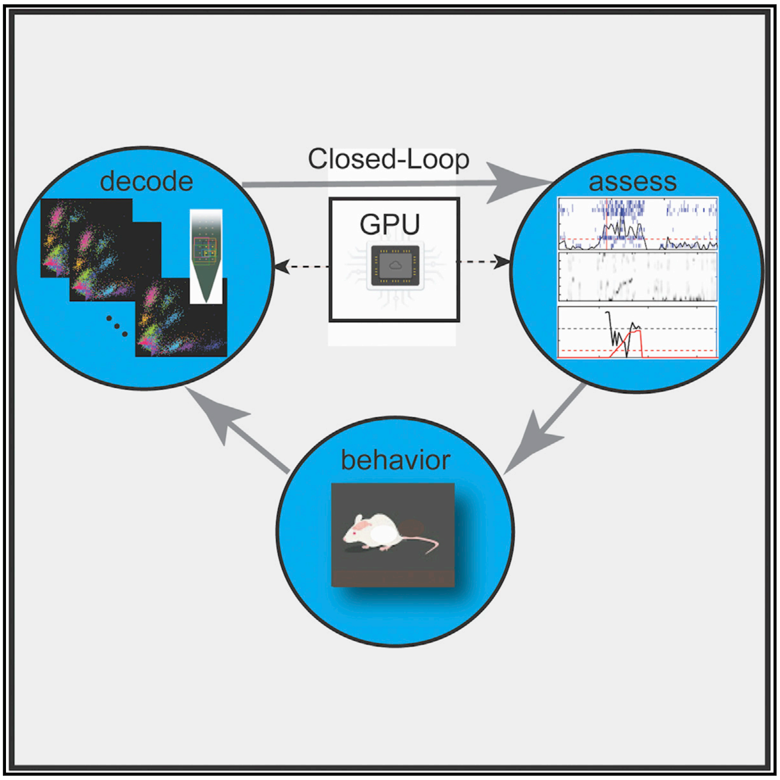
Real-Time Readout of Large-Scale Unsorted Neural Ensemble Place Codes

Cell Reports

Report

Real-Time Readout of Large-Scale Unsorted Neural Ensemble Place Codes

Graphical Abstract



Highlights

- Spike-sorting-free decoding reconstructs the rat's position with ultrafast speed
- GPU-powered population decoding significantly speeds up multi-core CPU-based system
- GPU computing empowers real-time assessment of decoded "memory replay" candidates
- Open-source software toolkit supports closed-loop content-triggered intervention

Authors

Sile Hu, Davide Ciliberti,
Andres D. Grosmark, ...,
Matthew A. Wilson, Fabian Kloosterman,
Zhe Chen

Correspondence

fabian.kloosterman@nerf.be (F.K.),
zhe.chen@nyulangone.org (Z.C.)

In Brief

The hippocampal and neocortical neuronal ensembles encode rich spatial information in navigation. Hu et al. develop computational techniques that accommodate real-time decoding and assessment of large-scale unsorted neural ensemble place codes during running behavior and sleep.

Hu et al., 2018, Cell Reports *25*, 2635–2642
December 4, 2018 © 2018 The Author(s).
https://doi.org/10.1016/j.celrep.2018.11.033

Real-Time Readout of Large-Scale Unsorted Neural Ensemble Place Codes

Sile Hu,[1,2] Davide Ciliberti,[3,4,5] Andres D. Grosmark,[6] Frédéric Michon,[3,4] Daoyun Ji,[7] Hector Penagos,[8] György Buzsáki,[9] Matthew A. Wilson,[8] Fabian Kloosterman,[3,4,5,*] and Zhe Chen[2,10,*]
[1]Department of Instrument Science and Technology, College of Biomedical Engineering and Instrument Science, Zhejiang University, Hangzhou, Zhejiang 310027, China
[2]Department of Psychiatry, Department of Neuroscience and Physiology, School of Medicine, New York University, New York, NY 10016, USA
[3]Neuro-Electronics Research Flanders (NERF), IMEC, Leuven, Belgium
[4]Brain & Cognition Research Unit, KU Leuven, Leuven, Belgium
[5]VIB, Leuven, Belgium
[6]Department of Neuroscience, Columbia University Medical Center, New York, NY 10019, USA
[7]Department of Molecular and Cellular Biology, Department of Neuroscience, Baylor College of Medicine, Houston, TX 77030, USA
[8]The Picower Institute for Learning and Memory, Massachusetts Institute of Technology, Cambridge, MA 02134, USA
[9]The Neuroscience Institute, New York University School of Medicine, New York, NY 10016, USA
[10]Lead Contact
*Correspondence: fabian.kloosterman@nerf.be (F.K.), zhe.chen@nyulangone.org (Z.C.)
https://doi.org/10.1016/j.celrep.2018.11.033

SUMMARY

Uncovering spatial representations from large-scale ensemble spike activity in specific brain circuits provides valuable feedback in closed-loop experiments. We develop a graphics processing unit (GPU)-powered population-decoding system for ultrafast reconstruction of spatial positions from rodents' unsorted spatiotemporal spiking patterns, during run behavior or sleep. In comparison with an optimized quad-core central processing unit (CPU) implementation, our approach achieves an ∼20- to 50-fold increase in speed in eight tested rat hippocampal, cortical, and thalamic ensemble recordings, with real-time decoding speed (approximately fraction of a millisecond per spike) and scalability up to thousands of channels. By accommodating parallel shuffling in real time (computation time <15 ms), our approach enables assessment of the statistical significance of online-decoded "memory replay" candidates during quiet wakefulness or sleep. This open-source software toolkit supports the decoding of spatial correlates or content-triggered experimental manipulation in closed-loop neuroscience experiments.

INTRODUCTION

An important task of systems neuroscience is to read out information encoded in high-dimensional multi-neuronal spatiotemporal spiking patterns. Advances in two- or three-dimensional multielectrode recording devices enable the collection of *in vivo* ensemble spike activity from neocortical and subcortical circuits, with electrode arrays consisting of hundreds or even thousands of channels (Berényi et al., 2014; Shobe et al., 2015; Michon et al., 2016; Rios et al., 2016; Jun et al., 2017a). To deal with large quantity of data, scaling and speeding up neural data analysis has become an emerging research topic in neuroscience. The identification of complex spatiotemporal spiking patterns and their statistical testing are challenging, error prone, and often time consuming.

The analysis challenge is especially daunting for online brain machine interface (BMI) applications that use high-density multielectrode sensors (Rossant et al., 2016; Jun et al., 2017b) and for which a real-time deadline has to be met. Such applications include closed-loop neuroscience experiments that allow scientists to investigate the causal role of specific neural activity patterns by delivering to targeted neural circuits a state-dependent neurofeedback (Grosenick et al., 2015; Buzsáki et al., 2015; El Hady 2016; Girardeau et al., 2009). Closed-loop neuroscience experiments aimed at investigating cognitive processes, like learning and memory, impose a demand to read out ("decode") neuronal population codes in real time at tens of millisecond latency (Tsai et al., 2017; Deng et al., 2016; Rothschild et al., 2017; Ciliberti et al., 2018).

Spatial navigation is a common rodent behavioral task for studying spatial and episodic memories. Neural coding of space, or "place codes," has been reported in many brain structures, including the hippocampus, entorhinal cortex, primary visual cortex (V1), retrosplenial cortex, and parietal cortex (O'Keefe and Dostrovsky, 1971; Hafting et al., 2005; Whitlock et al., 2008; Mao et al., 2017, 2018; Ji and Wilson, 2007; Haggerty and Ji, 2015). The readout of the content of memory reactivation during rest and slow wave sleep (SWS) is conventionally carried out in an offline analysis (Davidson et al., 2009; Pfeiffer and Foster, 2013; Roumis and Frank, 2015; Gomperts et al., 2015). An "online" extension of "place"-decoding analysis has been proposed using a Bayesian spike-sorting-free encoding and decoding framework (Chen et al., 2012; Kloosterman et al., 2014; Deng et al., 2015; Sodkomkham et al., 2016).

Our population analyses of unsorted ensemble spikes consist of two phases (Figure S1A). The encoding phase estimates the joint probability density of the feature vector of spike waveform

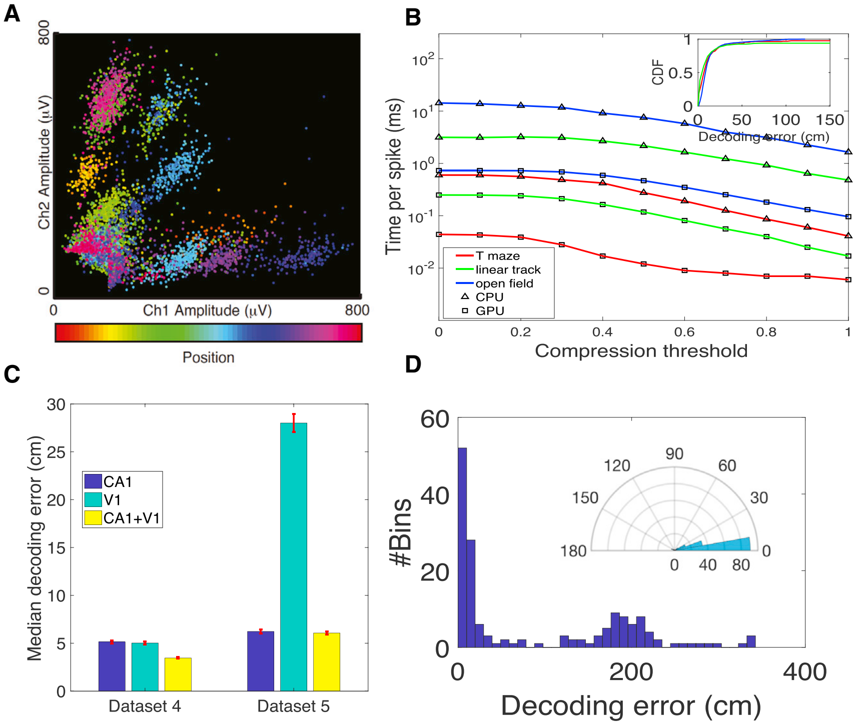


Figure 1. GPU Decoding Analysis on Rat Tetrode Recordings

(A) Illustration of spike peak amplitude features in two projected channels in rat hippocampal tetrode recordings. Spikes are color coded according to their associated spatial position on a linear track.

