A Teensy microcontroller-based interface for wide-field optical imaging and behavioral experiments

Michael Romano, Mark Bucklin, Dev Mehrotra, Robb Kessel, Howard Gritton, Xue Han

**Abstract**

Systems neuroscience experiments often require the integration of precisely timed data acquisition and behavioral monitoring. While specialized commercial systems have been designed to meet various needs of data acquisition and device controls, they often fail to offer flexibility to interface with new instruments and variable behavioral experimental designs. For example, it has been difficult to integrate recently developed sCMOS cameras with various input and output devices for high speed, large scale calcium imaging experiments during behavior. We here developed a Teensy (version 3.2) microcontroller-based interface that offers high-speed, precisely timed digital behavioral data acquisition, and digital and analog outputs for controlling sCMOS cameras and other devices. We demonstrate the efficacy and the temporal precision of the Teensy interface in two experimental settings. In one example, we used the Teensy interface for reliable recordings of an animal’s directional movement on a spherical treadmill, while controlling image acquisition from a sCMOS camera. In another example, we used the Teensy interface for temporally precise delivery of auditory and visual stimuli in a trace conditioning learning behavioral paradigm, while controlling image acquisition from a sCMOS camera. These examples demonstrate that the Teensy interface, developed here consisting of custom hardware module and software functions, provides a low-cost and flexible platform to integrate a sCMOS camerainto behavioral experiments, allowing high-speed and temporally precise imaging analysis during behavior.

**Introduction**

Recent advances in sCMOS camera technology and genetically encoded calcium sensors enable large scale fluorescence imaging of thousands of individual cells’ activity during behavior (Klaus, et al., 2017; Barbera, et al., 2016; Mohammed, et al., 2016; Markowitz, et al., 2018). One key technical aspect of neural network analysis during behavior is the temporal precision, where neural activities need to be precisely aligned with behavioral features (Solari, Sviatkó, Laszlovsky, Hegedüs, & Hangya, 2018). However, it has been difficult to integrate sCMOS cameras, deployed in large scale calcium imaging experiments, with devices needed to monitor and control behavioral experiments. Traditional Analog/Digital interfaces are often operated by programs, such as MATLAB, that offer a wide range of applications. However, using MATLAB or other PC-based programs can lead to undesired temporal delays if they rely on the PC for timing, as the operating system needs to balance the demands of many system’s operations at once.

Over the last decade, microcontrollers marketed to hobbyists have gained popularity across a variety of scientific fields (Sanders & Kepecs, 2014; D'Ausilio, 2012; Chen & Li, 2017; Husain, Hadad, & Zainal Alam, 2016). Microcontrollers are small, low-cost, and capable of delivering precisely timed digital outputs with microsecond time precision, while using user-friendly, open-source software functions. Arduino was the first major microcontroller to gain substantial popularity. Recently, Teensy microcontrollers were developed, which have all the key features of Arduino microcontrollers, as well as the additional feature of delivering analog output. Teensy’s utilize the same open-source Arduino software environments, and thus they are easy to program (D'Ausilio, 2012). Microcontrollers thus represent an attractive solution for systems neuroscience that can be easily adapted to various behavioral experimental needs, including the integration of newly developed instruments. Arduino devices have recently been integrated into two-photon imaging experiments (Wilms & Häusser, 2015; Takahashi, Oertner, Hegemann, & Larkum, 2016). In one study, an Arduino was used to generate the digital command to initiate the acquisition of an image sequence, where the timing of each image frame within the image sequence was not specified (Micallef, Takahashi, Larkum, & Palmer, 2017). Thus, this approach requires post-experimental data interpolation to re-align image frame timing with behavioral data, reducing temporal precision. One way to precisely time sCMOS image acquisition with behavior is through timed capture of each individual image frame.

