behaviorMate: An *Intranet* of Things Approach for Adaptable Control of Behavioral and Navigation-Based Experiments

John C. Bowler[1,3,4+,†], George Zakka[1,3+,†], Hyun Choong Yong[1,3], Wenke Li[6], Bovey Rao[1,2,3], Zhenrui Liao[1,3], James B. Priestley[5], Attila Losonczy[1,3,†]

[1] Department of Neuroscience
[2] Doctoral Program in Neurobiology and Behavior
[3] Mortimer B. Zuckerman Mind Brain Behavior Institute,
Columbia University, New York, NY 10027 USA
[4] Department of Neurobiology University of Utah, Salt Lake City, UT 84112, USA
[5] Brain Mind Institute, École polytechnique fédérale de Lausanne
[6] Aquabyte, San Francisco, CA 94111
[+] indicates equal contribution

† Correspondence should be addressed to:
A.L.: al2856@columbia.edu
J.C.B.: jack.bowler@utah.edu
G.Z.: gz2333@columbia.edu

1 Abstract

Investigators conducting behavioral experiments often need precise control over the timing of the delivery of stimuli to subjects and to collect the precise times of the subsequent behavioral responses. Furthermore, investigators want fine-tuned control over how various multi-modal cues are presented. behaviorMate takes an "Intranet of Things" approach, using a networked system of hardware and software components for achieving these goals. The system outputs a file with integrated timestamp-event pairs that investigators can then format and process using their own analysis pipelines. We present an overview of the electronic components and GUI application that make up behaviorMate as well as mechanical designs for compatible experimental rigs to provide the reader with the ability to set up their own system. A wide variety of paradigms are supported, including goal-oriented learning, random foraging, and context switching. We demonstrate behaviorMate's utility and reliability with a range of use cases from several published studies and benchmark tests. Finally, we present experimental validation demonstrating different modalities of hippocampal place field studies. Both treadmill with burlap belt and virtual reality with running wheel paradigms were performed to confirm the efficacy and flexibility of the approach. Previous solutions rely on proprietary systems that may have large upfront costs or present frameworks that require customized software to be developed. behaviorMate uses open-source software and a flexible configuration system to mitigate both concerns. behaviorMate has a proven record for head-fixed imaging experiments and could be easily adopted for task control in a variety of experimental situations.

2 Introduction

In this work, we present *behaviorMate*, an open-source system of modular components that communicate with each other over a local area network (LAN), orchestrated by a custom Java application. We validate our approach through benchmarking and experimental results, demonstrating the ability of behaviorMate to control head-fixed navigational experiments both in a self-powered treadmill as well as in a fully visual virtual reality (VR) setup with a running wheel. This solution permits researchers to select the configuration of cues they wish to present at run-time, resulting in a

system that is capable of adapting quickly to incorporate changes to experimental protocols. The modular design makes it easy to adapt to shifting demands of research on the same behavioral setups while producing a consistent, easy-to-parse output file describing the experiment.

Proprietary systems such as National Instrument's Data Acquisition Systems (DAQ) and their accompanying software controller, LabView, typically have large upfront and licensing costs for each experimental rig their system is deployed on. behaviorMate exclusively uses open-source software and is simple to construct with parts that are significantly cheaper. The customized electronics may be ordered as printed circuit boards (PCB) and assembled by hand or purchased preassembled. The latter approach saves assembly time while adding significant expense; however, this can be mitigated by bulk ordering. The total cost of one of VR rig including 5 computer monitors, Android computers to render the visual scene, a frame for holding the monitors, and a light-weight running wheel is approximately $3700. The total cost for a belt-based treadmill system is around $1700. For two-photon imaging, additional components may be needed, such as a light-tight and laser resistant enclosure, which were not factored into the calculation. While other open-source and modular behavioral control systems have been developed which permit a variety of use cases (Akam et al. 2022, Saunders et al. 2022), behaviorMate does not require the user to write software code and uses compiled Arduino programs that do not incur the overhead of code interpreters to maximize performance.

Many experiments rely on using head-restrained animals to study the neuronal circuits underlying complex behaviors; such immobilization poses a fundamental challenge to behavioral research as it stymies animals' natural movement through their environments. In order to address this issue, a plethora of "treadmill" systems have been developed relying on belt-based systems (Royer et al. 2012, Lovett-Barron et al. 2014, Jordan et al. 2021), VR (Dombeck et al. 2010), or augmented reality (AR), where experimenters move "spatial" cues to create the illusion of locomotion without actual movement in physical space (Jayakumar et al. 2019). Belt-based systems involve placing an animal on a physical belt treadmill where, as the animal runs, physical cues affixed to the belt provide spatial information. VR- and AR-systems operate under closed loop control, where the environment responds to the animal's behavior (Dombeck & Reiser 2012). Normally, these cues are visually displayed on one or more computer monitors, but olfactory and auditory stimuli

have also been utilized at regular virtual distance intervals to enrich the experience with more salient cues (Radvansky & Dombeck 2018, Fischler-Ruiz et al. 2021). Furthermore, belt-based and VR/AR-based elements have been combined to enrich experiences and enable more complex behavioral tasks. At times, traditional "open-loop" stimulus may be required, such as timed cue presentations, allowing pre- and post- event neuronal activity to be examined. Increasingly, studies combine open- and closed-loop behavioral paradigms to identify stimulus-response profiles or behavioral state driven changes to stimulus responses. The proliferation of various experimental paradigms demands a high level of flexibility in experimental setups, so that methods can be integrated within single experiments or across lines of research to interrogate neuronal function.

Many of the spatial representations observed in freely moving animals are conserved in head-restrained animal setups (Aronov & Tank 2014, Dombeck et al. 2010). Observations within the medial temporal lobe, focused on all subregions of the hippocampus as well as the entorhinal cortex, have shown that prominent neural correlates of navigation behaviors exist in both the treadmill belt style systems as well as in the visual only VR systems (Aronov & Tank 2014, Dombeck et al. 2010, Royer et al. 2012, Danielson et al. 2016). In CA1 (Danielson et al. 2017, Zaremba et al. 2017) and CA3 (Terada et al. 2022), place cells, goal-related tuning, and other feature-selective tuning have been observed. Additionally, in visual VR systems, grid cell-like tuning has been observed in the medial entorhinal cortex (MEC), as was distance tuning (Aronov & Tank 2014, Heys et al. 2014). The increased control over reward distribution and cue manipulation that VR systems afford has been pivotal for recent findings relating to how cells in CA1 form place fields, uncovering novel mechanisms underlying synaptic plasticity (Gonzalez et al. 2023, Bittner et al. 2017, Priestley et al. 2022), demonstrating the power of these systems to test specific theoretical predictions.

3 Overview

behaviorMate is an integrated system composed of several sub-components that communicate on LAN using JSON-formatted packets transmitted via a standard Datagram packet protocol (UDP)

(Fig. 1). This approach allows for modularity and encapsulation of different concerns. Moreover, acknowledgment packets are used to ensure reliable message delivery to and from connected hardware. We dub the resulting system as an "*Intranet* of Things" approach, mimicking the Internet of Things systems commonly found in modern "smart homes," which communicate exclusively on a dedicated LAN using statically assigned internet protocol (IP) addresses. This allows for a central user interface (UI) or “real-time” computer to seamlessly send and receive messages from various accessory downstream devices that have dedicated simple functions. Importantly, the techniques we describe are possible implementations of the behaviorMate system. In our experimental setups, we use behaviorMate to combine *in vivo* head-fixed mouse navigation experiments with two-photon (2p) microscopy, focusing on one-dimensional (1D) spatial navigation tasks (Fig. 3E). Due to its modular design, however, behaviorMate can be easily reconfigured to provide closed-loop or open-loop control during a variety of behavioral paradigms (and has been used for non-navigation based studies; such as the head-fixed trace fear conditioning task described in Ahmed et al. has been implemented on this system).

In addition to running a PC-based UI, we combine Arduino open-source micro controllers with custom designed printed circuit boards (PCBs) to run the experiments (Fig. 1, 2D, E *also see* Supplementary Materials). The circuits include Ethernet adapters, allowing the Arduinos to communicate on the network, as well as connectors that interface them to sensors or actuators for controlling the experiments or for reporting the behavioral state back to the UI. We outline a general-purpose input/output (GPIO) circuit that can connect to a variety of sensors and reward valves generally using a transistor-transistor logic (TTL) pulse/square wave as well as I^2C and serial peripheral interface (SPI) protocols (Fig. 2D), a position tracking circuit (Fig. 2E) that is dedicated mainly to reading updates of the animal’s location from a rotary encoder, and a VR controller setup which can be used to provide a more immersive visual display to animals during the behaviors. Importantly these individual components can be swapped out or added to with increased experimental demands. In most cases the provided software and hardware are sufficient to meet users needs, however, we do also point to certain targets for easy expansion which would require editing the software provided to interface with specialized hardware (for example adding a novel

SPI based extension to the GPIO circuit). In general, without editing software the user has two options for controlling hardware: using one of the digital/analog port on the GPIO board or over network via an Ethernet adapter (but this could requiring following the same JSON API as existing behaviorMate components). For additional integration of software components or testing, the UI can be configured to send messages to the PC's `localhost` address (i.e. 127.0.0.1 on most PCs). To speed up development, we created virtual implementations of the electronics in Python code to test UI changes before testing them on a physical system. Most modern programming languages, including Python and Matlab, have built-in support for JSON processing and UDP networking making component and new feature testing easy with the help of utility scripts that can be written in users' preferred environments (*see* Supplementary Materials for references to example Python code).

Lastly, since our main focus is on 1D navigational tasks for two-photon imaging, we provide two designs for physical rigs, which hold animals fixed in place under a microscope. However, they could also be adapted to any other methods for *in-vivo* recording. One system is a self-powered, belt-based treadmill system, wherein animals are placed on a fabric belt that they pull beneath them to advance their virtual position. Locations and contexts can be distinguished in this setup by having various fabrics and physical cues. Alternatively, the other system provides navigational contexts through a series of computer displays for a purely visual experience, but allowing for additional flexibility of within-task manipulations to context and cues. We describe the details of both of these systems and, additionally, note that with the following methods, it is possible to combine aspects from both belt-based and VR systems to meet novel experimental designs. To clarify, for the remainder of the paper, "treadmill" will refer to these belt-based systems while "VR" will refer to the visual-based system in which animals moving a plastic running wheel.

4 The User Interface

The central hub for running experiments is the behaviorMate user interface (UI) (Fig. 2A). The UI collects input from various sensors, assembling information about virtual position and behavior state data asynchronously. The UI performs three main functions: (1) accepting user input and displaying the current state of experiments to ensure proper functioning, (2) writing hardware information in a time-stamped streaming text file which can be processed later for analysis of the experiment, and (3) serving as the central communications hub for sending and receiving UDP-formatted messages to the downstream components. The system is event-driven such that the hardware components send messages to the UI when the state changes rather than the UI continuously polling. There are currently two versions of the UI. The first is a long-term support (LTS) release that evolved alongside the experimental systems and has been used for data collection for numerous published studies; the second is a streamlined beta version that has been rewritten to take advantage of the JavaFX visual development toolbox. Java was chosen for the UI because it is cross-platform in that it can be run on Windows, Linux, and Mac, and supports standard networking protocols. For consistency, interchangeability, and space optimization, we run the UI on an Intel(TM) NUC mini PC with a 2.4GHz base-frequency (up to 4.2GHz based on processor load) 4-core i5-1135G7 processor, 16GB DDR4-3200MHz RAM, and a 512GB NVMe M.2 SSD hard drive running Window 10 Professional Edition, although many other configurations are supported. Notably, due to the system's modular architecture, different UIs could be implemented in any programming language and swapped in without impacting the rest of the system. The only requirement for interchangeability of the UI is to implement sending and receiving of the same JSON formatted UDP messages packets to the downstream hardware components (*see* Supplementary Materials for arduino firmware which includes additional detail of the JSON messaging protocol).

The UI executes in three distinct phases during run-time. First a setup routine is executed, then the program enters the main event loop, and finally a termination routine is executed when an experiment is completed (Figure 2B). The main event loop begins executing as soon as the UI is loaded, but additional steps for checking and logging the contexts' state are added during the experiment. Devices still receive input and deliver output to the UI even when an experiment is

not currently running, allowing experimenters to confirm that everything is working before starting. Experiments using behaviorMate are configured by user-generated JSON files with key-value pairs specified in the documentation. These settings files can be manually generated in a text editor or programmatically generated using a script (*see also* Supplementary Materials).

The main structure of the application logic is an event loop that iterates over enabled *contexts*. Within behaviorMate, a context is grouping of one or more stimuli that get activated concurrently. For many experiments it is desirable to have multiple contexts that are triggered at various locations and times in order to construct distinct or novel environments. Contexts define presentation schemes for both simple stimuli such as an LED lighting up and water reward delivery, as well as more complex stimuli like flashing LEDs or even an entire visual VR scene. A context can be applied globally or to one or several locations; a list of locations and rules governing an individual context's activation is referred to as a Context List. When a Context List is active, the devices assigned to it will be triggered. For example, a user could configure an LED which blinks at several designated locations along the track. Context Lists are implemented in the UI using an Object-Oriented approach that streamlines implementing logic for governing the presentation and control of novel cue types. The main rules governing how a Context List works are implemented in a `check` method which is passed the current state of the experiment as input arguments and returns a Boolean variable as `true` or `false` corresponding to whether a Context List should be active or not. In this way novel Context List types may be added to the program easily through inheritance of a base class and overriding this one method. Additionally, Context Lists are instantiated using the "Factory Method" pattern (Gamma et al. 1994) and appended to a dynamic list. This pattern allows for adding complex functionality while minimizing the impact on performance and increasing code reuse and modularity. Existing Context List types can be modified by applying "decorators" (Gamma et al. 1994), resulting in *composable* behavioral paradigms, meaning that novel experiments can be implemented by nesting Context Lists within one or more existing decorators, without requiring any software modification. Settings files following JSON syntax specify the configuration as well as the context lists and decorators. For example, a particular decorator can be configured to cause a context to only be active on even laps while a second decorator may restrict the context from

being active until after a certain amount of time has passed. Composing these two rules means it is possible to trigger the contexts on even laps, only after 5 minutes have passed since starting the experiment. However, if the available set of decorators is not enough to implement the required task logic, some modifications to the source code may be necessary. These modifications, in most cases, would be very simple and only a basic understanding of object-oriented programming is required. A case where this might be needed would be performing a novel customized real-time analysis on behavior data and activating a stimulus based on the result. The JavaFX version of the behaviorMate UI additionally supports a plugin architecture which will further simplify the process of adding custom decorators and context list rules in the future.