(B) GPU versus CPU decode time with respect to the compression threshold for KDE training set (0 meaning no compression) in the T-maze (red), linear track (green), and open field environment (blue). The inset shows the cumulative distribution function (CDF) curve of decoding error derived from three datasets with a zero compression threshold (median error: T-maze 7.7 cm; linear track, 6.46 cm; open field, 8.76 cm; 30 μV BW used in all cases).

(C) Ten-fold cross-validated median decoding error in the figure “8” maze derived from V1 or CA1 multi-unit activity or combined (datasets 4 and 5). All results were produced without kernel compression. Error bar represents SEM.

(D) Error histograms for separate decoding the spatial position (median error: 19.23 cm) and head direction (inset, median error: 8.15°) from the rat anterior dorsal thalamus (dataset 6).

See also Figures S1 and S2.

and spatial position. The decoding phase reconstructs the optimal position that yields the maximum likelihood of a temporally marked point process (STAR Methods; Kloosterman et al., 2014). The challenge of the online scenario can be resolved using ultra-flexible multi-threaded software running on a multi-core central processing unit (CPU) system (Ciliberti and Kloosterman, 2017; Ciliberti et al., 2018). However, the scalability of this system and other BMIs running on multi-threaded CPU systems is dependent on the limited number of CPU cores (Fischer et al., 2014). Here, we show a significant speedup of the decoding algorithm by employing a highly customized graphics processing unit (GPU) implementation on a standard quad-core PC, which greatly enhances the speed and scalability potential compared to a pure CPU solution. We also extend the application of neural decoding of unsorted spikes from tetrode to high-density silicon probe recordings.

RESULTS

GPU-Powered Decoding Significantly Speeds Up the Decode Speed

We tested CPU- and GPU-based neural decoding implementations in eight datasets with rat hippocampal, neocortical, and thalamic ensemble recordings during spatial navigation in one- or two-dimensional environments (Figure S1C; Table S1). Recordings were performed using either tetrodes or silicon probes. As per previous implementations (Kloosterman et al., 2014), for tetrode recordings, we selected the peak amplitude of recorded spike waveforms to construct a four-dimensional feature vector (Figure 1A). The GPU-based implementation showed a significant speedup compared to the CPU implementation, confirming the benefit of parallelization at multiple levels in the decoding algorithm. In a representative one-dimensional spatial environment (dataset 1), a full or uncompressed model achieved good cross-validated decoding accuracy and a large speed gap between GPU and CPU. Progressively higher compression thresholds resulted in less accurate but faster decoding, as shown previously (Sodkomkham et al., 2016; Ciliberti et al., 2018). At a compression threshold of 0.5, we achieved a decode time of ~0.02 ms/spike in GPU decoding, as compared to ~0.44 ms/spike in quad-core 8-threaded CPU decoding (Figure 1B; Table S2), which is equivalent to ~20-fold speedup. Notably, the decode time depended jointly on the compression threshold and the kernel bandwidth (BW) parameter, but the compression threshold had a negligible effect on the decode time when using a small BW parameter (Figure S2A). Meanwhile, the decoding accuracy was robust with respect to a wide range of compression thresholds (Figure S2C). The results were robust and consistent in all rat hippocampal CA1 recordings (datasets 1–3). Notably, our approach performed well not only in decode time per spike but also in decoding accuracy, as demonstrated by representing multimodal distributions of two-dimensional trajectories in the open field environment (Figure S2D; Video S1).

The place code is by no means restricted to hippocampal regions. Next, we tested our approach using simultaneous tetrode recordings of rat CA1 and V1 (primary visual cortex) during a continuous spatial alternation task in a “figure 8” maze. We assessed the decoding accuracy using unsorted multi-unit activity (MUA) of CA1 or V1 or both combined. Our analysis suggested that V1 ensemble spike activity contained rich spatial information (Ji and Wilson, 2007; Haggerty and Ji, 2015), and combining CA1 and V1 spike data further improved

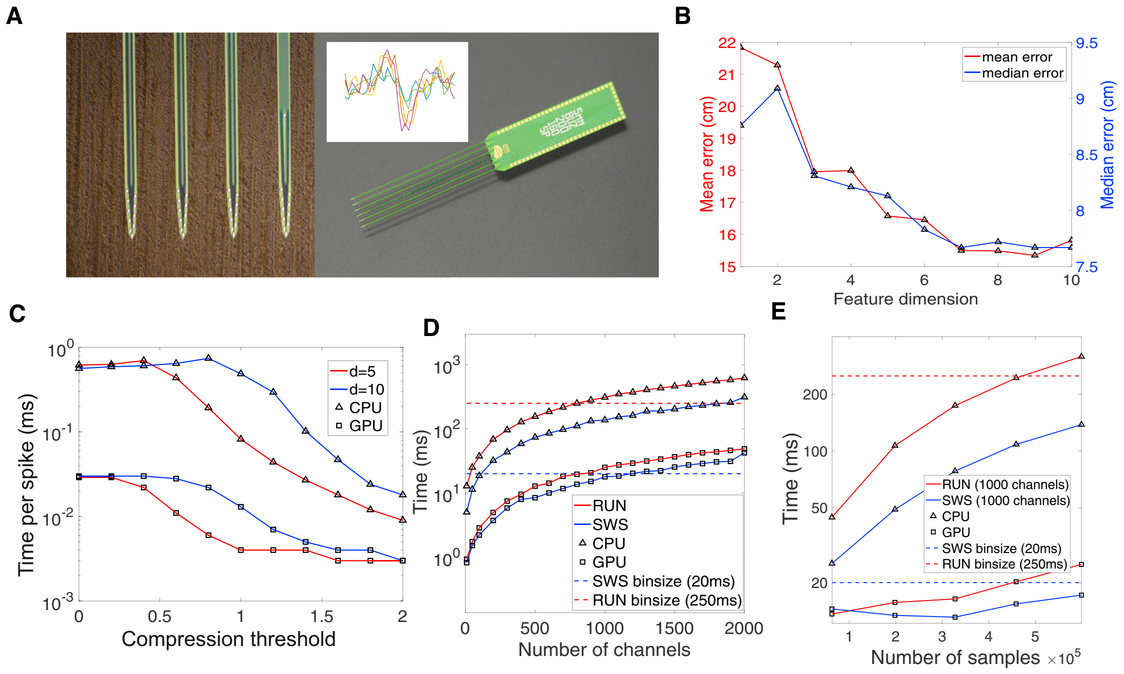


Figure 2. GPU Decoding Analysis on Rat Silicon Probe Recordings

(A) Custom high-density silicon probe (inset: representative spike waveforms from 5 channels). Each shank has 10 recording sites or channels.

(B) Mean (red) and median (blue) decoding error of unsorted CA1 multi-unit activity with respect to varying dimensions of feature vectors (dataset 7). At each condition, the BW parameters were optimized using grid search.

(C) Comparison of CPU versus GPU decode time with respect to varying compression thresholds and two different sets of features (d = 5 and d = 10).

(D) Scalability of GPU decoding for real-time processing in RUN and SWS with respect to the number of channels.

(E) Assuming 1,000 channels, decode time depended on the number of KDE samples.

See also Figures S1 and S3.

the decoding accuracy (Figure 1C). In joint CA1+V1 decoding, kernel BW parameters for spike amplitude were optimized separately for each region based on cross-validation (Figure S2E). However, the decoding accuracy was robust with respect to a wide range of BW values (10–40 μV). Generally, in the absence of compression (i.e., zero compression threshold), the use of a larger BW parameter resulted in a slower decoding speed, and this BW-speed relationship was more pronounced in CPU than in GPU (Figure S2F).

Furthermore, we tested our approach using tetrode recordings of rat anterior dorsal thalamus (ATN) while navigating in a circular maze. The ATN is a central component of Papez's circuit and a key neural circuit supporting memory and spatial navigation (Jankowski et al., 2013). A large fraction of neurons in the ATN are tuned to the animal's head orientation and are termed head direction cells. We evaluated the representation power by computing the decoding error separately for the head direction and position based on unsorted thalamic ensemble spikes. Our results showed that the MUA of anterior thalamus contained good representations for head direction (median decoding error: 8.15°) and spatial position (median decoding error: 19.23 cm; Figure 1D). The decoding accuracy of head direction was not dependent on the run velocity threshold, but the position decoding error was. The relatively larger decoding error in position stems from the fact that a given head orientation was linked to two positions as the rat alternated running in clockwise and counterclockwise trajectories (Figures S2G–S2I). In all tetrode recordings (datasets 1–6), our GPU-powered approach could easily handle ultrafast per-spike decoding analyses of unsorted hippocampal or cortical ensemble spike activity during run behavior, which scales more favorably with the number of training samples than the CPU implementation.

