SPARSE-VIEW INTERPRETABLE 3D ANIMAL BEHAVIOR REPRESENTATIONS FOR NEURAL ENCODING AND DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

A deeper understanding of brain function requires a precise, structured characterization of behavior. Yet, extracting behavioral representations from video in a form suitable for scientific analysis remains a fundamental challenge. Many prior studies represent behavior via pose estimation or nonlinear video embeddings. However, pose tracking discards rich information beyond predefined keypoints, while nonlinear video embeddings lack interpretability. We address this limitation with SABLE (**S**parse-view **A**nimal **B**ehavior **L**atent **E**mbeddings), a self-supervised framework that leverages a geometric inductive bias to learn behavior representations. By augmenting a multi-view transformer with priors from monocular depth and pose estimation, SABLE reconstructs 3D animal behavior from extremely sparse views while learning explicit 3D latent structure. Without ground-truth 3D labels, it reliably recovers 3D behavior from two-view videos, whereas state-of-the-art (SOTA) methods fail or yield degenerate solutions. Across the International Brain Lab and Cheese3D datasets, we demonstrate that SABLE learns 3D representations that match or exceed prior SOTA performance in neural encoding and decoding. Once pretrained across animals, SABLE serves as an off-the-shelf model that generalizes zero-shot to unseen animals without animal-specific calibration or retraining. Our method establishes 3D-aware video embeddings that capture complex behavior, opening new avenues for studying brain–behavior relationships.

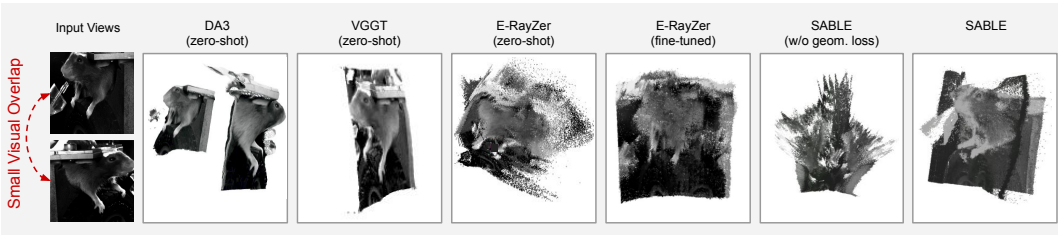

Figure 1: **Geometric inductive bias enables SABLE to reliably reconstruct 3D animal behavior from sparse-view IBL data, where existing 3D foundation models fail to recover coherent 3D structure.** Existing approaches struggle to recover coherent 3D structure in this challenging two-camera setting. Depth Anything 3 (DA3) (Lin et al., 2025) produces plausible monocular point clouds from depth estimation but fails to infer camera poses for cross-view fusion. Visual Geometry Grounded Transformer (VGGT) (Wang et al., 2025), applied zero-shot, collapses to a degenerate single-view 2D shortcut solution, and E-RayZer (Zhao et al., 2026) exhibits a similar failure mode; even after finetuning, it produces overlapping, flattened point clouds. In contrast, SABLE learns accurate camera parameters, avoids degenerate shortcut solutions, and reconstructs coherent 3D mouse structure by incorporating geometric inductive bias. Without the corresponding geometric loss ("w/o geom. loss"), reconstruction degenerates.

1 INTRODUCTION

Neural activity reflects and shapes interactions with the external environment, making behavior a primary lens for understanding the brain (Krakauer et al., 2017; Datta et al., 2019). Advances in large-scale video recordings enable detailed behavioral measurements (Gomez-Marin et al., 2014;

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Berman, 2018), motivating methods that extract informative features from high-dimensional video (Pereira et al., 2020). Pose estimation methods (Mathis et al., 2018; Pereira et al., 2019; Biderman et al., 2024; Aharon et al., 2026) provide interpretable summaries of behavior but miss its full complexity, while nonlinear video embeddings (Batty et al., 2019; Wang et al., 2026b) capture richer information but can be challenging to interpret (Whiteway et al., 2021). Moreover, both are typically computed from 2D projections of an animal that moves in three dimensions. Capturing behavior in 3D is essential for resolving viewpoint ambiguities and recovering fine-grained movements that cannot be inferred from 2D observations alone (Marshall et al., 2022). What is needed, then, are methods that extract dense, interpretable 3D representations of behavior from video.