Here, we demonstrate and characterize a flexible Teensy interface for concomitant synchronous and temporally precise digital data acquisition and delivery of analog and digital signals, in two experimental paradigms, during voluntary movement and during trace conditioning experiment. The Teensy interface can deliver digital pulses with microsecond precision to initiate individual image frame capture at a desired speed, while simultaneously collecting animal behavioral data. We also demonstrate the ability of the Teensy interface to generate analog high frequency sound waveforms simultaneously with other types of digital input and output. Together, these results demonstrate that the Teensy interface, containing a Teensy microcontroller equipped with specific hardware modules and a set of custom software functions, offers a flexible, accurate, and user friendly environment for imaging experiments during behavior, along with the option of analog output..

**Methods**

*General overview of construction of Teensy boards*

The two experimental designs used are shown in Figure 1, and the specialty components required to build these designs are shown in Tables 1 and 2. In both experiments, a Teensy 3.2 (PJRC.COM, LLC, part #: TEENSY32) is mounted on top of a printed circuit board via standard female headers (such as SparkFun Electronics, PRT-00115). Female headers were then soldered to the PCB for stability. Output from the Teensy was directed from pins on the female headers to standard SMA connectors (such as: Digi-Key, part # CON-SMA-EDGE-S-ND) via 22 gauge wires (for example: Digi-Key, part #1528-1743-ND). SMA cables (for example, Digi-Key, part # 744-1429-ND) were then used to connect Teensy to external devices. Teensy was connected to a computer via a standard USB-microUSB cable (for example: Digi-Key, part # AE11229-ND). To easily upload our code to the Teensy, we used PlatformIO (<https://platformio.org/>), an add-on to the widely-used Atom text editor (<https://atom.io/>), instead of the default Arduino programming environment. To easily set the sampling frequency and length of an experiment for the Teensy, we developed a simple MATLAB graphical user interface.

*Motor acquisition experiment*

In this experiment, we performed motion tracking using two ADNS-9800 gaming sensors (<https://www.tindie.com/products/jkicklighter/adns-9800-laser-motion-sensor/>, Tindie, part: “[NS-9800 Laser Motion Sensor](https://www.tindie.com/products/jkicklighter/adns-9800-laser-motion-sensor/)”, see Table 1), while delivering digital pulses to a sCMOS camera for image capture every 50 ms. The overall design for this experiment is shown in Figure 1A. Mice were positioned on top of a buoyant Styrofoam ball floated by house air as described previously (Dombeck, Khabbaz, Collman, Adelman, & Tank, 2007). Two ADNS-9800 gaming sensors were positioned at the equator of the sphere, at an angle of approximately 75 degrees from one another. For the counts per inch setting of the sensor, which determines the sensitivity of the sensors to external movement, we used a value of 3400 counts per inch. Thus, the total distance travelled at any time point was computed using the following equation:

Where yR and yL are the y readings from the left and right sensors, and is the angle between the two sensors (75 degrees). Velocity was computed as the distance divided by the time between two subsequent readings. These two sensors were connected to a Teensy via simple serial peripheral interface (SPI) connections with insulated 22 gauge wires as shown in Figure 2A.

To control experimental timing with high precision, we utilized the “IntervalTimer” function unique to the standard Teensy library, which repeatedly calls a function at specified intervals, which we set as 50,000 microseconds (50 ms). On every call of the “IntervalTimer” function, the accumulated displacement of the motion sensor reading since the previous call in both x and y directions? were collected and sent to the attached PC, and then a digital “on” pulse that lasts for 1 ms was sent out of a digital pin using the DigitalIO library (<https://github.com/greiman/DigitalIO>) to initiate image frame capture. Readings from motion sensors were extracted with freely available functions on Github (<https://github.com/markbucklin/NavigationSensor>). These functions read motion data from the “motion burst” register of each sensor.