In addition to tetrode arrays, custom high-density silicon probes have been widely used in rodent recordings (Berényi et al., 2014). We further tested our approach on a large-scale rat hippocampal recording based on two 64-channel silicon probes placed in the left and right hippocampi (Figure 2A). We selected feature vectors of various dimensions by varying the number of channels (1–10) per shank and the number of principal components (1–3) per channel during encoding. We assessed the decoding accuracy under different channel or feature combinations and observed a robust decoding performance (median decoding error: 7.6 cm; Figures S3A and S3B). Due to high redundancy of spike waveform features, a low-dimensional feature vector was sufficient to produce good decoding accuracy (Figure 2B). Notably, splitting the channels according to their spatial sites within a single shank yielded slightly degraded decoding accuracy (median error: 7.9 cm; one-sided Wilcoxon signed rank test; p = 0.405). Further increasing the feature dimension (up to 20) did not improve decoding performance. In the presence of large training sample size (>126,000 spikes in all shanks), increasing the feature dimension gradually improved the decoding accuracy, yet the GPU-powered decoding displayed a marked speed advantage (Figure 2C). In addition, employing kernel compression resulted in a further speedup of decoding.

GPU-Powered Decoding Scales up to Thousands of Channels

Our GPU-powered decoding system scales up to accommodate thousands of channels. To test the scalability, we replicated the silicon probe dataset to increase the channel count up to 2,000 and repeated the decoding analysis for RUN and SWS periods (Figure 2D; Table S2). By further optimizing GPU programming and the memory access strategy, the time required for decoding was well within the duration of the 250-ms time bin for RUN for all tested channels counts. For the more demanding case of decoding smaller time bins in SWS, decode time was within 20 ms bin duration for up to 1,200 channels (assuming no compression in the encoding model). Furthermore, for a fixed number of channels, the time needed for decoding increased much faster for CPU compared to GPU implementation as a function of the number of training samples (Figure 2E). As a result, in a typical one or two-dimensional spatial environment (with 50~800 spatial bins), our GPU-powered decoding system offers a highly scalable solution for accommodating 4–10 spike feature dimensions, tens to hundreds of thousands of training samples, and hundreds or even thousands of channels. A multi-GPU implementation can be readily deployed to accommodate a higher number of electrode channels.

Real-Time Decoding and Assessment of Memory Replay Events

Neural decoding approaches are particularly powerful for the identification of post-experience reactivated ensemble spiking patterns, such as hippocampal memory replay events. To reveal the causal contribution of replay events to learning and behavior, real-time decoding coupled to closed-loop feedback triggered by the occurrence of specific replay events is required. To illustrate how our method can be applied in this scenario, we decoded the unsorted hippocampal ensemble activity during 741 memory replay candidate events in SWS (dataset 7; Liu et al., 2018), where the percentage of active sorted CA1 units was small (~10%–15%; Chen and Wilson, 2017). Decoding unsorted MUA directly may maximize the usage of spiking data and improve the decoding accuracy in the presence of sparse ensemble spiking activity (Kloosterman et al., 2014). For a fair comparison, we only used the sorted spikes for unsorted decoding analysis. Compared to the standard likelihood-based decoding method using offline sorted ensemble spikes, we observed a trend toward improved reconstructed spatial trajectories for the replay candidate events (Figures 3A and S3C) and their associated significance statistics (Figure 3B). The analysis held for both SWS and quiet wakefulness (QW) conditions, as well as for tetrode and silicon probe data (Figures 3C and 3D). We also observed similar findings in decoding and replay analyses from another rat recording (dataset 8; Figures 3F and S3D).

As shown above, the GPU-based approach is sufficiently fast to decode ensembles recorded on a large number of electrodes at a fine timescale needed to detect replay events in real time. However, it is also desirable to assess the statistical significance of replayed neuronal ensemble representations online, either to provide feedback to the experimenter or as part of closed-loop perturbation experiments. Conventionally, the assessment of significance relies on computationally intensive operations using repeated independent Monte Carlo shuffles in offline analysis (Davidson et al., 2009; Liu et al., 2018). Based on our GPU-powered decoding system, we executed joint random shuffling operations (“shank or tetrode shuffle” and “spike time shuffle”). Upon decoding a typical candidate event of 200–300 ms duration with 1–30 spikes in each time bin, the computation to achieve 1,000 shuffling operations is ultrafast (<20 ms computation time or within the interval of next time bin) in both tetrode and silicon probe settings (Figures 3C–3F). Thus, our approach provides a feasible solution for online statistical assessment of decoded memory reactivation events within a short time delay.

Our approach provides an efficient solution to problem of online identification and real-time assessment of hippocampal memory replay. First, the hippocampal candidate replay event was detected based on the hippocampal MUA and a predetermined threshold (Figure 4A). Next, starting from the determined candidate event onset, the spatial position was reconstructed from unsorted ensemble spike activity at each time bin, and the ongoing “spatial trajectory” was assessed based on a weighted distance correlation metric (Liu et al., 2018) using online shuffling statistics (Figure 4B). Based on the derived p value, a cumulative score assessment was updated in time (Figure 4C). Once the cumulative score was above a predetermined threshold, the memory candidate event was deemed statistically significant. For each time bin, the computation time for statistical assessment was ~5 ms (Figure 4D). The identification latency (from the first time point that crossed our MUA threshold) for online significance assessment of replay events was ~10 to 11 bins (mean ± SEM: 208.0 ± 5.3 ms for dataset 7; 221.8 ± 6.6 ms for dataset 8) In our illustrations based on the predetermined threshold, the identified significant hippocampal replay events might differ between the offline and online assessment methods (Table S3; Figure S4). Two factors might contribute to this discrepancy. First, the significance criterion was based on a single Monte Carlo p value assessment in an offline setting but based on a cumulative score in an online setting instead; the choice of cumulative score threshold also affected the significance criterion. Second, offline assessment was evaluated on the complete period of a candidate event, whereas online assessment was evaluated on a shorter period of the candidate event. For the two investigated hippocampal datasets (datasets 7 and 8), the statistics varied and no consistent trend was found.

DISCUSSION

We have presented a GPU-powered system that provides an ultrafast and accurate readout of unsorted place codes from single or multiple brain regions. On a quad-core CPU-powered

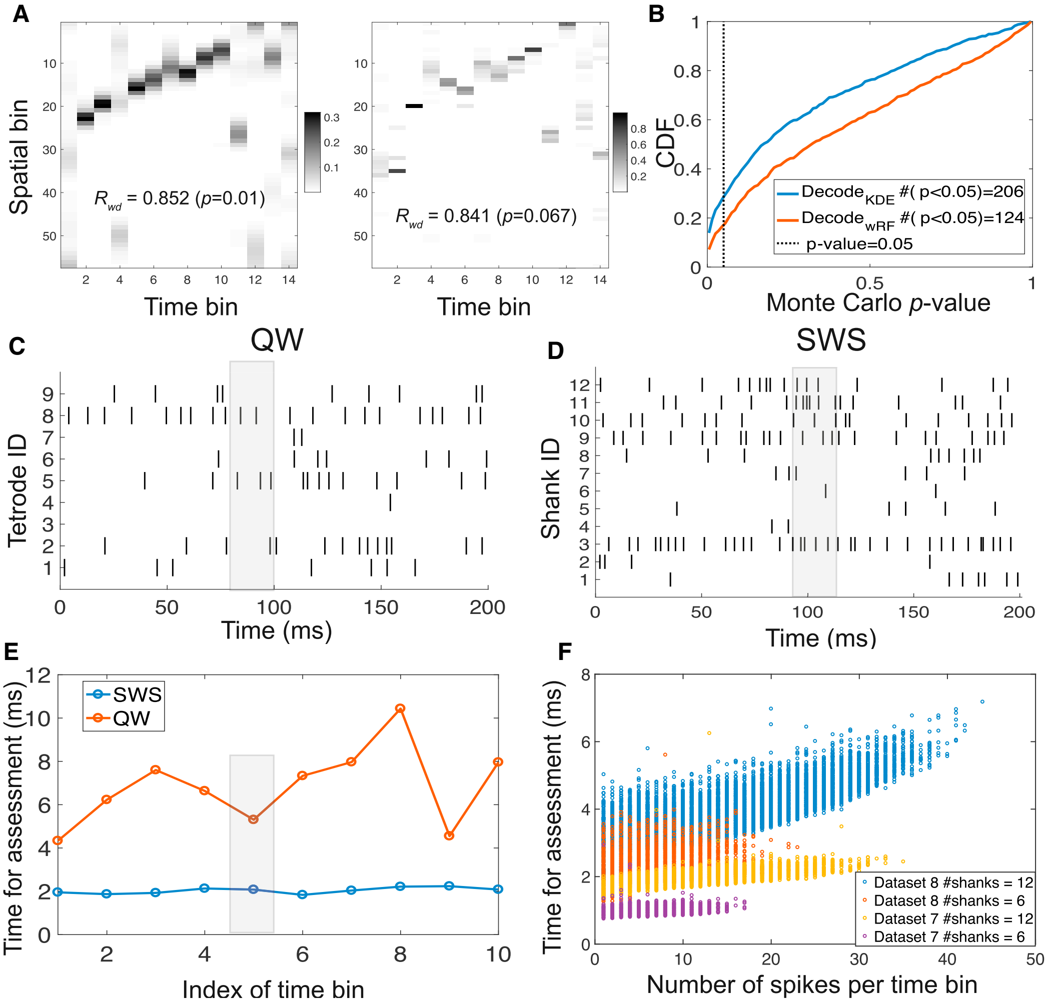


Figure 3. Decoding Memory Replay Events and Ultrafast Assessment of Statistical Significance

(A) Decoded memory reactivation events during post-SWS (dataset 7) derived from unsorted (left, R_{wd} = 0.852; Monte Carlo p = 0.01) and sorted (right, R_{wd} = 0.841; Monte Carlo p = 0.067) CA1 ensemble spike activity. Color bar represents the posterior probability, and dark pixel indicates high probability. Horizontal axis represents time bin (20 ms bin size), and vertical axis represents the linearized spatial bin (see more examples in Figure S3C). The significance results of two decoded trajectories were different, although their derived statistics R_{wd} were similar.