Recent computer vision foundation models have demonstrated strong 3D reconstruction performance from multi-view images and videos (Lin et al., 2025; Chen et al., 2025; Wang et al., 2025; Zhao et al., 2026). However, applying these approaches to animal behavioral data remains challenging, particularly in neuroscience where video distributions differ from standard benchmarks and camera views are sparse with minimal visual overlap (Yao et al., 2022). In particular, two-view imaging is a common configuration in animal behavior experiments (Musall et al., 2019; Warren et al., 2021), providing the minimal setup to recover 3D animal kinematics through stereo reconstruction (Li et al., 2024; Grier et al., 2026). Compared with larger multi-camera systems, a two-camera setup is more affordable, compact, and easier to deploy. Yet, current 3D foundation models are trained on datasets with dense visual overlap—often 10 views or more (Wang et al., 2025; Zhao et al., 2026)—and therefore struggle with the limited visual overlap present in the two-view setting. A representative example is the two-view dataset from the International Brain Laboratory (IBL) (IBL et al., 2025), which captures a mouse turning a wheel in a visual decision-making task. On this dataset, existing 3D foundation models often produce degenerate reconstructions that fail to capture the animal's behavior (Fig. 1). These limitations motivate models that can recover detailed 3D animal geometry from the *sparse* and *minimally overlapping* views typical of neuroscience experiments.

We address these challenges with a self-supervised 3D reconstruction framework that leverages inductive biases to learn explicit 3D behavioral representations from sparse views for downstream neural encoding and decoding. SABLE (**S**parse-view **A**nimal **B**ehavior **L**atent **E**mbeddings) employs masked modeling to reconstruct 3D Gaussian point clouds from multi-view video. Built on a multi-view transformer pretrained on diverse datasets with overlapping views (Zhao et al., 2026), it is fine-tuned on sparse, minimally overlapping two-view videos. By incorporating pseudo-point clouds derived from monocular depth and animal pose keypoints as geometric priors, SABLE regularizes the learned 3D structure to avoid degenerate solutions. We demonstrate that SABLE reconstructs high-fidelity 3D point clouds from IBL videos using only two sparse views and no 3D labels, whereas existing 3D models fail in this regime (Fig. 1). We further apply SABLE to Cheese3D (Daruwalla et al., 2026), a head-fixed mouse dataset comprising neural recordings and behavioral videos from six camera views, demonstrating its generalization beyond the two-view setting. Through comprehensive neural encoding on both datasets and neural decoding on IBL, we show that SABLE matches or outperforms previous SOTA in predicting neural activity and reconstructing video frames. Moreover, SABLE enables consistent decoding of 3D animal behavior directly from neural recordings and, once pretrained, generalizes zero-shot to unseen animals. To the best of our knowledge, this is the first demonstration that detailed 3D animal behavior can be reliably decoded from intracortical neural activity. The contributions of this work include:

- A self-supervised framework for reconstructing interpretable, stable 3D animal behavior from extremely sparse multi-view video using explicit geometric inductive biases, overcoming the degenerate solutions of prior 3D models.
- A new paradigm for neural encoding and decoding that leverages fine-grained 3D behavioral structure to better capture brain–behavior relationships.
- A pretrained model that generalizes to unseen animals with minimal adaptation and is readily deployable in labs collecting head-fixed mouse neural recordings and behavioral videos.

2 Related Work

Sparse keypoint representations for 3D animal behavior analysis. Deep learning has advanced video-based animal behavior quantification (Mathis & Mathis, 2020; Pereira et al., 2020), com-

monly through per-view 2D keypoint estimation (Mathis et al., 2018; Pereira et al., 2022; Biderman et al., 2024) followed by 3D triangulation (Nath et al., 2019; Karashchuk et al., 2021). Recent methods demonstrate that this pipeline can be improved by integrating information across views before the final 3D prediction: DANNCE fuses multi-view image features into a volumetric representation (Dunn et al., 2021), while Lightning Pose 3D uses a multi-view transformer with 3D reprojection supervision (Aharon et al., 2026). However, these supervised methods produce only sparse keypoints, rather than dense animal geometry or appearance.