To characterize the temporal precision, we recorded the digital outputs with a commercial system (Tucker Davis Technologies RZ5D (TDT RZ5D)) at 3051.76 Hz

*Trace eyeblink conditioning experiment*

In this experiment, Teensy was programmed to deliver a sound, an LED light and an eye puff, while delivering digital pulses to a sCMOS camera for image capture every 50 ms. To deliver an audible sound through the Teensy, we used a prop shield module available for Teensy (PJRC.COM, LLC., part #: PROP\_SHIELD). This add-on component amplifies analog output to drive speakers with resistances up to 8 ohms (shown in Figure 2B as pin A14). The prop shield was attached to the Teensy with 14x1 double insulator pins (PJRC.COM, LLC., part #: HEADER\_14x1\_D), and the output was connected to a speaker. The speaker, camera, and air valve for the eye puff were attached to the microcontroller through SMA cables as described above (also shown in Figure 1A).

We used the “elapsedMicros” function to time all of the experimental events. In order to periodically elicit a tone, we generated a 9500 Hz sine wave at the beginning of each experiment using the Teensy Audio library function “AudioSynthWaveformSine,” and originally set the amplitude to “0”. This function continuously outputs a tone with a sampling rate of 44.1 kHz from the analog pin. In order to toggle the tone “on” or “off”, we switched the amplitude to 0.05 or 0 (out of a maximum of 1), respectively. A value of 0.05 in combination with the amplifier and our speaker generated a tone of approximately 90 dB. After the tone was initialized, a single function was called every 50 ms. This function updated the status of the digital pins controlling the air valve for the “puff” stimulus and the LED light stimulus, and updated the amplitude of the sine wave. Immediately following these updates and within the same function, a brief, 1 ms digital pulse was delivered to instantiate a frame capture from a sCMOS camera. At the completion of each trial, this single function also initiated the following trial or signaled to terminate the experiment. To characterize the temporal precision, we recorded the digital outputs with a commercial system (TDT RZ5D) at 3051.76 Hz, and the analog output at 24414.0625 Hz without additional amplification.

*Statistics*

Linear models were constructed using the “fitlm” function in MATLAB 2017b. Theoretical timings, to which measured timings were compared, were each taken to be timings beginning at 0 seconds in equal increments of 50 milliseconds for both experiments.

**Results**

Microcontrollers such as Arduinos have gained popularity in neuroscience research due to their user-friendly interface, open-source nature, and their flexibility of device integration (D'Ausilio, 2012; Chen & Li, 2017; Micallef, Takahashi, Larkum, & Palmer, 2017). Recently, the Teensy 3.2 has been developed, which has analog output, a major improvement over the popular Arduinos. Teensy also has a comprehensive Audio library, and the IntervalTimer function to facilitate precisely timed event repetitions. Here, we present a Teensy-based interface to integrate and synchronize on a frame-by-frame basis sCMOS camera image acquisition with behavioral experimental control.

*Mice Motion tracking experiment*

To measure locomotion from awake head fixed mice, we implemented a novel design wherein we used a Teensy interface to record from two ADNS-9800 motion sensors. These sensors were affixed to a “spherical treadmill” setup, as described by Dombeck, Khabbaz, Collman, Adelman, & Tank (2007).

ADNS-9800 sensor boards are low cost, and are more sensitive than regular computer mice. They can measure up to 8200 counts per inch, allowing for more precise measurement of mouse movement than a normal mouse. For example, a more standard computer mouse, the Logitech M100 (Logitech, PN: 910-001601), can only read up to 1000 counts per inch. Further, wiring ADNS-9800 sensors to the Teensy is simple (Figure 2A). The connections demonstrated using dotted lines can be replaced with jumper wires or sturdier, longer lasting wire, as detailed in the *Methods*.

In an example experiment (Figure 3), we recorded a 10 minute long session of a mouse running on the spherical treadmill. Motion data was acquired at 20 Hz concomitantly with digital outputs that can be used to trigger individual image frame capture via a sCMOS camera. We calculated the velocity of the mouse, which averaged 7.1 + 6.9 cm/s (mean+ std) with a maximum velocity of 47.0 cm/s, which is in general agreement with previous studies (Dombeck, Khabbaz, Collman, Adelman, & Tank, 2007)). When we measured the timing of the Teensy digital output, we found that digital outputs have a near-perfect linear relationship with theoretical timings, with a small, 28.9 µs per second positive timing drift. Thus, the actual frequency was 19.999 Hz instead of 20 Hz.