(B) CDF curves of Monte Carlo p value derived from significance testing of 741 hippocampal memory replay candidates during post-SWS. Compared to the standard decoding analysis based on place receptive fields of sorted units (Decode$_{\text{wRF}}$), our proposed method (Decode$_{\text{KDE}}$) identified more significant events for memory replay, suggesting an enhanced detection ability of Decode$_{\text{KDE}}$ based on the unsorted hippocampal ensemble spike activity.

(C) Unsorted rat hippocampal ensemble spikes from 9 tetrodes (dataset 2) in a memory candidate event during quiet wakefulness (QW). Shaded area marks a 20-ms bin.

(D) Unsorted rat hippocampal ensemble spikes from 12 silicon probe shanks (dataset 7) in a memory candidate event during SWS.

(E) Computation time needed for statistical assessment for each 20-ms time bin of the two examples shown in (C) and (D). The numbers of KDE components were 397,493 (with compression threshold 2) for dataset 2 and 126,624 (with compression threshold 0) for dataset 7. The number of random shuffles was 1,000.

(F) Computation time of statistical assessment varied with respect to the number of shanks (12 or 6) and the number of spikes per time bin. Each symbol represents the result derived from a single time bin (total number of tested bins: n = 9,614 for 12 shanks and n = 9,261 for 6 shanks in dataset 7; n = 14,924 for 12 shanks and n = 14,386 for 6 shanks in dataset 8). The computation time mainly depended on the number of shanks and the number of KDE components.

See also Figure S3.

PC, our decoding implementation outperforms the optimized multi-threaded CPU-based decoding significantly in both speed and scaling by leveraging multiple levels of parallelization. In addition to accelerating offline analyses of large datasets, the GPU-based system enables online decoding and significance assessment of ensemble spiking patterns for immediate feedback to the experimenter and to provide opportunity for content-based closed-loop experimental manipulation.

Spatial representation plays an important role in spatial navigation, sensorimotor integration, and decision-making tasks.

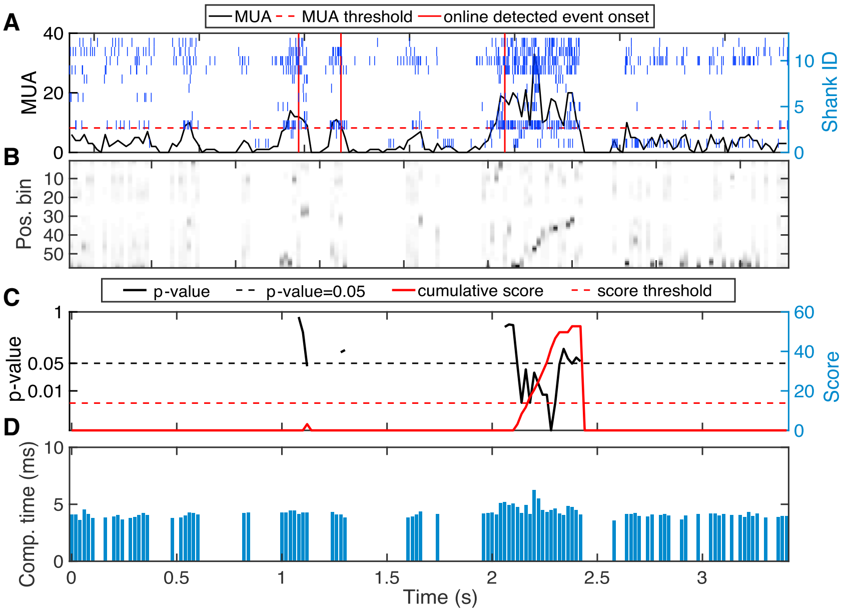


Figure 4. Online Identification and Assessment of Hippocampal Memory Replay

(A) Online event detection analysis: unsorted hippocampal ensemble spikes and replay burst detection based on the hippocampal MUA and a predetermined threshold (horizontal dashed line). The marked replay onset (vertical lines) was identified after three consecutive time bins that crossed the threshold.

(B) Starting from the candidate event onset, spatial position was reconstructed from unsorted ensemble spike activity at each time bin (20 ms). The ongoing decoded "spatial trajectory" was assessed based on the weighted distance correlation using online shuffling statistics.

(C) The p value for the online-evaluated replay (black). An accumulative score (red) was computed as the assessment was continuously updated. Finally, a decision was made for online experimental manipulation or intervention based on the accumulative score. The accumulative score was set to 0 at the detection onset and reset to 0 when the cumulative score threshold was reached.

(D) Computation time for evaluation at each time bin. The computation time includes both position decoding and statistical assessment involving both CPU and GPU resources. In this illustrated example, the statistical assessment time was nearly negligible compared to the decode time.

See also Figure S4.

Uncovering representations of place code in a state-dependent or content-specific neurofeedback provides valuable clues for closed-loop experimental manipulation during memory reactivations at the millisecond timescale (Grosenick et al., 2015; El Hady, 2016; Rothschild et al., 2017; Ciliberti et al., 2018). In addition, they can provide valuable input on the quality of sampled spatial representations during the early phase of experimental recording—for instance, whether the unit yield in targeted brain regions is sufficient or whether it is time to adapt the strategy and adjust the electrode placing.

To date, newly developed high-density electrode arrays have recently allowed us to simultaneously record large-scale ensemble spike activity from multiple brain regions (Jun et al., 2017a; Chung et al., 2018). However, population-decoding approaches based on real-time sorted spikes may have serious limitations in speed, scale, and accuracy. In contrast, decoding unsorted ensemble spike activity directly in our implementation is appealing for a wide range of applications (Ventura, 2008; Bansal et al., 2012; Kloosterman et al., 2014; Todorova et al., 2014; Ventura and Todorova, 2015). More importantly, our kernel density estimation (KDE)-based population-decoding analysis can be applied to various spatial or behavioral correlates, including the spatial position and head direction (Cho and Sharp, 2001; Peyrache et al., 2015; Jacob et al., 2017), as well as many other brain regions, such as the retrosplenial cortex, entorhinal cortex, and lateral septum.

To conclude, this open-source GPU-based neural decoding toolkit will expand opportunities for closed-loop rodent BMI systems to probe causal mechanisms of targeted neural circuits and to investigate memory processing across distributed brain circuits during various task behaviors (Girardeau et al., 2009; Ego-Stengel and Wilson, 2010; Jadhav et al., 2012; Roux et al., 2017).

STAR★METHODS

Detailed methods are provided in the online version of this paper and include the following:

- KEY RESOURCES TABLE
- CONTACT FOR REAGENT AND RESOURCE SHARING
- EXPERIMENTAL MODEL AND SUBJECT DETAILS
  - ○ Datasets 1-3: Rat CA1 tetrode recording
  - ○ Datasets 4 and 5: Rat CA1-V1 tetrode recording
  - ○ Datasets 6: Rat anterior dorsal thalamus tetrode recording
  - ○ Datasets 7 and 8: Rat CA1 silicon probe recording
- METHOD DETAILS
  - ○ Decoding unsorted neural ensemble spikes
  - ○ Spike waveform feature selection
  - ○ Offline identification of sleep replay candidate events
  - ○ GPU architecture and optimization
  - ○ GPU implementation for online statistical assessment of memory replay events
  - ○ Interface with the GPU code
- QUANTIFICATION AND STATISTICAL ANALYSIS
  - ○ Statistical tests
  - ○ Assessment of decoding performance
- DATA AND SOFTWARE AVAILABILITY

SUPPLEMENTAL INFORMATION

Supplemental Information includes six figures, four tables, and one video and can be found with this article online at https://doi.org/10.1016/j.celrep.2018.11.033.

ACKNOWLEDGMENTS

We thank Shizhao Liu for assisting the hippocampal replay analysis. This work was supported by the US National Science Foundation (NSF) grants IIS-130764 (Z.C. and M.A.W.) and CBET-1835000 (Z.C.); NIH grants R01-NS100065 (Z.C.), R01-MH118928 (Z.C. and M.A.W.), R01-MH106552 (D.J.), R01-MH107396 (G.B.), U01-NS090583 (G.B.), U01-NS099705 (G.B.), TR01-GM104948 (M.A.W.), R01-MH092638 (M.A.W.), and R21-EY028381 (M.A.W.); Flemish Research Project FWO G0D7516N (F.K.); National Science Foundation of China no. 31627802 (S.H.); Simons Foundations (G.B. and M.A.W.); and the Center for Brains, Minds and Machines, funded by the NSF STC award CCF-1231216 (M.A.W.). The following ORCIDs apply to the authors: 0000-0002-6483-6056 (Z.C.), 0000-0001-6680-9660 (F.K.), 0000-0002-3828-9618 (S.H.), 0000-0003-4115-5888 (D.J.), and 0000-0002-3100-4800 (G.B.).