5 Behavior Tracking & Control

5.1 GPIO/Behavior Controller

The primary GPIO circuit we implement, referred to as the Behavior Controller, is composed of an Arduino attached to a custom circuit board (Fig. 2D). The PCB handles voltage regulation and provides the necessary connectors to operate downstream devices. The Arduino program distributed with behaviorMate controls all connected devices and communication between them and the UI. The program wraps “action” messages with “routes” (Fig. 2C). This pattern of routing messages simplifies debugging since it is clear which classes of the Arduino code received the messages and where replies were generated. The messages are `JSON` formatted text, so they are human-readable and can be simulated within the behaviorMate PC for testing and debugging purposes (*also see* Supplementary Materials, for links to Arduino firmware and documentation on the JSON messaging protocol). The PCB has various connectors and other components that makes it easier to connect arbitrary hardware, including sensors and actuators, to the Arduino. Sensors and actuators can be connected to the controller using one of the 13 digital or 5 analog input/output connectors. Moreover, the controller provides an Ethernet adapter for the Arduino to permit LAN communication.

The Behavior Controller has several important functions; it receives signals from the UI to activate

rewards and cues such as odor valves, LEDs, and tone generators and passes input from sensors to the PC. In our setup for 2p imaging, the Behavior Controller also sends a synchronizing signal to the microscope to initiate imaging after an experiment is started through the behaviorMate UI. Some microscopes can be configured to begin imaging by a TTL synchronizing trigger. Alternatively, microscopes might send a trigger indicating that recording has commenced. Moreover, some setups can periodically send back synchronization signals to ensure alignment during long recordings that may be affected by clock drift. The Behavior Controller can both receive and send triggers for the purpose of synchronization which will be timestamped and logged by the UI. By default, this signal is a TTL pulse (3.3 V in our implementation, Fig. 2D *also see* Supplementary Materials) and can also be configured to activate other types of recording devices such as video cameras or electrophysiology interfaces. However it is important to note that the sampling rate of an Arduino Due (used in Behavior Controller) is incapable of the kilohertz sampling rate of typical electrophysiology devices. In these cases it might be necessary to send a periodic synchronization TTL from the behavior controller to the electrophysiology hardware to ensure proper timing of behavior and neural data. behaviorMate is compatible with a variety of alignment and synchronization schemes and it is left up to the users to implement the solution most appropriate for their experimental setups.

5.2 Position Controller

The position controller's function is to detect animal movement and report it to behaviorMate. It is composed of an Arduino Nano attached to a custom circuit which has an onboard quadrature decoder, 16-bit counter, and connectors for attaching the rotary encoder and lap reset circuit. For either treadmill or running wheel-based setups (i.e. for a VR setup), a quadrature rotary encoder device is coupled to the shaft of a wheel that turns as the animal runs. The turning of the rotary encoder generates necessary signals, or "ticks", which are passed to the quadrature decoder to calculate the instantaneous velocity of the animals. These quadrature ticks are decoded and counted in the counter. The custom circuit (Fig. 2E) uses a 10MHz oscillator which is capable of counting high resolution encoders (>4096 ticks/turn) at greater than 1K RPM. The onboard counter enables the tick counting to be decoupled from the speed of the on-board Arduino. Consequently,

the Arduino can check the counter at a much slower rate (100Hz in our setup, but can be as slow as 5Hz) without losing count of the ticks. The Arduino is only responsible for checking the counter and generating a `JSON` formatted text string to transmit to the behavior PC via Ethernet, and only transmits when there is movement. Significantly, this eliminates polling from the behavior PC and allows the communication between the Arduino and the PC to be one way instead of bidirectional. The behavior PC then translates the counts into a meaningful measurement of the linear distance traveled by the animal on the belt or running wheel.

6 Behavioral Apparatus

6.1 Treadmill system

Self-powered treadmill systems have been extensively used in the study of navigational behaviors (Royer et al. 2012, Tuncdemir et al. 2022, Lovett-Barron et al. 2014, Grienberger & Magee 2022, Geiller et al. 2017). The following section outlines a protocol for the construction of a behaviorMate compatible mouse treadmill setup that has been used for head-fixed imaging (Kaufman et al. 2020*a*, Terada et al. 2022, Rolotti et al. 2022b). In combination with the software and circuits described above, the treadmill system provides a flexible platform for implementing novel experimental paradigms.

6.1.1 Frame

The treadmill system requires a frame to support it and maintain wheel alignment, so we provide designs for a frame made of extruded aluminum rails (Fig. 3A). The rails support wheels on either end for a fabric "belt" that can be moved by a mouse. In addition to the wheels, we added two wheel-rollers that support the belt near the animal's fixed position (Fig. 3E). This design does not have a platform directly beneath the mouse, allowing the belt to absorb vibrations, which reduces motion artifacts and allows for a stable field of view during *in vivo* imaging experiments. Treadmill belts are constructed from 2-inch wide fabric ribbon and may be embellished by linking multiple

fabrics or adding various physical cues such as foam balls, sequins, or hook-and-loop fasteners for texture (Fig. 3F). These cues help the animal to orient itself and are useful for information encoding. In this setup, the animal is head-fixed beneath a microscope, but is able to move the belt and attached physical cues which mimics navigation while still allowing investigators to perform imaging of cellular and even sub-cellular regions of interest. To further optimize the treadmill for imaging, a two-goniometer device placed underneath the belt is used to adjust the angle of the animal's head along the roll (side-to-side) and pitch (forward and backward) axes. This setup allows the investigator to level the field of view, ensuring the neuronal plane of interest is perpendicular to the optical axis of the microscope, thus increasing the number of neurons that can be imaged and improving image clarity. However, this may not be needed in setups where the objective itself can be tilted; if the objective can be tilted along two-axes, no goniometers would be needed, whereas if the objective can only tilt along one axis, a single goniometer could be employed. Finally, the frame has a sliding mechanism to adjust the distance between the wheels and accommodate various sized belts to ensure proper tension. Proper belt tension ensures the animal can move easily and reduces the chance of belt slippage along the wheels.

6.1.2 Electronics and Peripherals

Interfacing between the treadmill and behaviorMate requires the previously described electronics. Any number of optional peripheral devices such as LEDs, odor release systems, etc., may be added. A rotary encoder is attached to one of the two wheel shafts and connected to the position tracking circuit to log the animal's movement. The behaviorMate UI handles the conversion of the rotary encoder's spinning to the linear distance traveled. It also maintains the animal's current lap number. An additional "lap-reset" feature prevents drift from accumulating between the digital tracking and physical treadmill belt. We provide designs for an optical lap-reset circuit for determining when a complete revolution of the belt has been made (Fig. 3A). Setting up a lap-reset circuit additionally permits a calibration function that can be temporarily enabled prior to running an experiment to adjust the scaling between rotary encoder "ticks" and distance moved along the treadmill belt.

Generally position tracking drift is small (Fig. 3C) but can occur (especially with more “slippery fabrics” or if belt tension is set too low). It is important to accurately specify the track length and calibrate the position scale factor to tightly couple the PC and the electronics. If the system is not set up properly or the belt is not sufficiently tight, the belt may slip, leading to inaccurate position readings. Therefore, we provide a benchmark of the position tracking performance under normal conditions. To compute the linear distance traveled, a small LED was attached to the top of the belt and the entire treadmill system was moved to a completely dark room with an IR camera pointed directly at the belt. The belt was moved by hand and upon each complete revolution, the LED would pass within the field of view of the camera. OpenCV (Bradski 2000) was used to extract the lap completion times from the recorded video. These were taken to be the “ground truth” lap completion times. The quantities of interest were the time and position differences between the LED and those tracked by the behaviorMate UI. The mean of the magnitude of the differences between lap times reported and the “ground truth” was 0.102 seconds (Fig. 3B, D), while the mean difference between position upon lap completion was 13 mm and did not exceed 23 mm (Fig. 3C). Given that the mean and max differences were a fraction of the average size of a subject mouse and significantly smaller than the typical size of reward zones, these metrics were deemed more than acceptable. Enabling the lap-reset feature on self-powered treadmill prevents accumulation of these errors, ensuring that position, as tracked by the UI, is aligned with the fabric belt and cue presentations are aligned with the positions tracked by the UI.

The UI synchronizes any number of additional sensory cues such as LEDs, tones, and odor release systems, which can be present at pre-defined times or locations along the belt (Fig. 3G). For example, a liquid reward can be coupled to a specific location on the treadmill belt that is either *operant* (delivered only if the animal licks within a "reward zone") or *non-operant* (delivered immediately when the animal reaches a "reward zone"). These "reward zones" are defined by a location and radius, so there is user-specified tolerance for the areas animals can receive operant rewards. The rules governing the rewards are fully customizable and specified in the settings file for the UI.

6.1.3 Use in Past Studies

The physical design of the treadmill was inspired by Royer et al. and influenced by several other hippocampus studies focusing on behavioral navigation tasks (Kaifosh et al. 2013, Lovett-Barron et al. 2014, Zaremba et al. 2017). Our design is flexible and has been validated by the myriad experimental paradigms that have been successfully implemented our lab (Tuncdemir et al. 2023, Vancura et al. 2023, O'Hare et al. 2022, Rolotti et al. 2022a,b, Terada et al. 2022, Geiller et al. 2022, Blockus et al. 2021, Grosmark et al. 2021, Geiller et al. 2020, Kaufman et al. 2020*a*, Turi et al. 2019, Zaremba et al. 2017, Danielson et al. 2017, 2016) as well as set up and run by collaborators (Tuncdemir et al. 2022, Dudok et al. 2021a,b). Multi-modal cues can be incorporated across several days of experiments to probe different aspects of navigation and associative learning (Fig. 3G). For example, the treadmill has been used to test goal-oriented learning, where goal locations are "hidden" and animals must report their understanding of the correct location by licking at the reward spout in order to trigger water delivery (Zaremba et al. 2017, Danielson et al. 2016). During these tasks, novel cues can be presented by introducing new treadmill belts as well as novel non-spatial cues such as background tones, blinking LEDs, or odor stimuli. Moving the reward location forces animals to learn updates to task rules mid-experiment and has been used to investigate deficits in mouse models of psychiatric disorder (Zaremba et al. 2017) as well as neural circuit analysis of the mechanisms behind reward learning (Kaufman et al. 2020*a*) and the development of signals underpinning spatial navigation (Rolotti et al. 2022b, Terada et al. 2022, Danielson et al. 2016). Cues can be tied to time, location, or both. Switching between a time-dependent presentation and a location-dependent one provides a powerful tool for assessing how the same cue may be processed differently based on the current context, location, or task (Terada et al. 2022, Tuncdemir et al. 2022). In addition to cue presentation, the setup has also been used to trigger optogenetic stimulation (Kaufman et al. 2020*b*, Geiller et al. 2022, O'Hare et al. 2022, Rolotti et al. 2022b); this coupling of navigational cues provided by the treadmill belt with stimulation in hippocampal region CA1 was used for the first successful optical induction of place fields (Rolotti et al. 2022b).

6.2 VR System

The use of Virtual Reality (VR) cues can add significant flexibility to behavioral studies and has thus become increasingly popular in neuroscience. Although different VR implementations have been described, for this manuscript, VR is defined as closed-loop control from the behaviorMate UI that integrates position updates virtually and advances cues past the animal to create the illusion of movement, unlike a belt which physically moves past the animal. Various systems for accomplishing this have been proposed and implemented previously (Dombeck et al. 2010, Harvey et al. 2012, Sheffield et al. 2017, Hainmueller & Bartos 2018, Campbell et al. 2018, Arriaga & Han 2019, 2017, Dudok et al. 2021). Mice can learn to use VR cues as landmarks for traversing environments to specific locations. Neuronal circuits thought to be involved in navigation in freely moving mice have been shown to be similarly engaged during the described VR tasks (Dombeck & Reiser 2012, Dombeck et al. 2010, Aronov & Tank 2014). behaviorMate represents a novel contribution in that it incorporates visual VR displays into its architecture for 1D navigation and is designed to adapt seamlessly to novel arrangements of VR displays with minimal computational load for each additional display (Priestley et al. 2022, Bowler & Losonczy 2023). This is achieved by rendering the virtual environments on separate android devices (one per display) with a separate application (VRMate). Since the same behaviorMate UI is used for treadmill and VR experiments, all of the experimental paradigms described in the treadmill section can be implemented in the VR system with minimal changes to the settings file. Additionally, VR permits rapid visual context switches and the addition of novel virtual objects in a familiar scene. Furthermore, “hybrid” setups are also supported in which visual displays can be added to physical treadmills in order to combine both tactile and visual cues.

