlation-derived motion-correction methods. It also offers a setting in which mouse must apply similar forces to begin or to terminate a motor sequence as it would in a freely-moving setting (Dombeck et al. (2007). Therefore, the mouse able to move at normal velocities. Generally, two computer mice are fit at the equator of the styrofoam ball at an angle of 90 degrees, which provides the experimenter with linear movement in the X-Y plane, as well as rotational information. Most of these techniques utilize LabView to obtain voltage readings from the computer mice (Dombeck et al., 2007, Aronov and Tank, 2014), which, though a comprehensive piece of software, is expensive proprietary. In our own lab, implementing high-level MATLAB implementations of TTL pulse-based data acquisition using a National Instruments data acquisition board in conjunction with ViRMEN software led to temporal delays. As described above, we needed a platform that was low-cost, scalable, and had high temporal fidelity.

Here we introduce a system for simultaneous wide-field calcium imaging and simultaneous motion three-dimensional treadmill tracking that necessitates only an Teensy 3.2 microcontroller (~$20.00), and two ADNS-9800 laser motion sensors (~$27.00x2) (<https://www.tindie.com/products/jkicklighter/adns-9800-laser-motion-sensor/>). This system offers an affordable, modular, open-source method of tracking mouse movement with high fidelity, temporal accuracy and without introducing confounding experimental variables.

*Teensy 3.2*

The Teensy 3.2 (<https://www.pjrc.com/store/teensy32.html>) is a less well-known microcontroller with several advantages compared with the Arduino. First, it has a higher clock rate than the Arduino (72 MHz vs 16 MHz), allowing for faster and more precise data acquisition. Second, it has an output voltage of 3.3 Volts, compared to the Arduino’s 5 Volt output. This offers a small practical advantage, as activating 5 Volt mode on the ADNS-9800 sensors requires additional soldering and modifications to the sensors. Third, this device is capable of utilizing “IntervalTimer” objects for microsecond-level precision in calling different functions. This allows us to reliably acquire velocity estimates from our sensors at 20 Hz or at any other speed.

*ADNS-9800 Sensors*

The ADNS-9800 Sensors are highly sensitive and have high maximum sampling rates, with a maximum read rate of 12000 frames per second, and 8200 counts per inch resolution (<https://datasheet.octopart.com/ADNS-9800-Avago-datasheet-10666463.pdf>). We have included in our software package drivers for these sensors that allow for easy interfacing and reading from the “motion burst” register, which returns displacement in the x and y directions in precalibrated metric units. Further, accumulated displacements can be stored in the sensors between readings and digital pulses (which occur at nearly simultaneous time points). This is possible because ADNS-9800 sensors store motion data in 16 bits instead of the standard 8 bits.

*Experimental design*

The overall design for this experiment is shown in Figure [INSERT FIGURE NUMBER HERE]. Two ADNS-9800 sensors are attached at the equator of a container in which a large, buoyant Styrofoam ball is floating. These sensors lie at an angle of approximately 90 degrees from one another. To compute linear velocity, we can simply take the Euclidean distance of the y-readings of both sensors. We can compute rotation using the x-readings if we wish. These two sensors are attached via simple serial peripheral interface (SPI) connections, the details of which can be seen in Figure [INSERT FIGURE NAME HERE]. This design can be achieved with inexpensive jumper wires and minimal soldering.

In order to begin experiments with the Teensy, we wrote two simple MATLAB [AND PYTHON-> add this!] graphical user interfaces that can be used on a desktop or laptop. Using this, the user can enter the length of the experiment and the frequency of data acquisition. This frequency will determine the frequency with which digital pulses are sent to notify the CMOS camera to capture a TIFF image, and also the frequency with which accumulated motor information will be recorded by this PC. The PC or laptop sends this information over a serial connection to the Teensy utilizing a bidirectional microUSB-USB cable.

**Hippocampal recording during operant conditioning**

Results

[Demonstration of accurate timing for teensy]

[demonstration of accurate timing for python script]

[simple plot of fluorescence acquisition?]

Discussion