To examine whether the small timing drift varied with the frequency of data acquisition or the timing of the digital outputs, we performed 5 minute long recording sessions without a live mouse at 20, 50, and 100 Hz. These recordings used an identical script, except due to the shorter frame duration we used a 500 microsecond long digital pulse for camera triggering instead of a 1 millisecond pulse. We found that the actual frequencies were 19.999, 49.999, and 99.997 Hz, respectively. These all equate to approximately 30 us delay per second, suggesting that the timing drift is independent of the data acquisition rate.

Having assessed the accuracy of the camera trigger timing, we next wanted to assess its precision. To do this, we looked at the root mean squared error (RMSE) of the model fit, which represents the square root of the average squared difference between measured values and values predicted by our linear model. Thus, it is similar to the standard deviation of the residual values of the model. The linear model fit demonstrated a root mean squared error of 38.9 microseconds, indicating that the camera trigger has at least microsecond-level precision. Together, these results demonstrate that Teensy can be used to trigger precise and accurate image frame capture during long behavioral experiments while maintaining alignment of imaging data with behavioral parameters.

*Trace conditioning learning behavioral experiment*

In the second experimental setting (Figure 1B and 2B), we constructed a Teensy-based interface for a trace conditioning learning experiment, where a mouse can be trained to associate conditioned stimuli (700ms long tone and/or light stimuli) with a subsequent unconditioned stimulus (a 100ms long eye puff) separated by a brief time window (250ms). In each session, 50 trials were performed with each trial lasting 20 seconds long.

In order to assess the accuracy and precision of the camera trigger timings in this setting, we recorded the timings of each digital pulse directed at a sCMOS camera and compared them to the theoretical rate of 20 Hz, (Figure 4A). Similar to the observation in the locomotion experimental design, the measured timings exhibit a perfect linear relationship with the theoretical timings, with a 33.4 microsecond delay per sample. Therefore, although the Teensy must control three other types of output in this experimental setting, it has similar accuracy to the Teensy interface used in the other experimental setting. The root mean squared error for the model fit was 13.3 us, consistent with the motor setup in its microsecond-level precision.

We next wanted to characterize the ability of the Teensy to deliver an analog output in the form of a tone simultaneously with repeated digital camera pulses. To do this, we first determined the length of time between when the tone is signaled to turn on and when an external device is able to measure output from the analog pin. All changes in tone state were synchronized with the onset of a camera pulse, so we used the timing of the camera pulse to benchmark the delay in tone onset. Latency was defined the difference in time between the onset of the analog output and the respective frame capture pulse, both as measured by an external device. (Figure 4Bi). In order to measure the onset of the audio signal, we took the raw analog recording and high-pass filtered the signal using a 6th-order Butterworth filter, a bandpass frequency of 1000 Hz, and a “zero-phase digital filter” (MATLAB command “filtfilt”). Then, we took the absolute value of the Hilbert transform of the filtered signal to acquire an amplitude envelope. After finding the amplitude envelope, we found those values that exceeded a value of 0.005. The first time point that the amplitude crossed this threshold was considered the tone onset, and the next time point that dropped below this threshold was considered the tone termination.

The measured tone latency was precise; it averaged 7.6 + 0.9 milliseconds. Because of this precision in latency, it would be easy to align this output with the onset of a camera frame if so desired. We further characterized the precision and accuracy of the tone component of our platform by measuring the length of the tone (Figure 4Bii). It lasted for 700 + 1 ms, with a range of 2.9 ms, which is not different from the programmed tone length (700 ms). Other implementations of the Audio library could potentially offer even more precision and accuracy. For example, if one needed to utilize a precise sound sequence in an experiment, they could upload the sound sequence as a .wav file and utilize the Teensy to play the pre-recorded sound (Solari, Sviatkó, Laszlovsky, Hegedüs, & Hangya, 2018). However, our design can be implemented very simply, utilizing code only a few lines of code within a single script.