6.2.1 VRMate

VRMate is the companion program to behaviorMate that runs the visual VR interface to display cues to animals as they move in virtual space. VRMate is a program developed using the Unity(TM) game engine and written in C#; it listens for incoming UDP packets carrying position updates from the behaviorMate UI. In our implementation, each visual display is connected to an ODROID C4 which

is a single-board computer equipped with an Amlogic S905X3 Quad-Core Cortex-A55 (2.016GHz) ARMv8-A Processor, 4 GiB of DDR4 RAM, Mali-G31 MP2 GPU, and 1 LAN port, all mounted on a single PCB and running the Android operating system. Each display-ODROID pair runs an instance of the VRMate program, which means scene rendering is done independently by dedicated hardware for each monitor. This allows for scalability to any number of additional displays without compromising frame rate or resolution. Prior to starting an experiment, behaviorMate supplies each instance of VRMate with all the information it needs to correctly display the virtual environment from a particular point of view (Fig. 4F). Notably, Unity is platform independent, so VRMate can be built for Windows, OSX, and other systems. Therefore, it is also not strictly necessary to use dedicated Android devices, and it is possible to run behaviorMate and VRMate on the same PC. Various setups can be configured to meet individual researchers' needs.

Specifically for researchers conducting imaging experiments, Unity supports the implementation of custom "shader programs" which can be enabled through the UI. Implementing custom shaders allows you to edit color-pixel values in the visual scenes just prior to when they are rendered. For example, this means it is possible to remove a particular color emission from the scenes or display with just a single color (Fig. 4F). This is particularly useful for experiments where green or red fluorescent proteins are used to indicate neuronal activity. In this case, rendering scenes with the green or red component of each pixel set to 0 minimizes the risk of artifacts in the recorded data.

Instead of having separate applications for scene rendering and task logic using VRMate and behaviorMate, respectively, some systems may involve a single application that implements task logic, scene rendering, and communication with electronic controllers such as an Arduino. This approach may be appropriate in certain use cases, however, the recommended behaviorMate architecture has several important advantages. Firstly, by rendering each viewing angle of a scene on a dedicated device, performance is improved by splitting the computational costs across several inexpensive devices rather than requiring specialized or expensive graphics cards in order to run. Moreover, the overall system becomes more modular and easier to test and debug especially when designing multiple different behavioral paradigms. behaviorMate tells VRMate what scene to display,

the viewing angle, and the current position while VRMate handles the rest–in this way separating the task controller logic from the VR display modules. Thirdly, implementing task logic in Unity would require understanding Object-Oriented Programming and C# (or learning some other 3D development platform) which is not always accessible to researchers that are typically more familiar with scripting in Python and Matlab. behaviorMate does not necessitate programming because it uses configuration files that can be modified in a standard text editor. Finally, some experiments may not require visual cues at all, in which case a system where the VR component is optional and not tightly integrated with the application is desirable. Of course, in the current implementation VRMate could still be configured to run on a single PC with the display stretched across multiple monitors. This would mitigate any concerns about potential frame rate offsets between screens and simplify the network setup. However, our testing shows the display to be smooth and continuous as VRMate is a lightweight application and the scenes are not graphically intensive and runs at a high frame rate on the suggested hardware.

6.2.2 Frame and Display Setup

Our standard setup for immersive mouse VR experiments contains 5 monitors surrounding the animal. We supply plans for a VR frame to construct this (Fig. 4A, *also see* Supplementary Materials). The VR frame is designed with extruded aluminum rails and holds the displays at the proper angle and distance from the animal. The goal is to completely encompass the animal's visual field with the virtual scene, so the displays are placed in an octagonal setup with the animal at the center. The angle between each display is 135 degrees and the distance between the mouse and the center display is 15 inches. Since mice primarily have a visual field-oriented overhead (Dräger & Olsen 1980), the mouse is positioned 5 inches above the bottom edge of each monitor in order to align the visual stimulus to the animal's field of view.

In cases when the standard 5 monitor setup will not fit due to space constraints, we use a wireless, tablet-based setup (Fig. S1A). When VRMate is built for Android OS, it can be installed on any Android tablet. Our tablet system consists of 5 "Fire HD 10" (2019, 9th Gen), 10.1 inch tablets with a screen resolution of 1920 x 1200 IPS while running Android 9. Additional specs include 2

GB RAM, 4xARM Cortex-A73 (2.0 GHZ) and 4xARM Cortex-A53 (2.0 GHz) processors, and an ARM Mali-G72 MP3 GPU. Tablets may be connected to the behaviorMate intranet using a wireless router. The PC running behaviorMate connects through this router to interface with the displays. Each device on the network is assigned a unique static IP address by configuring the IP tables on the Wi-Fi-enabled router to bind each tablet's IP address to its MAC address. A low-latency router with multiple antennae is recommended and should be placed as close as possible to the tablet displays.

Typical VR setups position animals on top of a running wheel which is less bulky and mechanically simpler than the fabric belt treadmill system. The running wheel design used in our setup, as described previously by Warren et al., is lightweight and optimized to have low rotational momentum. This provides animals with a natural feeling of running while minimizing the effort required to traverse the virtual environments. Mice are head-fixed on top of the wheel using custom head restraining bars and a matching holder using a dove-tail and clamping system. The implant holders (Fig. S1B) are two L-shaped components machined from solid steel. They have two M6 holes for being attached to pillar posts of aluminum rails. In addition, the implant holders each have a channel which allows the clamps to slide and accommodate head-bars of different lengths. Both the clamps and clamp holders have thumb screws not shown in the figure. The screw for the clamps tightly grips the head-bar and thus minimizes motion artifacts in imaging data. The screw for the clamp holders fixes the clamps' position and reduces motion artifacts. The head restraining bars for mice presented with visual cues (Fig. 4G) are designed to assist with blocking light that may otherwise be detected during imaging. These can be 3D printed from titanium and reused between animals if properly cleaned and sterilized.

In some cases, a modified running wheel design with goniometers is required to tilt both the implant holders and the entire wheel along the roll and pitch axes (Fig. 4B). For head-fixed two-photon imaging, this is used to align the imaging plane to maximize the size of the field of view or, as needed, level the glass coverslip placed over the brain (which minimizes the amount of the material the beam must travel through and maximizes image clarity). For this modified design, aluminum

rails are arranged in a "U" shape and then placed on top of two goniometers. The implant holders sit on top of these rails, and the wheel is suspended below the rails, offset from the goniometers. The entire assembly sits on sliding mounts connected to pillar posts so the elevation of the animal's head can be adjusted.

6.2.3 Example setup

A simple VR system could have 2 displays in an arrow configuration directly in front of a mouse which is head fixed and mounted on top of a running wheel. Attached to the back of each monitor would be a small Android computer (ODROID) connected via an HDMI cable. The left screen could be given an IP address of 192.168.1.141. The right machine will be assigned 192.168.1.142. Static IP assignment is done through the Android devices' settings menu. Arduinos running the Behavior and Position Controllers are given the IPs of 192.168.1.101 and 192.168.1.102 respectively (which is the default in our supplied code (*see Supplementary Materials*). Finally, a PC running the UI is assigned an IP of 192.168.1.100. We use standard IP address and ports for each of the displays and electronics; however, the IPs that the behaviorMate UI will use are specified in the settings file and determined at run-time (the Arduino IP assignments are made when the devices are loaded). Each device is also assigned a port which can be selected arbitrarily and can be changed as needed. The only requirement is that all devices are on the same subnet. In this example setup, all devices share the same network prefix of 192.168.1.x so they are part of the same subnet and will be able to communicate. Given that all components are connected to the same local network, usually via an unmanaged switch, packets will arrive to their intended recipient.

When the animal runs, turning the wheel and rotary encoder, the behaviorMate UI will receive and integrate movement updates from the Position Controller. Once the experiment is started from the UI, the PC will evaluate which contexts are active and send position updates to both screens simultaneously. The VRMate instances will receive these packets and render their view of the same scene independently. Latency between the UI and the computers running VRMate is fairly low (Fig. 4C) so this does not present an issue. As the animal runs, it enters a reward

zone. behaviorMate will send a message to the Behavior Controller triggering the "reward context", causing a reward valve to release a drop of water or sucrose reward. As the mouse runs, the visual scene updates accordingly. Once the mouse travels a linear distance equal to the "track_length" parameter specified in the settings file, behaviorMate advances the lap count and resets the position to 0. Between-lap "time outs" or dark periods may also be configured. Furthermore, the user may choose to run a "context-switch" experiment: this type of experiment might involve one scene being displayed on odd laps and another on even laps. If a scene change is necessary, between laps, behaviorMate will send a message to both running instances of VRMate that specify the switch to the alternate scene.

6.2.4 VR System Performance

One key concern with closed-loop control VR experimental setups is the latency between when the animal moves the physical wheel and when the displays update position in the virtual environment. We therefore performed a benchmarking test on our standard 5 monitors VR/running wheel setup (with scenes rendered by ODROIDs connected to the UI via an unmanaged Ethernet switch). An impulse was applied to the running wheel, and the time taken for the screens to update was observed. The screen delay was measured using a high-speed 150 fps camera. Latency was sampled 20 times and found to have a median value of 0.067 seconds, with no sample exceeding 0.087 seconds (Fig. 4C). Since mouse reaction times are approximately 0.1 seconds (Dombeck & Reiser 2012, Mauk & Buonomano 2004), VR position updates using this system are nearly instantaneous to the mouse. Moreover, because position updates are constantly being sent by behaviorMate to VRMate and VRMate is immediately updating the scene according to this position, the most the scene can become out of sync with the mouse's position is proportional to the maximum latency multiplied by the running speed of the mouse. Such a degree of asynchrony is almost always negligible.

As with the fabric treadmill, another characteristic that should be considered for the running wheel is the accuracy of the position tracking system. Accuracy refers to how closely the mouse's

virtual position recorded by behaviorMate matches the true linear distance it has run on the wheel. To find the "true" linear distance traveled, a small dim LED was attached to the side of the running wheel, and the whole setup was placed in a dark enclosure, so the only significant source of light was the LED. The wheel was advanced by hand, and using OpenCV (Bradski 2000), the total distance traversed by the LED was extracted. Concurrently, behaviorMate was detecting the current position using the attached rotary encoder. Graphs of position vs. time from the data collected using computer-vision and the data recorded by behaviorMate were overlaid. From 4D, it is clear that there is little discrepancy between the two positions. Across all time points, the mean difference between the true and recorded position was 4.1 mm.

6.2.5 Use in Past Experiments

While the introduction of a purely visual VR option for behaviorMate represents one of the newest updates to the setup, studies have already been published that specifically leverage this capability, primarily to examine the effect of rapid within-session context changes (Priestley et al. 2022, Bowler & Losonczy 2023). VR experiments have the ability to transport animals to completely novel scenes which has been useful for investigating the mechanisms of synaptic plasticity, since it is possible to capture the animals first exposure to an environment in a controlled way (Priestley et al. 2022). Since the visual VR setup is integrated into behaviorMate, it is also possible to test how animals respond to changes in the rules governing reward delivery and visual scene presentation. The system further allows for simultaneous delivery of multi-modal stimuli simultaneously such as audio and visual. Thus, our system comprises an adaptable framework for probing the function of navigational circuits in the brain and the relationship between goal and context representations (Bowler & Losonczy 2023).

7 Experimental Validation

To investigate if spatially tuned 'place cells' (O'Keefe & Dostrovsky 1971) with similar properties emerge in both the belt treadmill system (TM) and the VR running wheel system, we performed 2p

imaging in CA1 of the hippocampus while animals explored in either VR or TM. Mice were injected with GCaMP8m (AAV1-syn-GCaMP8m-WPRE, Addgene) and implanted with a glass window to have optical access to CA1 (Fig. 5). Once mice recovered from surgeries, they were trained to do a spatial navigation task in either VR or TM. After the mice were fully trained, 2p imaging was performed during navigation. Calcium signals were processed with Suite2p for motion correction, cell segmentation, and fluorescent signal and neuropil extraction (Pachitariu et al. 2017). Extracted raw fluorescent signals were corrected for neuropil contamination, and resulting ΔF/F was used to estimate spike events using OASIS (Friedrich et al. 2017). A median filter was applied to binarize spike amplitudes into bins of position, and all subsequent analyses used this binarized train as an estimation of each ROIs activity. We recorded on average 1533$\pm$481 and 1057$\pm$169 CA1 neurons in VR and TM, respectively. Although VR had significantly fewer CA1 neurons being classified as place cells compared to TM (VR: 34.9% (3253/9195), TM: 44.9% (6101/13454), two sample z-test, z=14.96, p=1.27e-50), place cells tiled the entire track in both VR and TM (Fig. 5B, S2). In all subsequent analyses, we used CA1 neurons identified as a place cell unless otherwise stated. To assess differences in the calcium signal of place cells found in context-environment pairs, we first compared the ΔF/F amplitude and frequency of individual ROIs identified as place cells. Only ΔF/F events that were 3 standard deviations above the mean ΔF/F were included in these analyses. To examine the effect of environment (VR vs. TM) and context (familiar (F) vs. novel (N)) on ΔF/F amplitude and frequency, we conducted two-way repeated measures of ANOVA with both environment and context as a within-subject factors. There was no significant effect of environment or context on mean ΔF/F amplitude (environment: F(1,5)=1.9387, p=0.222, context: F(1,5)=0.0037, p=0.9537, Fig. 5C, ST1) or on mean ΔF/F frequency (environment: F(1,5)=4.37, p=0.091, context: F(1,5)=0.0789, p=0.7901, Fig. 5D, ST1). These results suggest that calcium transient properties of place cells observed in both environments and contexts were similar. Next, we quantified individual place cells' specificity, sensitivity, and spatial information to identify any differences in place cell firing properties between context-environment pairs. As described above, two-way repeated measures of ANOVA were used with both context and environment as within-subject factors. The two-way ANOVA revealed a significant effect of environment on sensitivity (environment: F(1,5)=9.85, p=0.0257, context: F(1,5)=3.34, p=0.13, Fig. 5E, ST1) and specificity (environment: F(1,5)=53.32, p=0.0008, context: F(1,5)=2.55, p=0.17, Fig. 5F, ST1), suggesting that

place cells in TM have higher trial-by-trial in-field firing compared to those from VR. However, we did not observe any significant effect of environment or context on spatial information (Skaggs & Mcnaughton 1998) (environment: $F(1,5)=0.74$, $p=0.43$, context: $F(1,5)=3.79$, $p=0.11$, Fig. 5G, ST1). These results are consistent with previous findings that place cells in VR have reduced trial-by-trial reliability (Ravassard et al. 2013, Aghajan et al. 2015), but still encode spatial information (Dombeck et al. 2010, Aronov & Tank 2014, Hainmueller & Bartos 2018, Priestley et al. 2022). The observed differences in the properties of VR and TM place cells may be the result of the absence of proximal visual cues and tactile cues (which are speculated to be more salient than visual cues) during VR tasks (Knierim & Rao 2003, Renaudineau et al. 2007, Sofroniew et al. 2014). Next, we wanted to investigate if place cells remap in both VR and TM environments when mice are exposed to a novel context (Muller & Kubie 1987, Leutgeb et al. 2005). As expected, a subset of place cells changed their place tuning properties when exposed to novel contexts (Fig. 6A,B). Moreover, we did not observe differences in the between context rate map correlation between VR and TM, suggesting that remapping happens in both environment types. To further quantify the extent of remapping in each environment, we built a naive Bayes decoder to decode position based on population activity of CA1 neurons. When the decoder was trained with population activity from familiar trials and tested with population activity from a held-out familiar trial, the absolute decoding error was significantly below chance ($p<0.05$) in both VR and TM. However, when the decoder trained with familiar trials was tested to decode position in a novel trial, the absolute decoding error was at the chance level (Fig. 6C,D) in both VR and TM. These suggest that place cells remap in both VR and TM when exposed to novel context.