AUTHOR CONTRIBUTIONS

Z.C. and F.K. conceived and designed the experiment. S.H. and D.C. wrote the software. S.H. and Z.C. conducted the analysis. A.D.G., F.M., D.J., H.P., and F.K. collected the experimental data. M.A.W., G.B., and F.K. supervised the experimental projects. Z.C. wrote the manuscript, and all authors contributed to editing. Z.C. and F.K. have contributed equally and are the senior authors.

DECLARATION OF INTERESTS

The authors declare no competing interests.

Received: July 13, 2018
Revised: October 10, 2018
Accepted: November 6, 2018
Published: December 4, 2018

REFERENCES

Bansal, A.K., Truccolo, W., Vargas-Irwin, C.E., and Donoghue, J.P. (2012). Decoding 3D reach and grasp from hybrid signals in motor and premotor cortices: spikes, multiunit activity, and local field potentials. J. Neurophysiol. *107*, 1337–1355.

Berényi, A., Somogyvári, Z., Nagy, A.J., Roux, L., Long, J.D., Fujisawa, S., Stark, E., Leonardo, A., Harris, T.D., and Buzsáki, G. (2014). Large-scale, high-density (up to 512 channels) recording of local circuits in behaving animals. J. Neurophysiol. *111*, 1132–1149.

Buzsáki, G., Stark, E., Berényi, A., Khodagholy, D., Kipke, D.R., Yoon, E., and Wise, K.D. (2015). Tools for probing local circuits: high-density silicon probes combined with optogenetics. Neuron *86*, 92–105.

Chen, Z., and Wilson, M.A. (2017). Deciphering neural codes of memory during sleep. Trends Neurosci. *40*, 260–275.

Chen, Z., Kloosterman, F., Layton, S., and Wilson, M.A. (2012). Transductive neural decoding for unsorted neuronal spikes of rat hippocampus. Conf. Proc. IEEE Eng. Med. Biol. Soc. *2012*, 1310–1313.

Chen, Z., Grosmark, A.D., Penagos, H., and Wilson, M.A. (2016). Uncovering representations of sleep-associated hippocampal ensemble spike activity. Sci. Rep. *6*, 32193.

Cho, J., and Sharp, P.E. (2001). Head direction, place, and movement correlates for cells in the rat retrosplenial cortex. Behav. Neurosci. *115*, 3–25.

Chung, J.E., Joo, H.R., Fan, J.L., Liu, D.F., Barnett, A.H., Chen, S., Geaghan-Breiner, C., Karlsson, M.P., Lee, K.Y., Liang, H., et al. (2018). A polymer probe-based system for high density, long-lasting electrophysiological recordings across distributed neuronal circuits. bioRxiv. https://doi.org/10.1101/242693.

Ciliberti, D., and Kloosterman, F. (2017). Falcon: a highly flexible open-source software for closed-loop neuroscience. J. Neural Eng. *14*, 045004.

Ciliberti, D., Michon, F., and Kloosterman, F. (2018). Real-time classification of experience-related ensemble spiking patterns for closed-loop applications. eLife *7*, e36275.

Davidson, T.J., Kloosterman, F., and Wilson, M.A. (2009). Hippocampal replay of extended experience. Neuron *63*, 497–507.

Deng, X., Liu, D.F., Kay, K., Frank, L.M., and Eden, U.T. (2015). Clusterless decoding of position from multiunit activity using a marked point process filter. Neural Comput. *27*, 1438–1460.

Deng, X., Liu, D.F., Karlsson, M.P., Frank, L.M., and Eden, U.T. (2016). Rapid classification of hippocampal replay content for real-time applications. J. Neurophysiol. *116*, 2221–2235.

Ego-Stengel, V., and Wilson, M.A. (2010). Disruption of ripple-associated hippocampal activity during rest impairs spatial learning in the rat. Hippocampus *20*, 1–10.

El Hady, A., ed. (2016). Closed Loop Neuroscience (Academic Press).

Fischer, J., Milekovic, T., Schneider, G., and Mehring, C. (2014). Low-latency multi-threaded processing of neuronal signals for brain-computer interfaces. Front. Neuroeng. *7*, 1.

Girardeau, G., Benchenane, K., Wiener, S.I., Buzsáki, G., and Zugaro, M.B. (2009). Selective suppression of hippocampal ripples impairs spatial memory. Nat. Neurosci. *12*, 1222–1223.

Gomperts, S.N., Kloosterman, F., and Wilson, M.A. (2015). VTA neurons coordinate with the hippocampal reactivation of spatial experience. eLife *4*, e05360.

Grosenick, L., Marshel, J.H., and Deisseroth, K. (2015). Closed-loop and activity-guided optogenetic control. Neuron *86*, 106–139.

Grosmark, A.D., and Buzsáki, G. (2016). Diversity in neural firing dynamics supports both rigid and learned hippocampal sequences. Science *351*, 1440–1443.

Hafting, T., Fyhn, M., Molden, S., Moser, M.-B., and Moser, E.-I. (2005). Microstructure of a spatial map in the entorhinal cortex. Nature *436*, 801–806.

Haggerty, D.C., and Ji, D. (2015). Activities of visual cortical and hippocampal neurons co-fluctuate in freely moving rats during spatial behavior. eLife *4*, e08902.

Jacob, P.-Y., Casali, G., Spieser, L., Page, H., Overington, D., and Jeffery, K. (2017). An independent, landmark-dominated head-direction signal in dysgranular retrosplenial cortex. Nat. Neurosci. *20*, 173–175.

Jadhav, S.P., Kemere, C., German, P.W., and Frank, L.M. (2012). Awake hippocampal sharp-wave ripples support spatial memory. Science *336*, 1454–1458.

Jankowski, M.M., Ronnqvist, K.C., Tsanov, M., Vann, S.D., Wright, N.F., Erichsen, J.T., Aggleton, J.P., and O'Mara, S.M. (2013). The anterior thalamus provides a subcortical circuit supporting memory and spatial navigation. Front. Syst. Neurosci. *7*, 45.

Ji, D., and Wilson, M.A. (2007). Coordinated memory replay in the visual cortex and hippocampus during sleep. Nat. Neurosci. *10*, 100–107.

Jun, J.J., Steinmetz, N.A., Siegle, J.H., Denman, D.J., Bauza, M., Barbarits, B., Lee, A.K., Anastassiou, C.A., Andrei, A., Aydın, Ç., et al. (2017a). Fully integrated silicon probes for high-density recording of neural activity. Nature *551*, 232–236.

Jun, J.J., Mitelut, C., Lai, C., Gratiy, S., Anastassioiu, C., and Harris, T.D. (2017b). Real-time spike sorting platform for high-density extracellular probes with ground-truth validation and drift correction. bioRxiv. https://doi.org/10.1101/101030.

Kloosterman, F., Layton, S.P., Chen, Z., and Wilson, M.A. (2014). Bayesian decoding using unsorted spikes in the rat hippocampus. J. Neurophysiol. *111*, 217–227.

Liu, S., Grosmark, A.D., and Chen, Z. (2018). Methods for assessment of memory reactivation. Neural Comput. *30*, 2175–2209.

Mao, D., Kandler, S., McNaughton, B.L., and Bonin, V. (2017). Sparse orthogonal population representation of spatial context in the retrosplenial cortex. Nat. Commun. *8*, 243.

Mao, D., Neurmann, A.R., Sun, J., Bonin, V., Mohajerani, M.H., and McNaughton, B.L. (2018). Hippocampus-dependent emergence of spatial sequence coding in retrosplenial cortex. Proc. Natl. Acad. Sci. USA *115*, 8015–8018.

Michon, F., Aarts, A., Holzhammer, T., Ruther, P., Borghs, G., McNaughton, B., and Kloosterman, F. (2016). Integration of silicon-based neural probes and micro-drive arrays for chronic recording of large populations of neurons in behaving animals. J. Neural Eng. *13*, 046018.

O'Keefe, J., and Dostrovsky, J. (1971). The hippocampus as a spatial map. Preliminary evidence from unit activity in the freely-moving rat. Brain Res. *34*, 171–175.

Peyrache, A., Lacroix, M.M., Petersen, P.C., and Buzsáki, G. (2015). Internally organized mechanisms of the head direction sense. Nat. Neurosci. *18*, 569–575.

Pfeiffer, B.E., and Foster, D.J. (2013). Hippocampal place-cell sequences depict future paths to remembered goals. Nature *497*, 74–79.

Rios, G., Lubenov, E.V., Chi, D., Roukes, M.L., and Siapas, A.G. (2016). Nanofabricated neural probes for dense 3-D recordings of brain activity. Nano Lett. *16*, 6857–6862.

Rossant, C., Kadir, S.N., Goodman, D.F.M., Schulman, J., Hunter, M.L.D., Saleem, A.B., Grosmark, A., Belluscio, M., Denfield, G.H., Ecker, A.S., et al. (2016). Spike sorting for large, dense electrode arrays. Nat. Neurosci. *19*, 634–641.