We next wanted to characterize the precision and accuracy of our platform’s long digital output, the puff, by quantifying puff latency and puff duration. The latency of the puff was measured analogously to the latency of the sound. Each puff start and termination was aligned with the onset of a digital camera pulse. Therefore, we set the latency to be the length of time between the onset of the digital pulse from the puff pin and the onset of the respective digital pulse from the camera digital pulse pin. As shown in Figure 4Biii, the digital output for eye puff showed no delay from the corresponding camera trigger (mean latency = -0.004 + 0.012 (+ std) ms, range=0.04 ms seconds). The duration of the puff digital pulse was also both highly accurate and consistent, as shown in Figure 4Biv, and lasted 100.03+0.02 (+ std) ms over the course of the 50 trials, hardly differing from the expected duration of precisely 100ms. Therefore, this Teensy-based interface is precise and accurate in each of these three experimental capacities: tone generation, puff output, and simultaneous high-frequency digital pulse generation for a sCMOS camera.

**Conclusion and Discussion**

We demonstrate the accuracy and precision of a Teensy 3.2 microcontroller in integrating sCMOS camera image capture with behavioral parameter reading and behavioral output via two different experimental settings. In one setting, we designed a Teensy interface using recently developed ADNS-9800 gaming sensors. These sensors provided precise and high speed locomotion tracking while the Teensy interface simultaneously issued digital pulses for individual image frame capture. In a second experiment, we designed a Teensy interface capable of commanding four devices with precise timing during a trace conditioning experiment. These Teensy interfaces can be immediately adopted for the designed locomotion and trace conditioning behavioral experiments, or customized for other types of behavioral experiments, where sCMOS camera-based imaging is desired. These interfaces are additionally low-cost, open-source, and can be easily scaled for parallel experiments across many animals, or further customized for various types of behavioral experiments.

In both experiments, the timings of camera digital pulses sent by the Teensy interface were precise, as measured by the root mean square error of the model fits, to the level of microseconds, and accurate to within approximately 30 us per seconds. This small 0.003% drift of the Teensy processing clock is linear and can thus be calibrated if desired. This finding additionally underscores the necessity of having a central controller for experimental control. Synchronizing different devices only by a single signal at the start of an experiment can lead to problems if these devices have different temporal drifts, and particularly if experiments are long in duration. Alternatively, initiating or measuring experimental events on a frame-by-frame basis using a PC for timing can introduce timing jitter due to the multitude of tasks that a PC must attend to at any given point in time. This jitter can have a significant impact depending on the study. For example, a recent calcium imaging study in the striatum finds additional neurological structure related to motor activity on very short timescales, but finds only velocity correlated with neural activity on longer timescales (Markowitz, et al., 2018). This suggests that with sufficient timing jitter, correlations and information on short time scales could be missed, yielding erroneous conclusions.

Temporal accuracy is not only important for behavioral data acquisition. In the trace conditioning experiment, for example, precisely timed stimuli are desired as well. Therefore, we characterized the accuracy and precision of digital and analog outputs while the Teensy interface continued to elicit repetitive camera digital pulses. First, we show that our Teensy interface accurately and precisely delivers a 9500 Hz tone using the built-in Audio library. This Audio library can also be used for sound synthesis, reading, and mixing, all at 44.1 kHz at stereo quality. This provides a tool for many experimental conditions, especially those needing sound as a stimulus. Secondly, we show that our Teensy interface accurately and precisely delivers a longer digital pulse that can drive “puffs” while simultaneously producing these repetitive camera pulses. Ultimately, the precisions of both our puff and sound output are comparable to expensive, available systems such as the Habitest Modular system in conjunction with Coulbourn Graphic State 4 software, which itself offers 1 ms precision (<http://www.coulbourn.com/v/vspfiles/assets/manuals/Graphic%20State%204%20Users%20Manual.pdf>) , making the Teensy a viable, inexpensive alternative that is also able to simultaneously capture imaging data using our simple software design.