8 Discussion

behaviorMate is, most generally, a solution for controlling and collecting feedback from devices in behavioral experiments. The behaviorMate methodology is built around an “Intranet of Things” approach, expanding to incorporate additional components to meet experimental demands but also compatible with sparser setups to ease complexity when not needed. It has typically been used concurrently with neuronal imaging, since the time-stamped output of behavior events is relatively simple to align with imaging data, providing experimenters with the ability to correlate neuronal

events with behavior. In addition, behaviorMate is designed to be flexible, expanding to incorporate additional components as specified in the settings while making minimal assumptions. Experimenters may incorporate other techniques such as imaging and electrophysiology recordings or just run behavior. For experimental paradigms involving any combination of displays, one-dimensional movement (wheels, treadmills, etc.), sensors (LEDs, Capacitance lick sensors, etc.), and actuators (odor release systems, solenoid valves, etc.), this is a cost-effective and reliable solution.

The main modules of behaviorMate are the Java GUI application, referred to as the UI, Behavior Controller, and Position Controller. The UI coordinates most communication between devices, although sometimes devices directly communicate with each other for performance reasons. The Behavior Controller sends messages to and receives data from nearly any kind of peripheral devices such as 2-photon microscopes, LEDs, speakers, computer displays, electrophysiology systems, etc. The Position Controller is solely responsible for receiving position updates from the rotary encoder. Any number of additional devices may be used by connecting them to the behavior controller using standard connectors. Furthermore, the companion software of VRMate allows for low-cost integration of virtual reality experiments that are already compatible with behaviorMate. The appeal of behaviorMate is its modularity, reliability, and open source code base.

Finally, many published studies have relied on this system and the resulting data collected (Bowler & Losonczy 2023, Tuncdemir et al. 2023, Vancura et al. 2023, Priestley et al. 2022, O'Hare et al. 2022, Rolotti et al. 2022a, Tuncdemir et al. 2022, Rolotti et al. 2022b, Terada et al. 2022, Geiller et al. 2022, Blockus et al. 2021, Grosmark et al. 2021, Dudok et al. 2021a,b, Geiller et al. 2020, Kaufman et al. 2020*a*, Turi et al. 2019, Zaremba et al. 2017, Danielson et al. 2017, 2016). These paradigms varied greatly and included random foraging, goal-oriented learning, multi-modal cue presentations, and context switches (both in VR environments and physical treadmills). Instructions for downloading the UI software and the companion VRMate software, and assembling a Behavior Controller, Position Controller, running wheel, VR monitor frame, and treadmill, can be found on the Losonczy Lab website (www.losonczylab.org/software).

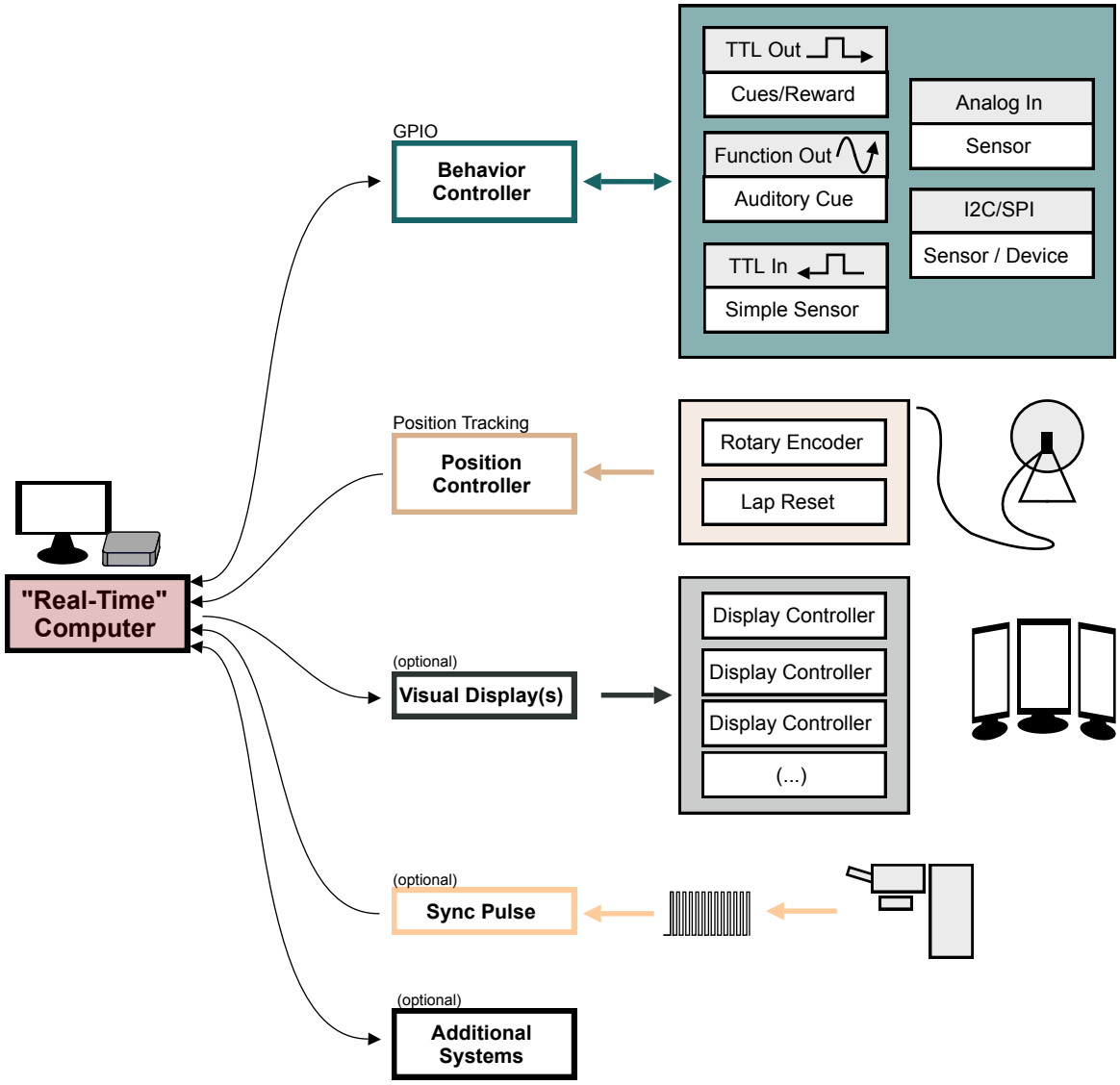

Figure 1. Overview of the behaviorMate system with optional components *Diagram shows the modular components behaviorMate can interact with. The colored arrows show the direction of information flow. For example, the Position Controller module only receives position updates and forwards them to the computer. The Visual Display module sends data to the Display Controllers to render the scene. The Behavior Controller performs a GPIO function and may both send and receive data to and and from the Computer. An optional external Sync Pulse is shown to demonstrate that behaviorMate will log any UDP received UDP packets, such as time stamped sync-signals which could be beneficial in certain setups to synchronize neural data with behavior. Additional Systems may also be implemented on the behaviorMate intranet, taking advantage of the flexible UDP messaging scheme.*

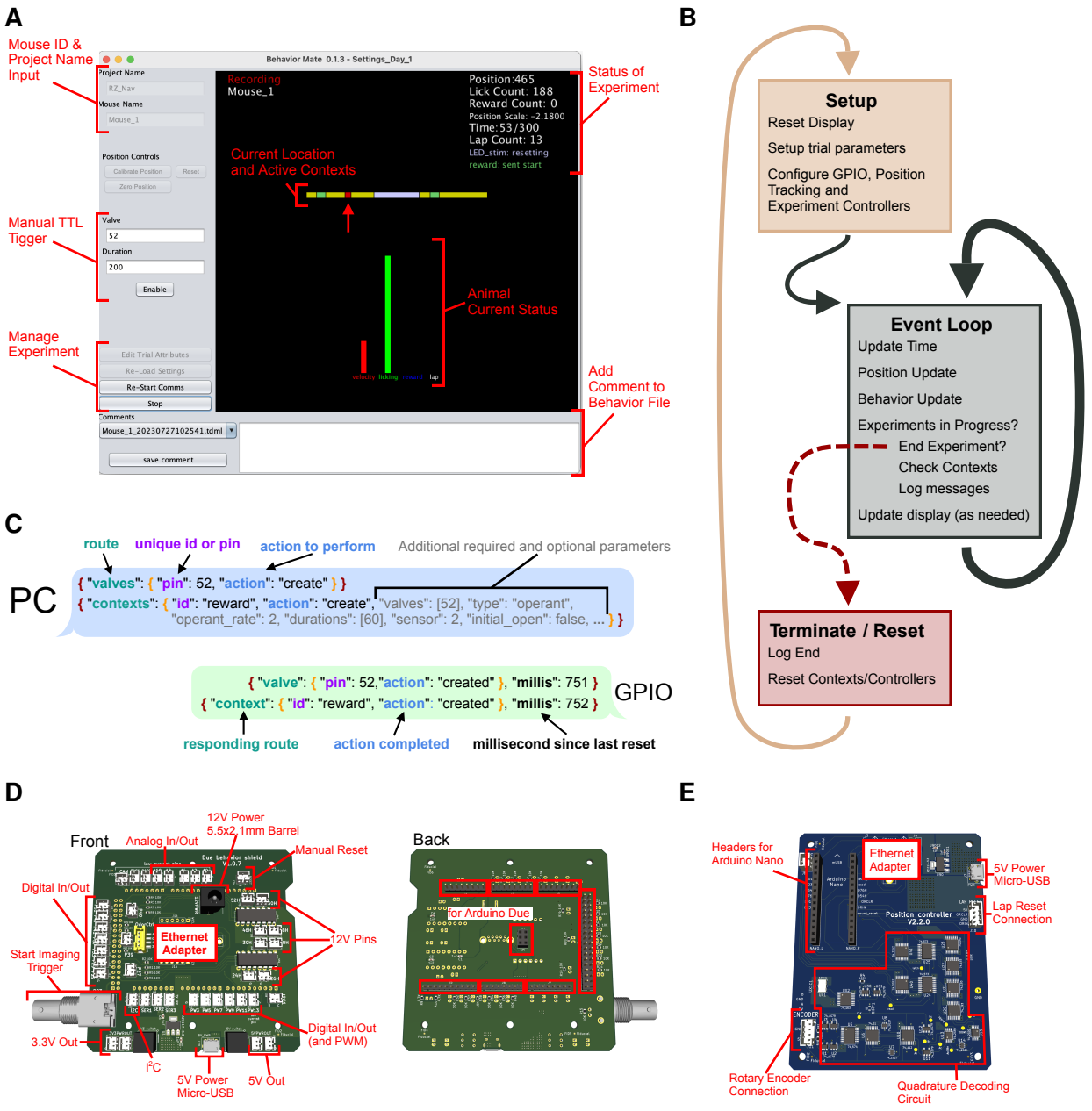

Figure 2. Details of UI function. (**A**) Screenshot of the UI. The interface provides a snapshot of the animal's current status (*center*) as well as cumulative information about the current experiment (*upper-left*). The control panel along the left side provide: *Top.* an input for a project name and mouse id, which control where the resulting behavior file will be saved (these boxes are disabled when an experiment is currently running), *Middle.* controls to trigger the GPIO to send a TTL pulse i.e. to issue a water reward or turn on an LED, and *Bottom.* controls to start/stop experiments as well as to load settings files. (**B**) Details of UI event loop. The UI is continuously executing an event loop on every update step which checks for messages, writes the behavior file and updates the display. (**C**) JSON formatted message passing. The PC sends and receives JSON formatted messages via UDP to active components of the system. Messages have a nested structure to allow for internal routing and subsequent actions to be taken. (**D**) Rendering of the GPIO circuit used. *Left.* JST headers connected to I/O pins and pull-down resisters to allow for easy connections with peripheral devices that can be activated by TTL pulses (both at 3.3V and 12V, depending on the pins used). Additionally, power connections and a BNC output trigger are provided. *Right.* This board attaches to an Arduino Due microcontroller. (**E**) Updates to the position have a dedicated circuit for decoding quadrature information and sending updates to the PC. Connections for a quadrature based rotary encoder, power, and Ethernet are provided. Additionally, an input for a "Lap Reset" sensor is provided and headers to interface with an Arduino Mini.