Rothschild, G., Eban, E., and Frank, L.M. (2017). A cortical-hippocampal-cortical loop of information processing during memory consolidation. Nat. Neurosci. *20*, 251–259.

Roumis, D.K., and Frank, L.M. (2015). Hippocampal sharp-wave ripples in waking and sleeping states. Curr. Opin. Neurobiol. *35*, 6–12.

Roux, L., Hu, B., Eichler, R., Stark, E., and Buzsáki, G. (2017). Sharp wave ripples during learning stabilize the hippocampal spatial map. Nat. Neurosci. *20*, 845–853.

Shobe, J.L., Claar, L.D., Parhami, S., Bakhurin, K.I., and Masmanidis, S.C. (2015). Brain activity mapping at multiple scales with silicon microprobes containing 1,024 electrodes. J. Neurophysiol. *114*, 2043–2052.

Sodkomkham, D., Ciliberti, D., Wilson, M.A., Fukui, K., Moriyama, K., Numao, M., and Kloosterman, F. (2016). Kernel density compression for real-time Bayesian encoding/decoding of unsorted hippocampal spikes. Knowl. Base. Syst. *94*, 1–12.

Todorova, S., Sadtler, P., Batista, A., Chase, S., and Ventura, V. (2014). To sort or not to sort: the impact of spike-sorting on neural decoding performance. J. Neural Eng. *11*, 056005.

Tsai, D., Sawyer, D., Bradd, A., Yuste, R., and Shepard, K.L. (2017). A very large-scale microelectrode array for cellular-resolution electrophysiology. Nat. Commun. *8*, 1802.

Ventura, V. (2008). Spike train decoding without spike sorting. Neural Comput. *20*, 923–963.

Ventura, V., and Todorova, S. (2015). A computationally efficient method for incorporating spike waveform information into decoding algorithms. Neural Comput. *27*, 1033–1050.

Whitlock, J.R., Sutherland, R.J., Witter, M.P., Moser, M.-B., and Moser, E.-I. (2008). Navigating from hippocampus to parietal cortex. Proc. Natl. Acad. Sci. USA *105*, 14755–14762.

STAR★METHODS

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited Data		
Datasets 2, 4, 5	https://github.com/wilsonlab/CRCNS_Shared_Data	N/A
Datasets 7, 8	https://crcns.org/data-sets/hc/hc-11	N/A
Experimental Models: Organisms/Strains		
Long-Evans rats	Charles River Labs	RRID: RGD_2308852
Software and Algorithms		
MATLAB	MathWorks	N/A
CUDA	Nvidia	N/A
PYTHON	Open source	N/A
Custom code for KDE decoding and GPU implementation	https://github.com/yuehusile/real_time_read_out_GPU	N/A

CONTACT FOR REAGENT AND RESOURCE SHARING

Further information and requests for reagent and resource should be directed to and will be fulfilled by the Lead Contact, Zhe Chen (zhe.chen@nyulangone.org).

EXPERIMENTAL MODEL AND SUBJECT DETAILS

We have used eight rat recordings collected from three laboratories based on various multielectrode arrays. A summary of experimental datasets is shown in Table S1.

Datasets 1-3: Rat CA1 tetrode recording

Young adult Long-Evans rats were running in T-shaped 3-arm maze (~6.4 m), linear track (~10 m) and open field (~1.8 m diameter) environment (Figure S1C). Custom tetrode arrays were implanted to record neural ensemble spike activity from the dorsal hippocampal CA1 of freely behaving animals. The 6.4-m and 10-m track datasets consisted of 15 and 9 implanted tetrodes, respectively; and the open field dataset consisted of 9 implanted tetrodes. In the first dataset, the animal protocol was approved by the NERF Committee on Animal Care. In the remaining two datasets, the animal protocol was approved by the Massachusetts Institute of Technology (MIT) Committee on Animal Care and followed the National Institutes of Health (NIH) guidelines. Technical details are referred to a previous publication (Davidson et al., 2009).

Datasets 4 and 5: Rat CA1-V1 tetrode recording

Young adult Long-Evans rats were trained to run an alternation task on a ‘Figure 8’-shaped maze (~4.7 m track length, Figure S1C). Animals learned to alternate two trajectories (LR: from the left reward site L to the right reward site R, and RL: from R to L) via a central track on a figure-‘8’ for food reward. The maze was placed inside a dark curtain without obvious distal visual cues except for the irregular wrinkles on its wall, but with various local visual cues, mainly stripes with different orientations and simple geometric shapes, on the maze floors and walls. The animal training and recording protocols were approved by the MIT Committee on Animal Care and followed the NIH guidelines.

After the animal reached equal or greater than 80% accuracy, a custom tetrode array was implanted to record multiple single units simultaneously from the dorsal hippocampus CA1 and the visual cortex (deep layers: L5/6). In Dataset 4, ten tetrodes were located in visual cortex (8 in V1, 2 in V2; AP −7.3 relative to Bregma, ML 3.5 relative to midline) and three tetrodes were located in CA1 (AP −3.8 relative to Bregma, ML 2.2 relative to midline). In Dataset 5, four tetrodes were located in V1 and four tetrodes were located in CA1. Details of experimental protocols and data have been published (Ji and Wilson, 2007).

Datasets 6: Rat anterior dorsal thalamus tetrode recording

A Long-Evans rat was running back and forth on a circular maze (0.61 m radius; Figure S1C). The maze had a wall divider that defined start and end points used for reward delivery as follows: the rat was initially placed at the start location and, upon reaching the end location, the animal received a small amount of liquid chocolate reward (~0.1 mL). Then the rat turned around and when it reached the start location, it received additional liquid chocolate reward. This behavior went on for about 17 minutes. Seven tetrodes in a circular bundle were aimed at the anterior dorsal thalamus (−2.1 mm AP, 1.3 mm ML, relative to Bregma) and were positioned to maximize the detection of spiking activity. Position and head direction were monitored through an overhead camera (30 Hz sampling rate) and a pair of colored LEDs mounted on the headstage of the rat. The animal protocol was approved by the Massachusetts Institute of Technology (MIT) Committee on Animal Care and followed the National Institutes of Health (NIH) guidelines.

Datasets 7 and 8: Rat CA1 silicon probe recording

Male Long-Evans rats were bilaterally implanted with two 6-shank silicon probes parallel to the septo-temporal axis of the left and right dorsal hippocampi, totaling 128 channels. Each shank of the 6-shank silicon probes had 10 sites (or channels). All sites were vertically staggered along the shank with 20 μm spacing between sites. We selected one recording session of rat ('Achilles') on November 1, 2013. The recording session consisted of a long (~4 hr) pre-RUN sleep epoch in a familiar room, followed by a RUN epoch (~45 minutes) in a novel circular maze (1 m diameter, Dataset 6) or linear track (1.6 m, Dataset 7; Figure S1C). After the RUN epoch the animal was transferred back to its home cage in the familiar room where another long (~4 hour) post-RUN sleep was recorded. The protocol was approved by the Institutional Animal Care and Use Committee of New York University School of Medicine. Details of the experimental protocol and data have been published (Grosmark and Buzsáki, 2016; Chen et al., 2016). The electrophysiological data are publicly available (https://crcns.org/data-sets/hc/hc-11/).

METHOD DETAILS

Decoding unsorted neural ensemble spikes

Our population decoding analysis consists of encoding and decoding phases (Figure S1A), and neither phase requires spike sorting. The essential operation of the encoding phase is to estimate a joint probability density function (pdf) using nonparametric or semi-parametric density estimation methods (Chen et al., 2012; Kloosterman et al., 2014). Let $\boldsymbol{x}$ denote the 1D or 2D spatial position, and let $\boldsymbol{a}$ denote the feature vector that is associated with each spike. In the case of tetrode recording, we used the spike peak amplitude of four channels; whereas in the case of silicon probe, we used the spike waveform principal components (PCs) of each channel. Specifically, we represented the joint pdf $p(\boldsymbol{a},\boldsymbol{x})$ with a kernel density estimation (KDE):

$$p(\boldsymbol{a},\boldsymbol{x})=\frac{1}{N}\sum_{n=1}^{N}K_{H_{ax}}\left(\begin{bmatrix}\boldsymbol{a}\\ \boldsymbol{x}\end{bmatrix}-\begin{bmatrix}\tilde{\boldsymbol{a}}_n\\ \tilde{\boldsymbol{x}}_n\end{bmatrix}\right) \tag{1}$$

where $(\tilde{\boldsymbol{a}}_n,\tilde{\boldsymbol{x}}_n)$ denotes the n-th sample for d-dimensional variables $(\boldsymbol{a},\boldsymbol{x})$, N denotes the number of training samples, and $K_{H_{ax}}(\bullet)$ denotes the kernel function with a specific bandwidth (BW) parameter H_{ax}. In multielectrode recordings, individual tetrodes (or silicon probe shanks) were assumed to be mutually independent. At each tetrode (or shank), we ran a kernel compression algorithm to reduce redundancy in the training samples by progressively merging samples based on a compression threshold and updating the sample covariance matrix and weight accordingly (Figure S1B; Sodkomkham et al., 2016).