Finally, a major advantage of the Teensy over other microcontrollers is its ability of generating a true analog signal. While Arduino devices can generate analog signal, they need extra devices such as resistors and capacitors to create an analog-like signal. The precision and flexibility of our Teensy 3.2 microcontroller interfaces make this a user-friendly, easily adaptable, accurate, and precise tool for to utilize in different experimental designs that benefit from synchronous image capture and behavioral control and data acquisition.

**Figures**

**Figure 1.** Diagrams of the two experimental device setups using Teensy interface. A, a floating, spherical treadmill setup for locomotion recording (A), **A** This experimental design consists of a Teensy 3.2 connected to two ADNS-9800 sensors via serial-peripheral interfaces, and a CMOS camera through a coaxial cable with SMA connectors. Every 50 milliseconds, a digital pulse was send to the CMOS camera to initiate an image frame capture, as well as to acquire motion data from both ADNS sensors and sending them to a PC via a USB. The PC initiates each experiment by sending serial data consisting of the length of the experiment and the digital output frequency to the Teensy. **B** A trace eye-blink conditioning setup. throughThe length and the number of experimental trials were specified in Matlab. In each trial, the Teensy generates? Or initiates? a 9500 Hz tone at a sampling rate of 44.1 kHz. Thetone stimuli are followed by an eye puff.

**Figure 2.** Detailed electrical wiring schematics for both the motor-control and tone-puff systems **A.** A schematic demonstrating the wiring connections between a Teensy 3.2, prop shield, and an external speaker. Dotted lines indicate solid connections. All connections between the Teensy 3.2 and prop shield were made using 14x1 double insulated pins, and the output to the speaker from the prop shield was made using regular wire and a coaxial cable. Some extraneous and unused pins on the Teensy and the prop shield were not included in this diagram. **B.** A schematic demonstrating the wiring of Teensy to two ADNS-9800 sensors via serial peripheral interface connections (SPIs). Solid dots at intersections between lines indicate connections. Some unused pins on the Teensy 3.2 were not included in this schematic.

**Figure 3.** Example recording using the motor control setup **A.** Part of a sample 10 minute recording session during which a head-fixed animal was allowed to run on the three-dimensional treadmill. Shown in **Figure 1A**. **B.** Times of digital pulses sent by the Teensy 3.2 as measured internally by the Teensy, vs times of the digital pulses as measured by an external device. Red indicates linear model, and in black are experimental data, down-sampled by a factor of 200. The linear model estimates a slope of 1.000028927 + .000000005 (t(11997)= 2.0381e+08, p < 0.001, R2=1; intercept = 0.000107 + 0.000002, t(11997) = 63.243, p < 0.001), indicating an excellent fit and very nearly a 1:1 correspondence of time stamps.

**Figure 4.** Example recording using the tone-puff setup **A.** Timing of the digital pulses as measured by the Teensy 3.2 in the tone/light-puff setup versus timing as measured by an external device. Linear model fit is shown in red, and in black are experimental data down-sampled by a factor of 200 for visualization. These measurements have a correspondence near 1:1 (R2=1, slope: 1.0000334 + 0 (to machine precision), t(14998)=infinite, p<0.001). **B.** Timing measured by the Teensy for (i) and by the TDT system for (ii-iv) over the course of fifty trials; (i) shows the latency between the theoretical onset of the tone and the measured timing of the tone as measured by the TDT device (mean=7.6 + 0.9 ms, range=2.9 ms); (ii) shows the consistency of the length of tone intervals across all trials (mean=700 + 1 ms, range=2.9 ms); (iii) shows the consistency of the latency of the puff stimulus, as measured by the TDT sytem (mean= -0.004 + 0.012 ms, range=0.04 ms); (iv) shows the consistency of the length of the puff across all trials (mean = 100.03+0.02 ms). (all + std).