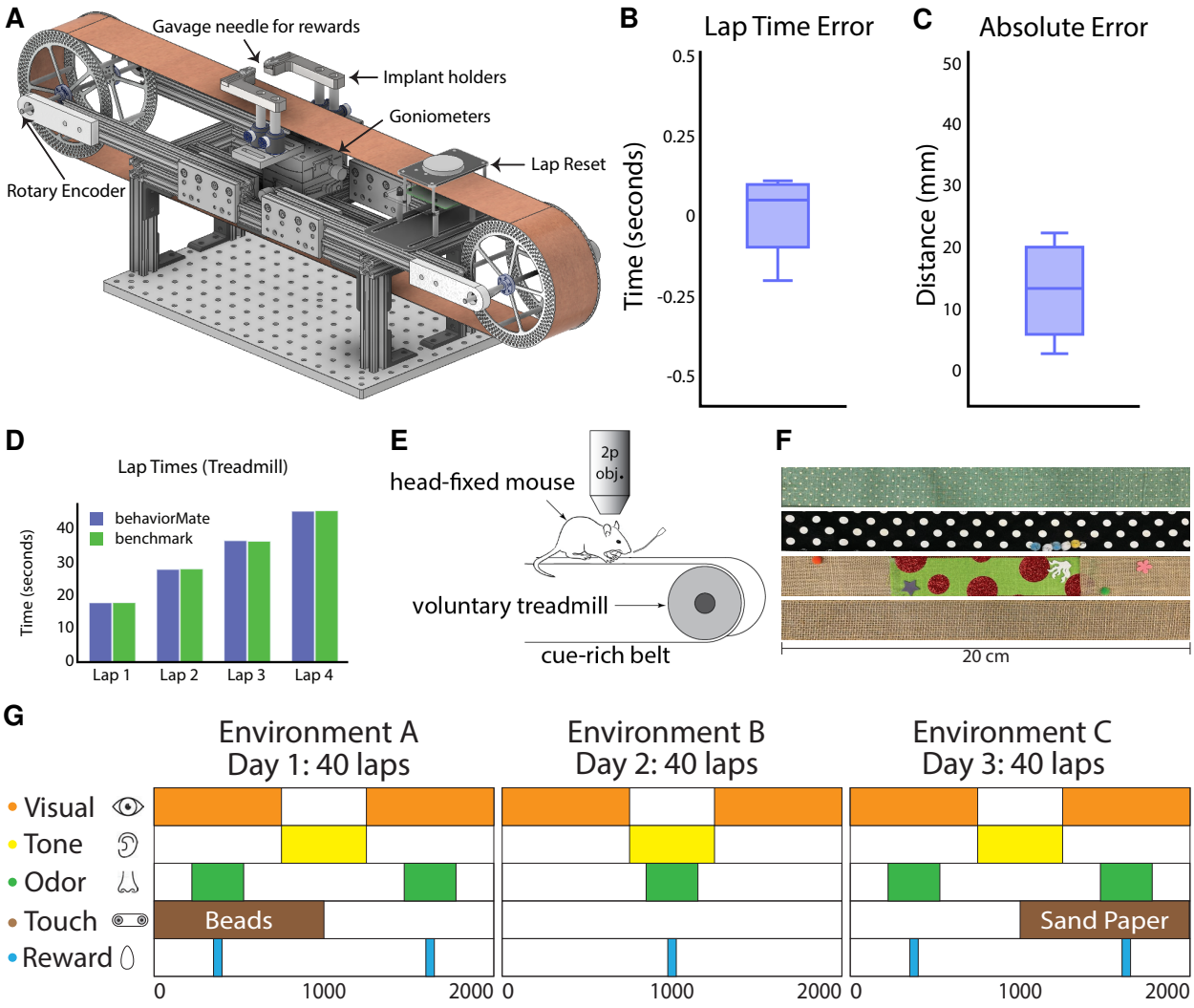

Figure 3. Details of Treadmill System. (**A**) CAD model of voluntary treadmill that can expand or contract to accommodate different sized running belts. Implant holders sit on goniometers to allow mouse head angle to be adjusted. (**B**) Lap time error is defined for each lap (full turn of the treadmill belt) as the difference between the time recorded by behaviorMate and the computer vision benchmark in seconds (time error $= 0.0025 \pm 0.1128$s, mean $\pm$ std). (**C**) Absolute error is defined for each lap as the difference between the position in behaviorMate and the benchmark position in millimeters (position error $= 13 \pm 7.6581$mm, mean $\pm$ std). (**D**) The time to complete a lap according to behaviorMate and the benchmark were nearly identical. (**E**) Schematic of head-fixed behavioral task. Two-photon objective. (**F**) Cue-rich belts for treadmill behavioral tasks. Belts are easily interchangeable. (**G**) Example test condition spanning 3 days and involving 5 sensory cues. Each row represents a sensory modality (visual, auditory, etc.). Cues are tied to locations on the belt or wheel as indicated by the colored rectangles. Locations and durations of cue presentations can be changed across trials.

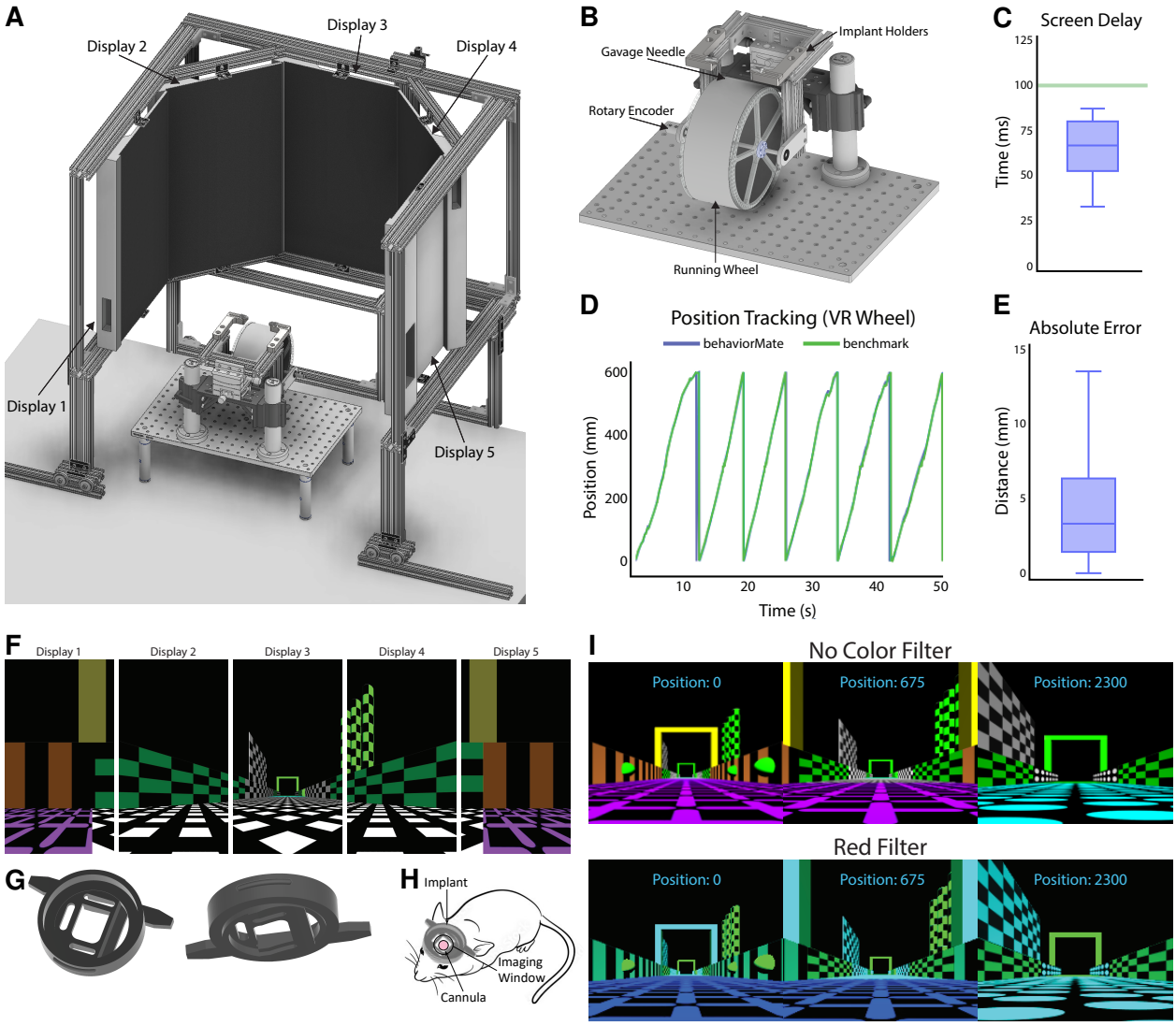

Figure 4. Details of VR System. (**A**) The complete assembly. 5 monitors are suspended in an octagonal pattern in front of the subject and display the virtual scenes. (**B**) A running wheel attached to a goniometer to allow for angle of subject's head to be adjusted. Placed directly underneath the objective. (**C**) Box plot describing the delay between when the running wheel is moved and when the virtual scene is updated to reflect the new position (delay $= 0.0660 \pm 0.01570$s, mean $\pm$ std). (**D**) Plot showing the difference between the position of the animal computed by behaviorMate and the ground-truth position tracked using computer vision. (**E**) Box plot describing the absolute difference between the current position of the mouse according to behaviorMate and the computer vision benchmark at each point in time (position error $= 4.1258 \pm 3.2558$mm, mean $\pm$ std). (**F**) A 2-D projection of the displays which will resemble what the test animal will see. When viewed on the actual monitors, the scene will appear more immersive. (**I**) A VR scene as the subject moves down the virtual track. Modifying a few numerical parameters in the settings files allows one to change the appearance and view angle of a scene. Bottom: A scene with all red color shaders removed. (**G**) Left: Top view of implant. This side will make contact with microscope objective. Right: Bottom view. This side will make contact with the mouse's brain. (**H**) Sketch of how the implant will appear after being surgically attached to skull.

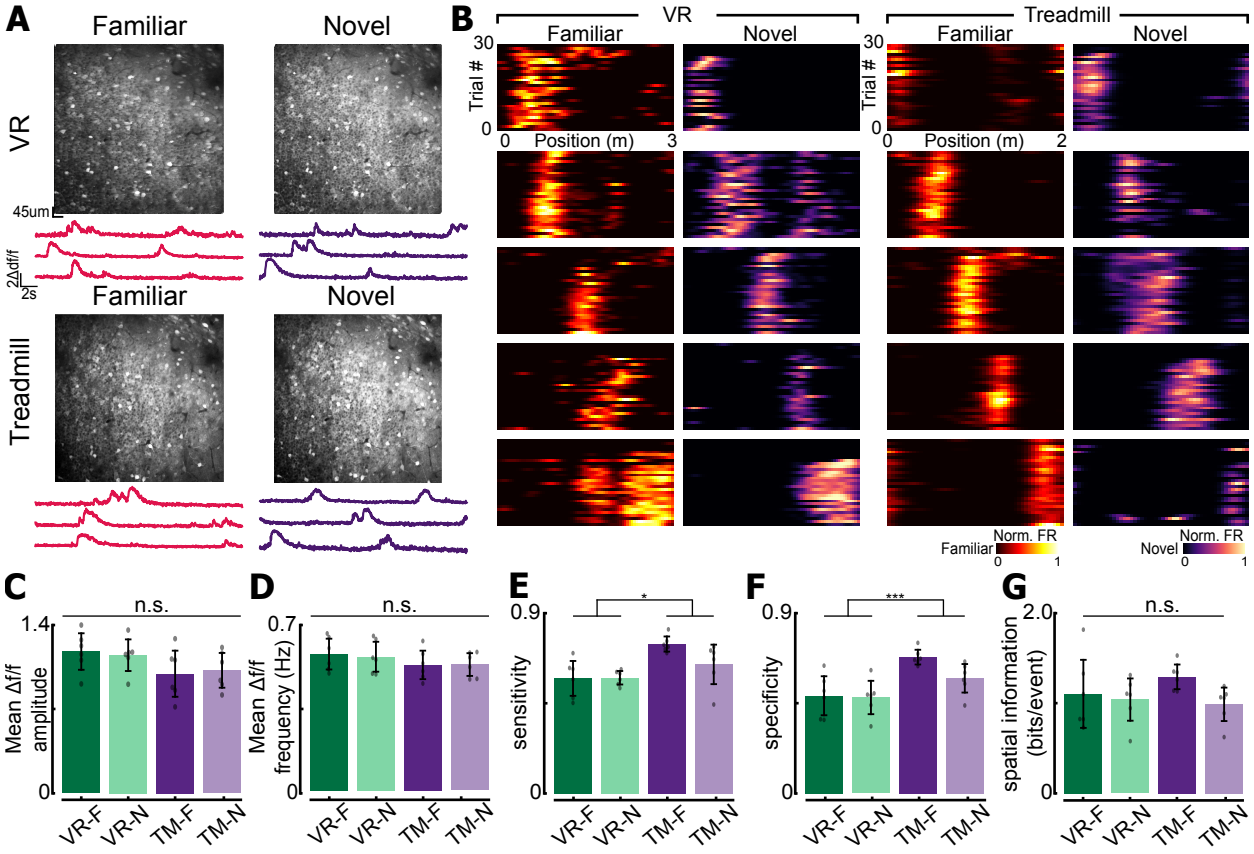

Figure 5. 2-photon imaging of CA1 population for experimental validation (A) Field of view obtained from virtual reality (VR) and treadmill (TM) **(B)** Examples of place cells recorded using VR or TM. In both familiar and novel contexts, place cells encoding different locations on the track are found. Note that they are different place cells from individual animals. **(C)** Mean Δf/f amplitude of place cells recorded each context-environment pair (F = familiar context, N = novel context). **(D)** Mean Δf/f frequency of place cells. For (C) and (D), only significant Δf/f events were used. See methods for details. **(E)** Place sensitivity in each context-environment pair. Significant main effect of environment (VR vs. TM, p = 0.0257). **(F)** Place cell specificity in each context-environment pair. Significant main effect of environment (VR vs. TM, p = 0.008) **(G)** Spatial information (bits/event) in each context-environment pair. Two-way repeated measures of ANOVA was used unless otherwise stated. Refer to Table ST1 for ANOVA table. For panels C-G, n=6 mice (all mice were recorded in both VR and TM systems), 3253 cells in VR classified as significantly tuned place cells VR, and 6101 tuned cells in TM. For all panels, n.s. = non-significant, *P<0.05, ***P<0.001