The compression threshold is defined as the Mahalanobis distance below which a new sample is merged with an existing sample. The threshold is lower bounded by 0: a zero compression threshold represents no compression, and an infinity compression threshold implies using only one sample in the limit. The higher the compression threshold, the fewer samples were used in KDE and encoding analysis. To reduce computational cost, we employed an isotropic Gaussian kernel $K_{H_{ax}}$ in KDE, and rewrote *Equation 1* as

$$p(\boldsymbol{a},\boldsymbol{x})=\sum_{n=1}^{N}s_n\exp(g_n(\boldsymbol{x})+h_n(\boldsymbol{a})) \tag{2}$$

$$s_n=\frac{w_n}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}|}} \tag{3}$$

$$g_n(\boldsymbol{x})=-\frac{1}{2}(\boldsymbol{x}-\tilde{\boldsymbol{x}}_n)^{\mathrm{T}}\sum\nolimits_{\boldsymbol{x}}^{-1}(\boldsymbol{x}-\tilde{\boldsymbol{x}}_n) \tag{4}$$

$$h_n(\boldsymbol{a})=-\frac{1}{2}(\boldsymbol{a}-\tilde{\boldsymbol{a}}_n)^{\mathrm{T}}\sum\nolimits_{\boldsymbol{a}}^{-1}(\boldsymbol{a}-\tilde{\boldsymbol{a}}_n) \tag{5}$$

where $d=d_a+d_x$ denotes the combined dimension of spike feature vector and spatial position. The $d\times d$ covariance matrix $\boldsymbol{\Sigma}=\begin{bmatrix}\sum\nolimits_{\boldsymbol{a}} & 0\\ 0 & \sum\nolimits_{\boldsymbol{x}}\end{bmatrix}$ defines the kernel BW. We assumed a strict diagonal BW structure in all KDE analyses. In total, $\{\tilde{\boldsymbol{a}}_n,\tilde{\boldsymbol{x}}_n,\boldsymbol{\Sigma}\}_{n=1}^{N}$ represent the set of N Gaussian components in KDE representations, and w_n is a nonnegative weight coefficient associated with the n-th component. If there is no sample compression, then $w_n = 1$ and N is the number of training samples. Given $\{\tilde{\boldsymbol{a}}_n, \tilde{\boldsymbol{x}}_n\}_{n=1}^N$ and predetermined stimulus vector $\boldsymbol{x} = \{\boldsymbol{x}_m\}_{m=1}^M$ (where M denotes the number of spatial bins), the scaling factor s_n and stimulus-dependent component $g_n(\boldsymbol{x})$ were pre-computed. In summary, the encoding-decoding algorithm (termed 'Decode$_{\mathrm{KDE}}$') consists of the following steps (Kloosterman et al., 2014; Sodkomkham et al., 2016)

Step 1: Compute $\boldsymbol{s}_n$ and $\boldsymbol{g}_n(\boldsymbol{x})$ according to *Equation 3* and *Equation 4* based on $\{\tilde{\boldsymbol{a}}_n, \tilde{\boldsymbol{x}}_n\}_{n=1}^{\boldsymbol{N}}$.

Step 2: Compute $\boldsymbol{h}_n(\boldsymbol{a})$ according to *Equation 5* for the feature $\boldsymbol{a}$ associated with the observed spike.

Step 3: Estimate the joint probability distribution $\boldsymbol{p}(\boldsymbol{a}, \boldsymbol{x})$ according to *Equation 2*.

Step 4: Compute the likelihood by accumulating the spikes collected from K independent tetrodes (or shanks). At any time interval [t, t+Δt), the likelihood is given by

$$Likelihood = \prod_{k=1}^{K} \left\{ (\Delta t)^{n_{k,t}} \left[\prod_{i=1}^{n_{k,t}} \lambda_k(\boldsymbol{a}_{k,i}, \boldsymbol{x}) \right] \left[e^{-\Delta t \lambda_k(\boldsymbol{x})} \right] \right\} \tag{6}$$

where $n_{k,t}$ denotes the number of spikes observed at the k-th tetrode (or shank) during the interval [t, t+Δt). In *Equation 6*, the generalized rate functions $\lambda_k(\boldsymbol{a}, \boldsymbol{x})$ and $\lambda_k(\boldsymbol{x})$ are defined by the density ratios

$$\lambda_k(\boldsymbol{a}, \boldsymbol{x}) = \mu_k \frac{p_k(\boldsymbol{a}, \boldsymbol{x})}{\pi(\boldsymbol{x})} \text{ and } \lambda_k(\boldsymbol{x}) = \mu_k \frac{p_k(\boldsymbol{x})}{\pi(\boldsymbol{x})},$$

where $p_k(\boldsymbol{a}, \boldsymbol{x})$ and $p_k(\boldsymbol{x})$ denote the joint and marginal pdfs derived from KDE at the k-th tetrode (or shank), respectively; μ_k denotes the mean firing rate at the k-th tetrode (or shank); and $\pi(\boldsymbol{x})$ denotes the spatial occupancy probability distribution estimated from KDE. Finally, we sought the decoded estimate of among all candidate positions $\{\boldsymbol{x}_m\}_{m=1}^M$ that produced the maximum likelihood in *Equation 6*. We used temporal bin size Δt = 250 ms during RUN and Δt = 20 ms during SWS.

In the KDE-based simulated online decoding analysis, *Step 1* was executed offline and reused in decoding. *Steps 2-4* were executed on the fly for each spike collected during online decoding. Since *Step 1* does not influence the online performance, we focused on the implementation and optimization of other steps. The input of the algorithm for a single tested spike included a pre-computed scaling factor array $\{s_n\}_{n=1}^N$, a stimulus-dependent component array $\{g_n(\boldsymbol{x})\}_{n=1}^N$, a total of N Gaussian components of spike feature, and the feature vector associated with the tested spike. We set a CUTOFF threshold to exclude out-of-range components.

Spike waveform feature selection

In tetrode recordings, we used the peak amplitude of spike waveform from each channel, yielding a 4-dimensional feature vector for each spike. The BW parameter was assumed identical for all four dimensions and optimized with grid search. We used an amplitude threshold (~80-100 μV) to remove putative 'noisy' spikes. In rat hippocampal CA1 recordings, we adapted a spike width threshold criterion to include or exclude putative interneurons. In Datasets 4 and 5, BW parameters were optimized separately for CA1 and V1 tetrode recordings (Figure S2D).

*In silico*n probe recordings, we conducted principal component analysis (PCA) on the spike waveforms for each channel, and extracted the first and second principal components (PCs) associated with the greatest variance. We considered two options to construct the feature vector in encoding analysis. The first and standard option used varying number (1-10) of channels in each shank, and each channel used one or two PCs, resulting a feature vector with 1-10 (or 2-20) dimensions. The second option used combinations of local neighboring sites in each shank (e.g., splitting 10 channels into two groups: channels 1-5 and channels 6-10), and treated the divided groups as independent tetrodes. In each combination, two PCs of each channel were used to construct a 10-dimensional feature vector. The second option was motivated by the fact that distinct units across cortical layers or cortical structures are spatially distributed. In both options, the BW parameter of each feature dimension was assumed identical.

Offline identification of sleep replay candidate events

In offline analysis, we used the electromyography (EMG) and LFP for sleep staging. SWS was primarily determined by the low EMG amplitude and high delta/theta power ratio in LFP activity. For screening the candidate events of memory replay during SWS, we used the hippocampal LFP ripple band (150-300 Hz) power combined with hippocampal MUA. In offline analysis, we selected 741 pre-identified candidate events (Dataset 7) and1015 candidate events (Dataset 8) during post-SWS epochs according to a previously established criterion (Liu et al., 2018). To assess the significance of decoded trajectories during SWS epochs, we computed the weighted distance correlation (R_{wd}) of decoded trajectory from each candidate event and the associated Monte Carlo *p-value* based on shuffled statistics (Liu et al., 2018). Two types of random shuffling operations were considered: one is tetrode (or shank) ID shuffle, and the other is spike time shuffle. A total of 1,000 independent random samples were used to compute the Monte Carlo *p-value*.

GPU architecture and optimization

The GPU implementation of our decoding algorithm was based on the parallel nature of KDE, and was further optimized to achieve the best performance by accounting for GPU features and custom optimization techniques. We designed a three-level hierarchical parallelism structure with the NVIDIA CUDA programming model and maximized the parallelization benefit, as shown in Figure S5A. The highest-level parallelism took advantage of the independence between tetrodes (or shanks), and the computation task for each tetrode (or shank) was assigned to a single CUDA stream. As such, the memory copy between GPU and CPU in one stream could be executed while the computation continued in a different stream, and we could minimize the majority of memory transfer delay. The medium and lowest-level parallelisms were based on the independence between the computation of each KDE component, each spike and each spatial bin; they were mapped to CUDA blocks and threads, respectively.

In CUDA applications, serial codes running on CPU is called ‘host codes’, and the parallel codes running on the GPU device are called ‘kernels’, which execute the same set of instructions on massive data that are mapped to blocks and threads (Figures S5B–S5D). Unlike the standard GPU implementation that uses a single kernel, we designed a two-kernel solution to avoid re-computing $h_n(\boldsymbol{a})$ (*Equation 5*) and to speed up computation. We implemented these operations with two CUDA kernels: *Kernel1* for calculating $h_n(\boldsymbol{a})$, each thread of the kernel calculates the h value of a single component, and *Kernel2* for calculating the joint pdf $p(\boldsymbol{a},\boldsymbol{x})$ based on the result of *Kernel1*. The pseudocodes of two kernel operations are summarized in ***Algorithm1*** and ***Algorithm2*** (Table S4).