**Tables**

Table 1. Specialty components necessary to build a motor output system

|  |  |  |  |
| --- | --- | --- | --- |
| **Part name** | **Website** | **Part number** | **Cost per unit** |
| Teensy 3.2 | https://www.pjrc.com/store/teensy32.html | TEENSY32 | $19.80 |
| ADNS-9800 sensors | https://www.tindie.com/products/jkicklighter/adns-9800-laser-motion-sensor/ | ADNS-9800 Laser Motion Sensor | $27.50 |

Table 2. Specialty components necessary to build a tone-puff system.

|  |  |  |  |
| --- | --- | --- | --- |
| **Part name** | **Website** | **Part number** | **Cost per unit** |
| Teensy 3.2 | https://www.pjrc.com/store/teensy32.html | TEENSY32 | $19.80 |
| 14x1 Double insulator pins | https://www.pjrc.com/store/header\_14x1\_d.html | HEADER\_14x1\_D | $0.85 |
| Prop shield | https://www.pjrc.com/store/prop\_shield.html | PROP\_SHIELD | $19.50 |

# References

Aranov, D., & Tank, D. W. (2014). Engagement of Neural Circuits Underlying 2D Spatial Navigation in a Rodent Virtual Reality System. *Neuron, 84*(2), 442-456.

Barbera, G., Liang, B., Zhang, L., Gerfen, C. R., Culurciello, E., Chen, R., . . . Lin, D.-T. (2016, October 5). Spatially Compact Neural Clusters in the Dorsal Striatum Encode Locomotion Relevant Information. *Neuron, 92*(1), 202-213.

Chen, X., & Li, H. (2017, December). ArControl: An Arduino-Based Comprehensive Behavioral Platform with Real-Time Performance. *Frontiers in Behavioral Neuroscience, 11*, 1-9.

D'Ausilio, A. (2012). Arduino: A Low-Cost Multipurpose Lab Equipment. *Behavior Research Methods, 44*(2), 305-313.

Dombeck, D., Khabbaz, A. N., Collman, F., Adelman, T. L., & Tank, D. W. (2007). Imaging Large-Scale Neural Activity with Cellular Resolution in Awake, Mobile Mice. *Neuron, 56*(1), 43-57.

Klaus, A., Martins, G. J., Paixao, V. B., Zhou, P., Paninski, L., & Costa, R. M. (2017). The Spatiotemporal Organization of the Striatum Encodes Action Space. *Neuron, 95*(5), 1171-1180.e7.

Markowitz, J. E., Gillis, W. F., Beron, C. C., Neufeld, S. Q., Robertson, K., Bhagat, N. D., . . . Datta, S. R. (2018). The Striatum Organizes 3D Behavior via Moment-to-Moment Action Selection. *Cell, 174*(1), 44-58.e17.

Micallef, A. H., Takahashi, N., Larkum, M. E., & Palmer, L. M. (2017, May). A Reward-Based Behavioral Platform to Measure Neural Activity during Head-Fixed Behavior. *Frontiers in Cellular Neuroscience*, 1-8.

Mohammed, A. I., Gritton, H. J., Tseng, H.-a., Bucklin, M. E., Yao, Z., & Han, X. (2016). An Integrative Approach for Analyzing Hundreds of Neurons in Task Performing Mice Using Wide-Field Calcium Imaging. *Scientific Reports, 6*, 20986.

Solari, N., Sviatkó, K., Laszlovsky, T., Hegedüs, P., & Hangya, B. (2018). Open Source Tools for Temporally Controlled Rodent Behavior Suitable for Electrophysiology and Optogenetic Manipulations. *Frontiers in Systems Neuroscience, 12*(May).

Yoav, A., Kim, J. J., Brinks, D., Lou, S., Wu, H., Mostajo-Radji, M. A., . . . Cohen, A. E. (2018). All-Optical Electrophysiology Reveals Brain-State Dependent Changes in Hippocampal Subthreshold Dynamics and Excitability. *bioRxiv*. doi:https://doi.org/10.1101/281618