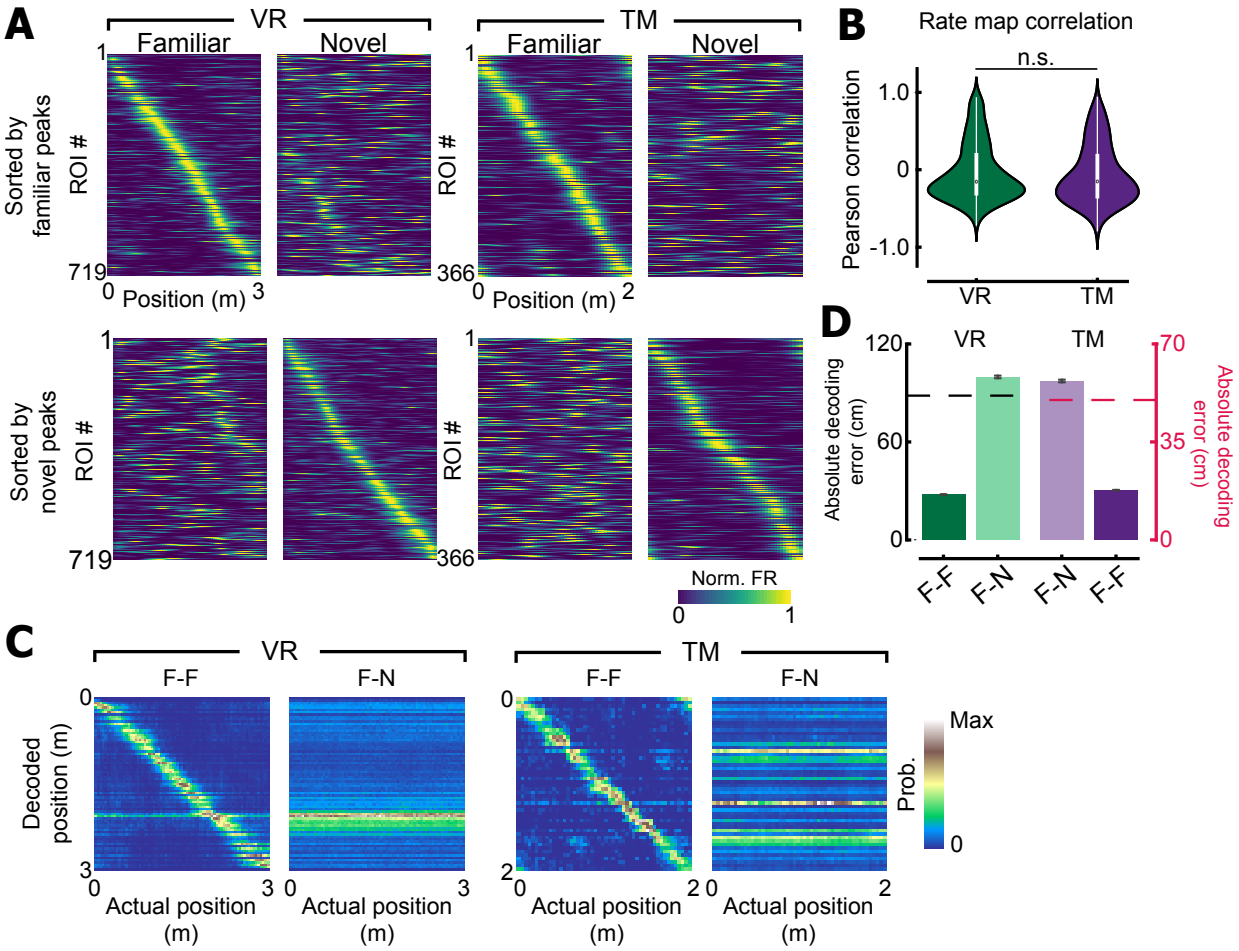

Figure 6. Place cells remap in both environments (A) Place cells recorded in VR or TM sorted by their familiar peak (top) or novel peak (bottom) locations. A subset of CA1 neurons remaps when exposed to novel context in both VR and TM. **(B)** Rate map correlation of individual place cells between familiar and novel contexts. No difference was observed between VR and TM (Wilcoxon rank-sum test, F=1.38, p=0.17). **(C)** Confusion matrices of the Bayesian decoder for VR and TM. F-F indicates a model trained and tested with trials from the familiar context. F-N is a model trained with trials from familiar context, and tested with a novel trial. **(D)** Absolute decoding error for the maximum likelihood estimator. A dashed line indicates a chance level (88±1.96cm for VR/ 50cm for TM, see methods). Note that when the decoder is trained and tested with familiar context trials, absolute decoding accuracy is below chance in both VR and TM (VR: 27.62±1.44cm, p<0.01, TM: 17.78±0.44cm, p<0.01, permutation test). However, when the decoder is trained with familiar context trials, but tested with novel context trials, decoding accuracy was increased to the chance level (VR: 99.80±3.95cm, p=1.00, TM: 56.82±2.46cm, p=1.00).

Acknowledgments

J.C.B. is supported by NIMH F31NS110316. J.B.P. is supported by the EPFL ELISIR fellowship and the Fondation Marina Cuennet-Mauvernay. A.L. is NIMH R01MH124047, NIMH R01MH124867, NINDS R01NS121106,NINDS U01NS115530, NINDS R01NS133381, NINDS R01NS131728, NIA RF1AG080818. We thank Drs. Ali Kaufman and Ally Lowell for troubleshooting and debugging the initial iterations of behaviorMate as well as Dr. Andres Grosmark for continued feedback on the project. Additionally, we thank Dr. Stephanie Herrlinger, Abhishek Shah, and all other members of the Losonczy lab for comments and discussion on the manuscript.

Author Contributions

J.C.B. and A.L. conceived of the original method and experimental requirements. Funding support from A.L. J.C.B. developed software with assistance from W.L. and Z.L. W.L developed custom PCBs. G.Z. developed a JavaFX version of behaviorMate, performed benchmarking and mechanical design. J.C.B., G.Z. and H.C.Y. wrote the original draft with input from all authors. H.C.Y. performed the experimental validation with assistance from B.R. J.B.P. developed the VR layout. J.C.B., G.Z., B.R., Z.L. and J.B.P. performed testing and debugging. All authors contributed to review and editing.

Declaration of Interests

The authors declare no competing interests.

9 Methods

9.1 Lead Contact and Materials Availability

Further information and requests for resources and reagents should be directed to the Lead Contact Attila Losonczy (al2856@columbia.edu). All unique resources generated in this study are available from the Lead Contact with a completed Materials Transfer Agreement.

9.2 Experimental Model and Subject Details

9.2.1 Animals

All experiments were conducted in accordance with the NIH guidelines and with the approval of the Columbia University Institutional Animal Care and Use Committee. Experiments were performed with six adult mice between 8-16 weeks. 5 mice were Unc5b crossed with C57Bl/6, and 1 mouse was VGAT-cre crossed with Ai9 (Jackson Laboratory). 3 female and 3 male mice were used in this experiment to balance for sex.

9.2.2 Surgical procedures

Surgeries were done as previously described (Lovett-Barron et al. 2014). Briefly, animals were anesthetized with isoflurane and injected with jGCaMP8m (rAAV1-syn-GCaMP8m-WPRE, Addgene: 162375-AAV1) in dorsal CA1 (AP: 2.1, ML: 1.4, DV: 1.3, 1,2, 1,1, 100uL per site) using Nanoject III (Drummond). Animals were allowed to recover from the injection surgery for 7 days, and the cortex lying above CA1 was aspirated to implant a glass cannula to enable optical access to CA1. Finally, a titanium headpost was cemented to the skull using dental cement to head-fix animals for 2-photon functional imaging. Analgesics were provided as per Columbia University Institutional Animal Care and Use Committee prior and after the surgeries for up to 3 days.

9.2.3 2-photon imaging and behavior

After recovering from surgeries, animals were habituated to head fixation in a VR environment for 15 minutes. Subsequently, animals were trained to run on a lightweight wheel, and 5% sucrose was given at random locations to motivate animals' to run in the VR or TM. When animals reliably

ran at least 60 laps in VR, or 30 laps in TM, the reward was fixed to either 1 or 2 locations. A 3m VR track was used for all animals except jy030, which had 4m VR track instead, and 2m TM belt was used in all animals. For VR experiments, animals were imaged for one session, and the context switch happened after animals ran 30 laps in familiar context. Animals were required to run at least another 30 laps in the novel context. In TM, imaging was done over two days (first day: familiar belt, second day: novel belt). 2p imaging was performed using a 8kHz resonant scanner (Bruker) and a 16x Nikon water immersion objective (0.8 NA, 3mm working distance). In order to image GCaMP8m signals, 920nm excitation wavelength was used (Coherent), and the power from the tip of the objective never exceeded 100mW. GCaMP signals were collected using a GaAsP photomultiplier tube detector (Hamamatsu, 7422P-40) following amplification via a custom dual stage preamplifier (1.4 x 105 dB, Bruker). All imaging was performed using 512 x 512 pixels with digital zoom between 1 and 1.4.

9.2.4 Calcium signal processing

Imaging data was processed as previously described (Priestley et al. 2022). In short, the SIMA software package (Kaifosh et al. 2014) was used to organize the imaging data set, and the imaging data was processed with Suite2p (Pachitariu et al. 2017) for motion correction, signal/neuropil extraction, and ROI detection. Following ROI detection, individual ROIs were manually curated using the Suite2p GUI to exclude non-somatic ROIs. ΔF/F signals were calculated after subtracting the neuropil and correcting for baseline.

9.2.5 Data analysis

As previously described, all analyses were done using binarized event signals (Ahmed et al. 2020, Priestley et al. 2022). Briefly, ΔF/F signals were deconvolved using OASIS (Friedrich et al. 2017), and the resulting estimated spike amplitude train was binarized by selecting events whose amplitudes were above 4 median absolute deviations of the raw trace. We do not claim that this reflects the true spiking activities of individual ROIs. Binarized event trains were binned into 4cm bins and smoothed with a Gaussian filter (S.D. = 2) to calculate tuning curves. Individual ROIs were classified as a place cell by identifying bins that have activities greater than the 99th percentile

of surrogate tuning curves (Priestley et al. 2022). In short, surrogate average tuning curves were calculated by performing 1,000 circular shuffles of their occupancy of individual trials. Bins that had activity greater than 99th percentile of the tuning curves were identified as potentially significant bins. In order to be classified as a place cell, ROIs needed to have at least 3 consecutive significant bins (12cm), but less than 25 consecutive bins (100cm). Moreover, in order to avoid spurious detection of significant bins, binarized events must have been detected in at least 50% of trials within those significant bins.

For mean ΔF/F amplitude and frequency calculations, only bins with 3 standard deviations above the mean were used. For place cell sensitivity, representing the proportion of laps that had active events within the significant field, the number of laps that had at least 1 binarized event within the significantly detected fields was divided by the total number of laps. For place cell specificity, total number of events within the significant field was divided by the total number of events observed within the lap. Then, it was averaged across all laps to have a single value for each ROI. If multiple fields were detected, they were computed separately and averaged across fields to have a single value for each ROI. For spatial information, it was calculated as described previously (Skaggs & Mcnaughton 1998). For remapping analysis, one animal was used for the analysis (jy065). Given that TM imaging happened across two days, FOVs recorded from TM were matched using CellReg (Sheintuch et al. 2017), and spatial mask correlation was used to find the same ROIs. Results were manually curated within the Suite2p GUI. This was not necessary for VR since the context switch happened within the same session. To calculate rate map correlation, the Pearson correlation coefficient was computed between average tuning curves from the famliar and novel contexts. For TM, given that place cells may anchor on seams of the belt, the shift that generated the maximum population vector correlation between familiar and novel contexts was calculated. All ROIs recorded in the novel context were circularly shifted by the same amount, and the rate map correlation was calculated as described above. A naive Bayesian classifier was used to decode animals' position on the track (Zhang et al. 1998). For each frame and spatial bin:

$$P(pos|a_{all}) = C\left(\prod_{i=1}^{N} f_i(pos)^{a_i}\right) e^{-\tau \sum_{i=1}^{N} f_i(pos)}$$

where N is the total number of cells, $f_i(pos)$ is the average binned activity of a cell, τ is the bin size, and a_i is the activity on the frame. C is the normalization constant. The position bin with the highest probability was selected as a decoded position. Chance level for VR was calculated by randomly shuffling cell labels of the testing set. For TM, a chance level was determined as track length / 4, which is a chance level of a circular environment (50cm). The decoder was trained with n-1 trials and tested on the left out trial. To match ROI numbers recorded between VR and TM, 500 ROIs were randomly selected, and the decoding accuracy was calculated 50 times. Average absolute decoding accuracy was compared to the chance level calculated from each iteration, and the number of times that it was below the chance level was considered to be its p value.

References

Aghajan, Z. M., Acharya, L., Moore, J. J., Cushman, J. D., Vuong, C. & Mehta, M. R. (2015), 'Impaired spatial selectivity and intact phase precession in two-dimensional virtual reality.', *Nature neuroscience* **18**, 121–8.