Each thread in *Kernel1* computed $h_n(\boldsymbol{a})$ for a single component for one spike, the execution details are shown in ***Algorithm1.*** Computing for a single position of one spike required looping over all the components. To obtain an optimal performance, we divided these components into m non-overlapping subsets, and assigned related computations to different threads. As shown in ***Algorithm2***, each thread in *Kernel2* computed $p(\boldsymbol{a},\boldsymbol{x})$ for a single position with one spike. Note that both *Kernel1* and *Kernel2* could be launched for computing multiple spikes in parallel. In Figure S5A, we set *B1* and *B2* to be multiples of 32, which was equal to the number of threads in a GPU execution warp, in order to maximize the occupancy of GPU cores. Among other factors influencing the overall occupancy, we obtained the best performance by setting $B1 = B2 = 64$.

We also applied several optimization tricks to significantly boost the GPU speed. In *Kernel2*, we used the on-chip shared memory (as opposed to the off-chip DDR memory) to contain $h_n(\boldsymbol{a})$ and s_n, both of which were frequently accessed by every thread in this kernel. The shared memory is accessed over 10 times faster than the device memory. Consequently, moving frequently accessed data from the device memory to shared memory significantly reduced the memory access cost. This optimization was very effective when the corresponding memory cost was predominant.

However, the size of shared memory was often limited. When the number of components N was too large (e.g., n = 12K in float precision or 16K in double precision) to fit $h_n(\boldsymbol{a})$ and s_n in the shared memory (48 KB for our GPU), we divided the components into multiple partitions and computed each partition one by one in order to take advantage of the shared memory. The number of partitions and the size of each partition were determined based on the size of the shared memory. We also reduced the floating-point precision (using 2 bytes instead of 4 bytes) to store these data. As a result, the number of components that can be computed in one partition doubled. We further rescaled the data within a proper range to minimize the difference induced by precision conversion.

The computational cost of GPU-based decoding analysis mainly depended on four factors: the number of tetrodes (or shanks) K, the size of sample (or component) N used in KDE per tetrode (or shank), the dimensionality of feature vector d, and the number of spatial bins M. Among these factors, parallelization in the dimensions of K and M is straightforward. The order of decoding complexity is $\mathcal{O}(K*N*d*M)$ plus additional computational overhead for data processing. Depending on their relative sizes and bin size Δt, the main memory bottleneck called for different solutions or further optimization.

GPU implementation for online statistical assessment of memory replay events

An important step in analyzing memory replay of decoded spatial trajectories is their statistical assessment using measure that characterizes the spatial-temporal structure (Davidson et al., 2009). To assess the significance of the decoding result, we need to shuffle the timestamps and tetrode (or shank) ID of the collected spikes and compare the statistics of the decoding results derived from the shuffled spikes with those of derived from the raw spikes (Liu et al., 2018). The assessment was performed on the replay candidate event—-the set of continuous time bins that may contain the memory replay. The hippocampal replay candidate event (including the onset and offset) was determined by the hippocampal MUA (Figure 4A). Alternatively, we can assess the significance upon reaching a duration threshold for each candidate event (e.g., 80-100 ms). For each candidate event, the online assessment was performed progressively for each 20 ms of new data from the burst onset and consists of the following three steps:

Step 1: Generate the shuffled samples. Specifically, the shuffling of time stamps is realized by making each shuffled sample as a random subset of all the collected spikes from multiple time bins, and a random tetrode (or shank) ID is assigned to each spike in this subset. The size of the subset is set to the number of raw spikes in current time bin.

Step 2: Evaluate the joint pdf $\boldsymbol{p}(\boldsymbol{a},\boldsymbol{x})$ according to KDE (*Equations 2–5*) for raw spikes and all the shuffled samples generated in Step 1, and then compute the likelihood of each shuffled sample (*Equation 6*) based on the evaluated $\boldsymbol{p}(\boldsymbol{a},\boldsymbol{x})$.

Step 3: Repeat *Step 1* and *Step 2*, while monitoring a criterion (e.g., sufficient time bins collected after a detected MUA burst onset) for triggering the assessment. When the trigger criterion is met, conduct the assessment based on the likelihood statistics from Step 2 and other statistical criteria (e.g., weighted distance correlation; Liu et al., 2018).

Assuming that the number of shuffled samples is S, we have to run the decoding analysis (*Step 2*) of the same time bin by S times (typically $S \geq 1{,}000$). Running KDE decoding analyses more than 1,000 times in real time is a challenging task even with GPU, especially in the case of small time bin (~20 ms) during memory replay events. Here, we proposed a computationally efficient solution to significantly reduce the computation load based on the following two observations: (i) The evaluation of $p(\boldsymbol{a}, \boldsymbol{x})$ for each spike in *Step 2* only relies on the spike features and the tetrode (or shank) ID that each spike has been assigned to. For each spike, K different evaluation results are derived from K possible tetrode (or shank) IDs. (ii) When S is large, there is a high probability that many spike-ID pairs exist in multiple shuffled samples, resulting in a considerable amount of repeated evaluations of $p(\boldsymbol{a},\boldsymbol{x})$. Considering the fact that $K \ll S$, we can simply run all possible evaluations of $p(\boldsymbol{a}, \boldsymbol{x})$ for every spike and reuse these results based on the spike-ID pair in each shuffled sample.

In light of these two observations, our online shuffling method is described as follows (Figure S6): Within each time bin, we evaluated $p(\boldsymbol{a}, \boldsymbol{x})$ for each spike with all possible IDs and saved these results sequentially in a pre-allocated buffer space. When a sufficient number of results were buffered, we executed *Step 1* in an alternative way: the shuffled samples were determined by the randomly shuffled indices for selecting different subsets of buffered results. Since the indices were stored across time bins in the buffer, the time and tetrode (or shank) ID shuffling could be performed at the same time. In *Step 2*, the evaluation of the joint pdf $p(\boldsymbol{a}, \boldsymbol{x})$ was only carried out for raw spikes, and the computation of the likelihood of each shuffled sample was executed by reusing the buffered evaluation results based on the shuffled indices. However, generating shuffled indices remained time consuming. To resolve this issue, we used a constant set of indices instead of re-generating the shuffled indices online. Since the new results of each time bin were added to the beginning of the buffer, a constant set of indices still preserved different evaluation sets for different time bins.

We used the MUA and a predetermined threshold (e.g., mean+s.d.) to detect the onset of candidate event (Figure 4A). The onset was determined when there were 3 consecutive time bins (i.e., 60 ms) above the threshold. From the detected event onset, we run the unsorted decoding analysis for each time bin (with inclusion of the first 3 time bins that crossed the MUA threshold). From the decoded probability traces (Figure 4B), we then run the online shuffle (1,000 samples) and computed the *p-value* (from the weighted distance correlation) for statistical assessment (Figure 4C). In addition, we computed a cumulative score as follows

$$Score_i = \begin{cases} Score_{i-1}, & \text{if } MUA < 2 \text{ or } p_{value}(i) > 0.05 \\ Score_{i-1} - \log\left(p_{value(i)}\right), & \text{otherwise} \end{cases}$$

where i denotes the time bin index, and $Score_0 = 0$ at the time of event onset. Once the online cumulative score was above a predetermined threshold (e.g., $-3\log(0.01)$ used in our demonstration examples and current analyses), the event was deemed statistically significant and we reset the cumulative score to be 0. It is important to point out that, depending on the goal of online assessment, we could further optimize the detection and cumulative score thresholds to achieve a desired trade-off (speed versus accuracy).

Interface with the GPU code

We developed an interface for the GPU codes such that the algorithm was written in CUDA® C programming language, with C++ and Cython wrappers that could accommodate user configuration and GPU decoding from C++ and Python environments. All decoding analyses were run on a PC (Linux Ubuntu OS) with Intel Core i7-7700K CPU (quad core, 8 threads, 4.2 GHz, 32 GB DDR4 RAM). A single NVIDIA GeForce GTX 1080Ti graphics card with 11 GB memory was used for GPU-based computation.

QUANTIFICATION AND STATISTICAL ANALYSIS

Statistical tests

The *p-value* was derived from Monte Carlo shuffling statistics, and Monte Carlo $p < 0.05$ was considered to be statistically significant. Other statistical testing involved Mann-Whiteny test.

Assessment of decoding performance

In hippocampal CA1 recordings, we evaluated the decoding accuracy during RUN epochs (velocity threshold: 8.5 cm/s for Datasets 1-3; 5 cm/s for Datasets 4-5; 20 cm/s for Dataset 6, 15 cm/s for Dataset 7-8) by the absolute error between the animal's actual position and decoded position: $|\boldsymbol{x}_{\text{true}} - \widehat{\boldsymbol{x}}_{\text{decode}}|$. In the case of V1 decoding (Datasets 4 and 5), we imposed no velocity threshold. We split the recordings into training and testing data. We assessed the decoding accuracy on the held on data via the median decoding error and error cumulative distribution function (CDF) curve.

DATA AND SOFTWARE AVAILABILITY

Datasets 2, 4 and 5 are available from https://github.com/wilsonlab/CRCNS_Shared_Data. Datasets 7 and 8 are publicly available (http://crcns.org). The remaining datasets are available from the lead author upon request. The open-source software for CPU/GPU-based decoding based on Python/C programming is available upon request and will be distributed at http://www.cn3lab.org/software.html.