Ahmed, M. S., Priestley, J. B., Castro, A., Stefanini, F., Canales, A. S. S., Balough, E. M., Lavoie, E., Mazzucato, L., Fusi, S. & Losonczy, A. (2020), 'Hippocampal network reorganization underlies the formation of a temporal association memory.', *Neuron* **107**, 283–291.e6.

Akam, T., Lustig, A., Rowland, J. M., Kapanaiah, S. K., Esteve-Agraz, J., Panniello, M., Márquez, C., Kohl, M. M., Kätzel, D., Costa, R. M. & Walton, M. E. (2022), 'Open-source, python-based, hardware and software for controlling behavioural neuroscience experiments', *eLife* **11**, e67846.
URL: *https://doi.org/10.7554/eLife.67846*

Aronov, D. & Tank, D. W. (2014), 'Engagement of neural circuits underlying 2d spatial navigation in a rodent virtual reality system', *Neuron* **84**, 442–456.
URL: *http://dx.doi.org/10.1016/j.neuron.2014.08.042*

Arriaga, M. & Han, E. B. (2017), 'Dedicated hippocampal inhibitory networks for locomotion and immobility', *Journal of Neuroscience* **37**(38), 9222–9238.
URL: *https://www.jneurosci.org/content/37/38/9222*

Arriaga, M. & Han, E. B. (2019), 'Structured inhibitory activity dynamics in new virtual environments', *eLife* **8**, e47611.
URL: *https://doi.org/10.7554/eLife.47611*

Bittner, K. C., Milstein, A. D., Grienberger, C., Romani, S. & Magee, J. C. (2017), 'Behavioral time scale synaptic plasticity underlies ca1 place fields.', *Science (New York, N.Y.)* **357**, 1033–1036.
URL: *http://www.ncbi.nlm.nih.gov/pubmed/28883072*

Blockus, H., Rolotti, S. V., Szoboszlay, M., Peze-Heidsieck, E., Ming, T., Schroeder, A., Apostolo, N., Vennekens, K. M., Katsamba, P. S., Bahna, F., Mannepalli, S., Ahlsen, G., Honig, B., Shapiro, L., de Wit, J., Losonczy, A. & Polleux, F. (2021), 'Synaptogenic activity of the axon guidance molecule robo2 underlies hippocampal circuit function.', *Cell reports* **37**, 109828.

Bowler, J. C. & Losonczy, A. (2023), 'Direct cortical inputs to hippocampal area ca1 transmit complementary signals for goal-directed navigation', *Neuron* .

URL: *https://linkinghub.elsevier.com/retrieve/pii/S0896627323007018*

Bradski, G. (2000), 'The OpenCV Library', *Dr. Dobb's Journal of Software Tools* .

Campbell, M. G., Ocko, S. A., Mallory, C. S., Low, I. I. C., Ganguli, S. & Giocomo, L. M. (2018), 'Principles governing the integration of landmark and self-motion cues in entorhinal cortical codes for navigation.', *Nature neuroscience* **21**, 1096–1106.

Danielson, N. B., Kaifosh, P., Zaremba, J. D., Lovett-Barron, M., Tsai, J., Denny, C. A., Balough, E. M., Goldberg, A. R., Drew, L. J., Hen, R., Losonczy, A. & Kheirbek, M. A. (2016), 'Distinct contribution of Adult-Born hippocampal granule cells to context encoding', *Neuron* **90**(1), 101–112.

URL: *https://doi.org/10.1016/j.neuron.2016.02.019*

Danielson, N. B., Turi, G. F., Ladow, M., Chavlis, S., Petrantonakis, P. C., Poirazi, P. & Losonczy, A. (2017), 'In vivo imaging of dentate gyrus mossy cells in behaving mice.', *Neuron* **93**, 552–559.e4.

Dombeck, D. A., Harvey, C. D., Tian, L., Looger, L. L. & Tank, D. W. (2010), 'Functional imaging of hippocampal place cells at cellular resolution during virtual navigation', *Nature Neuroscience* **13**(11), 1433–1440.

URL: *https://doi.org/10.1038/nn.2648*

Dombeck, D. A. & Reiser, M. B. (2012), 'Real neuroscience in virtual worlds', *Current Opinion in Neurobiology* **22**(1), 3–10. Neurotechnology.

URL: *https://www.sciencedirect.com/science/article/pii/S0959438811001814*

Dräger, U. C. & Olsen, J. F. (1980), 'Origins of crossed and uncrossed retinal projections in pigmented and albino mice.', *The Journal of comparative neurology* **191**, 383–412.

URL: *http://www.ncbi.nlm.nih.gov/pubmed/7410600*

Dudok, B., Klein, P. M., Hwaun, E., Lee, B. R., Yao, Z., Fong, O., Bowler, J. C., Terada, S., Sparks, F. T., Szabo, G. G., Farrell, J. S., Berg, J., Daigle, T. L., Tasic, B., Dimidschstein, J., Fishell, G., Losonczy, A., Zeng, H. & Soltesz, I. (2021), 'Alternating sources of perisomatic inhibition during

behavior', *Neuron* **109**(6), 997–1012.e9.

URL: *https://doi.org/10.1016/j.neuron.2021.01.003*

Dudok, B., Klein, P. M., Hwaun, E., Lee, B. R., Yao, Z., Fong, O., Bowler, J. C., Terada, S., Sparks, F. T., Szabo, G. G., Farrell, J. S., Berg, J., Daigle, T. L., Tasic, B., Dimidschstein, J., Fishell, G., Losonczy, A., Zeng, H. & Soltesz, I. (2021b), 'Alternating sources of perisomatic inhibition during behavior.', *Neuron* **109**, 997–1012.e9.

Dudok, B., Szoboszlay, M., Paul, A., Klein, P. M., Liao, Z., Hwaun, E., Szabo, G. G., Geiller, T., Vancura, B., Wang, B.-S., McKenzie, S., Homidan, J., Klaver, L. M. F., English, D. F., Huang, Z. J., Buzsáki, G., Losonczy, A. & Soltesz, I. (2021a), 'Recruitment and inhibitory action of hippocampal axo-axonic cells during behavior.', *Neuron* **109**, 3838–3850.e8.

Fischler-Ruiz, W., Clark, D. G., Joshi, N. R., Devi-Chou, V., Kitch, L., Schnitzer, M., Abbott, L. & Axel, R. (2021), 'Olfactory landmarks and path integration converge to form a cognitive spatial map', *Neuron* **109**(24), 4036–4049.e5.

URL: *https://www.sciencedirect.com/science/article/pii/S0896627321007285*

Friedrich, J., Zhou, P., Paninski, L., Staneva, V., Chklovskii, D. & Pnevmatikakis, E. (2017), 'Fast online deconvolution of calcium imaging data', *PLOS Computational Biology* **13**, e1005423.

URL: *http://dx.plos.org/10.1371/journal.pcbi.1005423*

Gamma, E., Helm, R., Johnson, R. & Vlissides, J. M. (1994), *Design Patterns: Elements of Reusable Object-Oriented Software*, 1 edn, Addison-Wesley Professional.

Geiller, T., Fattahi, M., Choi, J.-S. & Royer, S. (2017), 'Place cells are more strongly tied to landmarks in deep than in superficial ca1', *Nature Communications* **8**, 14531.

URL: *http://www.nature.com/articles/ncomms14531*

Geiller, T., Sadeh, S., Rolotti, S. V., Blockus, H., Vancura, B., Negrean, A., Murray, A. J., Rózsa, B., Polleux, F., Clopath, C. & Losonczy, A. (2022), 'Local circuit amplification of spatial selectivity in the hippocampus.', *Nature* **601**, 105–109.

Geiller, T., Vancura, B., Terada, S., Troullinou, E., Chavlis, S., Tsagkatakis, G., Tsakalides, P.,

Ócsai, K., Poirazi, P., Rózsa, B. J. & Losonczy, A. (2020), 'Large-scale 3d two-photon imaging of molecularly identified ca1 interneuron dynamics in behaving mice.', *Neuron* **108**, 968–983.e9.

Gonzalez, K. C., Negrean, A., Liao, Z., Polleux, F. & Losonczy, A. (2023), 'Synaptic basis of behavioral timescale plasticity 1 2 3 4 5', *bioRxiv* .
URL: *https://doi.org/10.1101/2023.10.04.560848*

Grienberger, C. & Magee, J. C. (2022), 'Entorhinal cortex directs learning-related changes in ca1 representations.', *Nature* .
URL: *http://www.ncbi.nlm.nih.gov/pubmed/36323779*

Grosmark, A. D., Sparks, F. T., Davis, M. J. & Losonczy, A. (2021), 'Reactivation predicts the consolidation of unbiased long-term cognitive maps', *Nature Neuroscience* **24**, 1574–1585.
URL: *https://www.nature.com/articles/s41593-021-00920-7*

Hainmueller, T. & Bartos, M. (2018), 'Parallel emergence of stable and dynamic memory engrams in the hippocampus', *Nature* **558**(7709), 292–296.
URL: *https://doi.org/10.1038/s41586-018-0191-2*

Harvey, C. D., Coen, P. & Tank, D. W. (2012), 'Choice-specific sequences in parietal cortex during a virtual-navigation decision task', *Nature* **484**(7392), 62–68.
URL: *https://doi.org/10.1038/nature10918*

Heys, J. G., Rangarajan, K. V. & Dombeck, D. A. (2014), 'The functional micro-organization of grid cells revealed by cellular-resolution imaging', *Neuron* **84**, 1079–1090.
URL: *https://linkinghub.elsevier.com/retrieve/pii/S0896627314009684*

Jayakumar, R. P., Madhav, M. S., Savelli, F., Blair, H. T., Cowan, N. J. & Knierim, J. J. (2019), 'Recalibration of path integration in hippocampal place cells', *Nature* **566**, 533–537.
URL: *http://dx.doi.org/10.1038/s41586-019-0939-3 http://www.nature.com/articles/s41586-019-0939-3*

Jordan, J. T., McDermott, K. D., Frechou, M. A., Shtrahman, M. & Gonçalves, J. T. (2021), 'Treadmill-based task for assessing spatial memory in head-fixed mice.', *STAR protocols* **2**, 100770.

Kaifosh, P., Lovett-Barron, M., Turi, G. F., Reardon, T. R. & Losonczy, A. (2013), 'Septo-hippocampal GABAergic signaling across multiple modalities in awake mice', *Nature Neuroscience* **16**(9), 1182–1184.
URL: *https://doi.org/10.1038/nn.3482*

Kaifosh, P., Zaremba, J. D., Danielson, N. B. & Losonczy, A. (2014), 'Sima: Python software for analysis of dynamic fluorescence imaging data', *Frontiers in Neuroinformatics* **8**.
URL: *http://journal.frontiersin.org/article/10.3389/fninf.2014.00080/abstract*

Kaufman, A. M., Geiller, T. & Losonczy, A. (2020*a*), 'A role for the locus coeruleus in hippocampal ca1 place cell reorganization during spatial reward learning', *Neuron* **105**(6), 1018–1026.e4.
URL: *https://doi.org/10.1016/j.neuron.2019.12.029*

Kaufman, A. M., Geiller, T. & Losonczy, A. (2020*b*), 'A role for the locus coeruleus in hippocampal CA1 place cell reorganization during spatial reward learning', *Neuron* **105**(6), 1018–1026.e4.
URL: *https://doi.org/10.1016/j.neuron.2019.12.029*

Knierim, J. J. & Rao, G. (2003), 'Distal landmarks and hippocampal place cells: Effects of relative translation versus rotation', *Hippocampus* **13**, 604–617.

Leutgeb, S., Leutgeb, J. K., Barnes, C. A., Moser, E. I., McNaughton, B. L. & Moser, M.-B. (2005), 'Independent codes for spatial and episodic memory in hippocampal neuronal ensembles', *Science* **309**, 619–623.
URL: *https://www.science.org/doi/10.1126/science.1114037*

Lovett-Barron, M., Kaifosh, P., Kheirbek, M. A., Danielson, N., Zaremba, J. D., Reardon, T. R., Turi, G. F., Hen, R., Zemelman, B. V. & Losonczy, A. (2014), 'Dendritic inhibition in the hippocampus supports fear learning', *Science* **343**(6173), 857–863.
URL: *https://www.science.org/doi/abs/10.1126/science.1247485*

Mauk, M. D. & Buonomano, D. V. (2004), 'The neural basis of temporal processing', *Annual Review of Neuroscience* **27**(1), 307–340. PMID: 15217335.
URL: *https://doi.org/10.1146/annurev.neuro.27.070203.144247*

Muller, R. & Kubie, J. (1987), 'The effects of changes in the environment on the spatial firing of hippocampal complex-spike cells', *The Journal of Neuroscience* **7**, 1951–1968.
URL: *https://www.jneurosci.org/lookup/doi/10.1523/JNEUROSCI.07-07-01951.1987*

O'Hare, J. K., Gonzalez, K. C., Herrlinger, S. A., Hirabayashi, Y., Hewitt, V. L., Blockus, H., Szoboszlay, M., Rolotti, S. V., Geiller, T. C., Negrean, A., Chelur, V., Polleux, F. & Losonczy, A. (2022), 'Compartment-specific tuning of dendritic feature selectivity by intracellular ca2+ release.', *Science (New York, N.Y.)* **375**, eabm1670.

O'Keefe, J. & Dostrovsky, J. (1971), 'The hippocampus as a spatial map. preliminary evidence from unit activity in the freely-moving rat', *Brain Research* **34**, 171–175.
URL: *https://linkinghub.elsevier.com/retrieve/pii/0006899371903581*

Pachitariu, M., Stringer, C., Dipoppa, M., Schröder, S., Rossi, L. F., Dalgleish, H., Carandini, M. & Harris, K. D. (2017), 'Suite2p: beyond 10,000 neurons with standard two-photon microscopy', *bioRxiv* .
URL: *https://doi.org/10.1101/061507*

Priestley, J. B., Bowler, J. C., Rolotti, S. V., Fusi, S. & Losonczy, A. (2022), 'Signatures of rapid plasticity in hippocampal ca1 representations during novel experiences', *Neuron* **110**(12), 1978–1992.e6.
URL: *https://doi.org/10.1016/j.neuron.2022.03.026*

Radvansky, B. A. & Dombeck, D. A. (2018), 'An olfactory virtual reality system for mice', *Nature Communications* **9**.
URL: *https://doi.org/10.7554/eLife.67846*

Ravassard, P., Kees, A., Willers, B., Ho, D., Aharoni, D. A., Cushman, J., Aghajan, Z. M. & Mehta, M. R. (2013), 'Multisensory control of hippocampal spatiotemporal selectivity.', *Science (New York, N.Y.)* **340**, 1342–1346.

Renaudineau, S., Poucet, B. & Save, E. (2007), 'Flexible use of proximal objects and distal cues by hippocampal place cells', *Hippocampus* **17**, 381–395.

Rolotti, S. V., Ahmed, M. S., Szoboszlay, M., Geiller, T., Negrean, A., Blockus, H., Gonzalez, K. C., Sparks, F. T., Canales, A. S. S., Tuttman, A. L., Peterka, D. S., Zemelman, B. V., Polleux, F. & Losonczy, A. (2022b), 'Local feedback inhibition tightly controls rapid formation of hippocampal place fields', *Neuron* **110**, 783–794.e6.
URL: *https://linkinghub.elsevier.com/retrieve/pii/S089662732100996X*

Rolotti, S. V., Blockus, H., Sparks, F. T., Priestley, J. B. & Losonczy, A. (2022a), 'Reorganization of ca1 dendritic dynamics by hippocampal sharp-wave ripples during learning.', *Neuron* **110**, 977–991.e4.

Royer, S., Zemelman, B. V., Losonczy, A., Kim, J., Chance, F., Magee, J. C. & Buzsáki, G. (2012), 'Control of timing, rate and bursts of hippocampal place cells by dendritic and somatic inhibition', *Nature Neuroscience* **15**(5), 769–775.
URL: *https://doi.org/10.1038/nn.3077*

Saunders, J. L., Ott, L. A. & Wehr, M. (2022), 'Autopilot: Automating experiments with lots of raspberry pis', *bioRxiv* .

Sheffield, M. E., Adoff, M. D. & Dombeck, D. A. (2017), 'Increased prevalence of calcium transients across the dendritic arbor during place field formation', *Neuron* **96**(2), 490–504.e5.
URL: *https://www.sciencedirect.com/science/article/pii/S0896627317308735*

Sheintuch, L., Rubin, A., Brande-Eilat, N., Geva, N., Sadeh, N., Pinchasof, O. & Ziv, Y. (2017), 'Tracking the same neurons across multiple days in ca2+ imaging data.', *Cell reports* **21**, 1102–1115.

Skaggs, W. E. & Mcnaughton, B. L. (1998), 'Spatial firing properties of hippocampal ca1 populations in an environment containing two visually identical regions', *The Journal of neuroscience* **18**, 8455–66.

Sofroniew, N. J., Cohen, J. D., Lee, A. K. & Svoboda, K. (2014), 'Natural whisker-guided behavior by head-fixed mice in tactile virtual reality', *The Journal of Neuroscience* **34**, 9537–9550.

Terada, S., Geiller, T., Liao, Z., O'Hare, J., Vancura, B. & Losonczy, A. (2022), 'Adaptive stimulus

selection for consolidation in the hippocampus', *Nature* **601**(7892), 240–244.
URL: *https://doi.org/10.1038/s41586-021-04118-6*

Tuncdemir, S. N., Grosmark, A. D., Chung, H., Luna, V. M., Lacefield, C. O., Losonczy, A. & Hen, R. (2023), 'Adult-born granule cells facilitate remapping of spatial and non-spatial representations in the dentate gyrus.', *Neuron* .

Tuncdemir, S. N., Grosmark, A. D., Turi, G. F., Shank, A., Bowler, J. C., Ordek, G., Losonczy, A., Hen, R. & Lacefield, C. O. (2022), 'Parallel processing of sensory cue and spatial information in the dentate gyrus', *Cell Reports* **38**.

Turi, G. F., Li, W. K., Chavlis, S., Pandi, I., O'Hare, J., Priestley, J. B., Grosmark, A. D., Liao, Z., Ladow, M., Zhang, J. F., Zemelman, B. V., Poirazi, P. & Losonczy, A. (2019), 'Vasoactive intestinal polypeptide-expressing interneurons in the hippocampus support goal-oriented spatial learning', *Neuron* **101**, 1150–1165.e8.

Vancura, B., Geiller, T., Grosmark, A., Zhao, V. & Losonczy, A. (2023), 'Inhibitory control of sharp-wave ripple duration during learning in hippocampal recurrent networks.', *Nature neuroscience* **26**, 788–797.

Warren, R. A., Zhang, Q., Hoffman, J. R., Li, E. Y., Hong, Y. K., Bruno, R. M. & Sawtell, N. B. (2021), 'A rapid whisker-based decision underlying skilled locomotion in mice', *eLife* **10**, e63596.
URL: *https://doi.org/10.7554/eLife.63596*

Zaremba, J. D., Diamantopoulou, A., Danielson, N. B., Grosmark, A. D., Kaifosh, P. W., Bowler, J. C., Liao, Z., Sparks, F. T., Gogos, J. A. & Losonczy, A. (2017), 'Impaired hippocampal place cell dynamics in a mouse model of the 22q11.2 deletion', *Nature Neuroscience* **20**(11), 1612–1623.
URL: *https://doi.org/10.1038/nn.4634*

Zhang, K., Ginzburg, I., McNaughton, B. L. & Sejnowski, T. J. (1998), 'Interpreting neuronal population activity by reconstruction: unified framework with application to hippocampal place cells.', *Journal of neurophysiology* **79**, 1017–44.

10 Supplementary Materials

10.1 Table of Contents

- **Link to website for quickstart guides and file downloads:**

  https://www.losonczylab.org/behaviormate

- **Link to example settings files:**

  https://github.com/losonczylab/behaviorMate/tree/main/example_settings_files

- **Link to source code for the UI:**

  https://github.com/losonczylab/behaviorMate

- **Link to online java documentation for the UI:**

  https://www.losonczylab.org/behaviorMate-1.0.0

- **Link to electronics schematics:**

  https://github.com/losonczylab/Hardware/tree/main/Electronics

- **Link to Arduino Firmware and Debugging Utilities:**

  https://github.com/losonczylab/behaviormate_utils/

- **Link to treadmill and VR CAD files:**

  https://github.com/losonczylab/Hardware/

- **Figure S1: Additional details on VR Frame**
- **Figure S2: Experimental Validation**
- **Table ST1: ANOVA results**

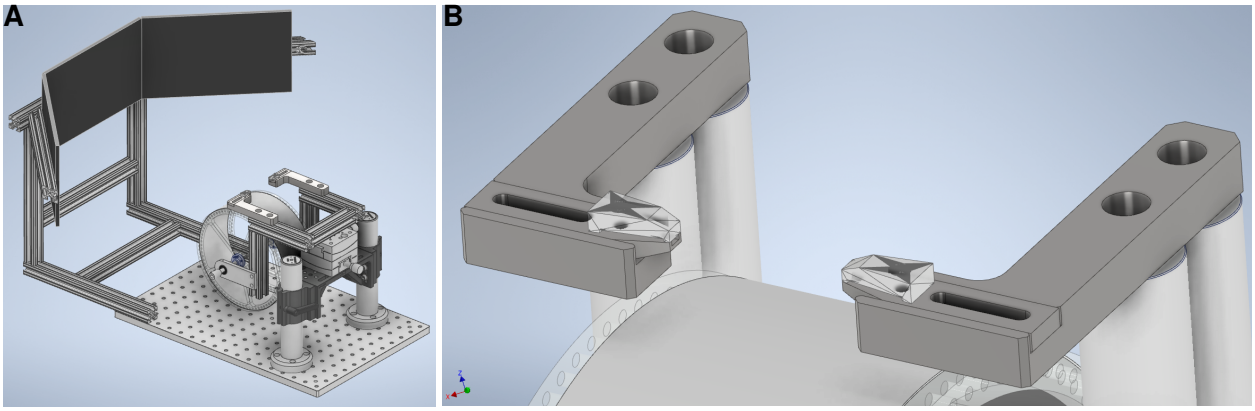

Figure S1. Alternative VR system (**A**) A version of the VR system that can better fit in tight spaces. Can be made to be self-contained in the event it needs to be moved or swapped with other systems. (**B**) Fixes the position of the mouse's head by tightly clamping onto each side of the bar implanted into its skull. The screws used to close the two clamps are not shown.

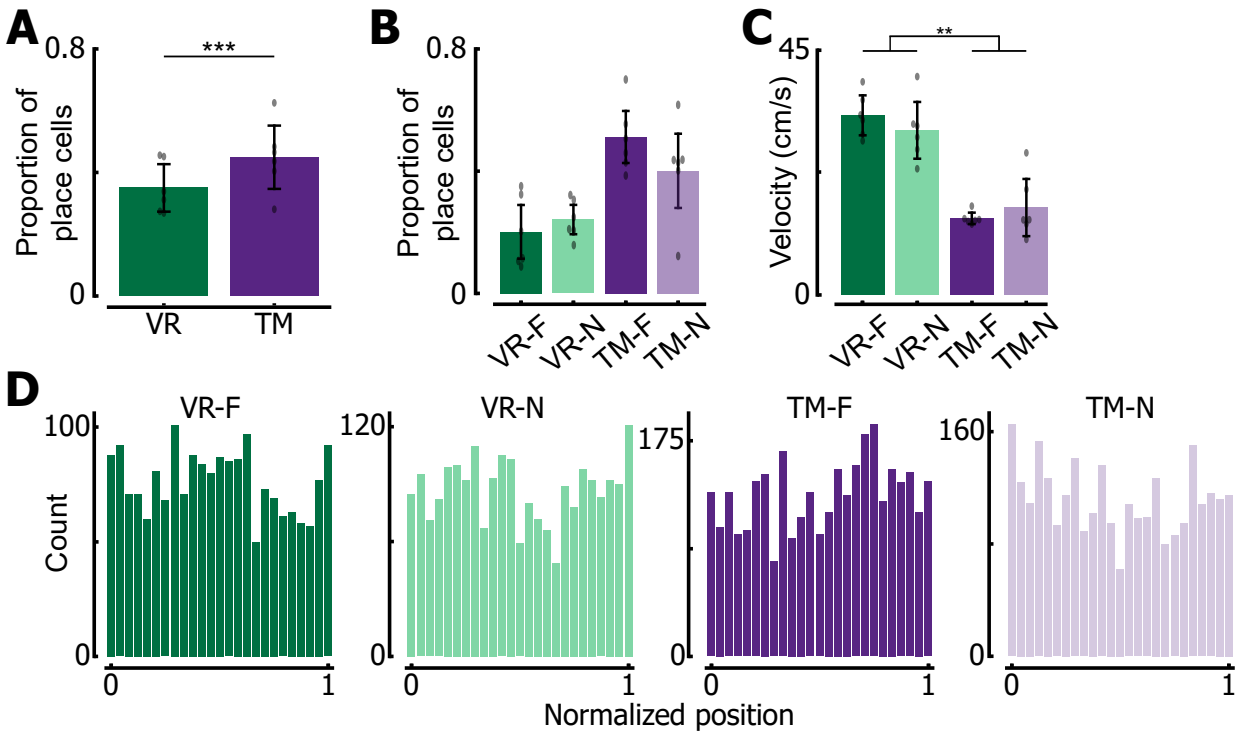

Figure S2. Additional place cell properties (A) Proportion of place cells in VR and TM. VR had significantly less ROIs that were classified as a place cell compared to TM (two sample z-test, F=14.96 p=1.27e^{-50}). **(B)** Proportion of place cells in each context-environment pair. **(C)** Mean velocity per lap in each context-environment pair. Significant main effect of environment (F(1,5)=35.95, p=0.0019). **(D)** Peak location of place cells recorded in each context-environment pair. Note that place cells tile entire track in all pairs.

Table ST1. ANOVA Results

$\triangle$ **F/F Amplitude** (Fig. 5C)	**F value**	**DOF**	**PR > F**
Context (familiar vs. novel)	0.0037	(1, 5)	0.9537
Environment (VR vs Treadmill)	1.9387	(1, 5)	0.2226
Interaction	1.4364	(1, 5)	0.2844

$\triangle$ **F/F Frequency** (Fig. 5D)	**F value**	**DOF**	**PR > F**
Context (familiar vs. novel)	0.0789	(1, 5)	0.7901
Environment (VR vs Treadmill)	4.3798	(1, 5)	0.0906
Interaction	0.2100	(1, 5)	0.6660

Sensitivity (Fig. 5E)	**F value**	**DOF**	**PR > F**
Context (familiar vs. novel)	3.3375	(1, 5)	0.1273
Environment (VR vs Treadmill)	9.8469	(1, 5)	0.0257
Interaction	4.8578	(1, 5)	0.0787

Specificity (Fig. 5F)	**F value**	**DOF**	**PR > F**
Context (familiar vs. novel)	2.5521	(1, 5)	0.1710
Environment (VR vs Treadmill)	53.3198	(1, 5)	0.0008
Interaction	3.7118	(1, 5)	0.1120

Spatial Information (Fig. 5G)	**F value**	**DOF**	**PR > F**
Context (familiar vs. novel)	3.7945	(1, 5)	0.1090
Environment (VR vs Treadmill)	0.7352	(1, 5)	0.4304
Interaction	2.3396	(1, 5)	0.1867

Velocity (Fig. S2C)	**F value**	**DOF**	**PR > F**
Context (familiar vs. novel)	0.1262	(1, 5)	0.7369
Environment (VR vs Treadmill)	35.9548	(1, 5)	0.0019
Interaction	2.7198	(1, 5)	0.1600