Large-scale foundation models for 3D reconstruction. Recent 3D foundation models mark a paradigm shift toward direct, feed-forward inference of 3D structure from multi-view data (Wang et al., 2024; 2025). Visual Geometry Grounded Transformer (VGGT) jointly estimates cameras, depth, and point clouds with geometric supervision (Wang et al., 2025). Depth Anything 3 (DA3) predicts consistent depth and ray maps (Lin et al., 2025); and E-RayZer learns 3D-aware representations self-supervised through Gaussian-splatting reconstruction (Zhao et al., 2026). Despite this progress, sparse, minimally overlapping animal views are out of distribution for these models, often causing unstable camera poses, degenerate geometry, or 2D collapse (Fig. 1). This motivates domain-adapted geometric priors for 3D animal behavior reconstruction.

Learning behavior representations for neural analysis. Existing approaches learn behavioral representations for downstream neuro-behavioral analyses, but they remain limited to 2D or depend on dense multi-view setups. For example, BEAST learns video representations for neural encoding (Wang et al., 2026b), but its ImageNet-pretrained ViT backbone (He et al., 2022) primarily captures 2D semantic information while overlooking cues encoded in structure, motion, and depth. MoReMouse reconstructs dense 3D mouse pose using a canonical mouse avatar learned from synthetic data (Zhong et al., 2026), but this reliance on synthetic data limits its applicability to real-world sparse-view recordings. Pose Splatter (Goffinet et al., 2025) and BEAST3D (Wang et al., 2026a) were both developed for animal behavior analysis and learn dense 3D representations, but both require at least three to four camera views and cannot operate in the two-view setting. Thus, existing approaches cannot learn dense 3D behavioral representations from sparse views.

3 DATASET

The IBL Brainwide Map dataset records brain-wide neural activity and behavior in head-fixed mice performing a standardized visual decision task: mice discriminate visual stimuli presented on the left or right side of a screen and report their choice by turning a response wheel (IBL et al., 2025). To show generalizability, we also evaluate SABLE on Cheese3D (Daruwalla et al., 2026), an additional head-fixed mouse dataset (Appendix A.2.1).

Neural data. Neural activity is measured using Neuropixels probes, which provide high-density, multichannel recordings of spiking activity from hundreds of neurons simultaneously (Jun et al., 2017). In our experiments, we analyze data from ten sessions across ten animals, using seven for pretraining and holding out three to test generalization to unseen data. Each recording session includes up to two probes, with insertion sites varying across animals to sample diverse brain areas. From each held-out session, we randomly sample neural activity from 400 trials, yielding 1,200 trials in total, each lasting 1 second. Spike trains are binned at $\sim$16.7 ms resolution (60 bins per 1s trial, matching the 60 Hz video frame rate), and neurons with mean firing rates below 5 Hz are excluded. In total, we analyze 2,619 units from 6 probe insertions across 3 mice, covering 37 anatomical regions.

Behavioral data. Behavioral measurements include wheel movements and video-based pose estimates (Biderman et al., 2024) from the held-out sessions (Appendix A.1.2). In each session, neural recordings are paired with synchronized videos from two cameras (left and right) capturing wheel-turning behavior. The cameras record at 60 Hz and 150 Hz; the latter is resampled to 60 Hz ($\sim$16.7 ms per frame) via nearest-neighbor interpolation. The two views have limited overlap, primarily covering the nose, forepaws, and wheel (see Fig. 1). In each session used for pretraining, we select 1,000 frames using a Principal Component Analysis (PCA) + k-means strategy to capture diverse behaviors (Appendix A.1.1) for model training. For evaluation, we extract frames aligned with the timestamps of the 400 sampled neural trials in each session, yielding a total of 24,000 pairs of frames per session for downstream analyses.

4 METHODS

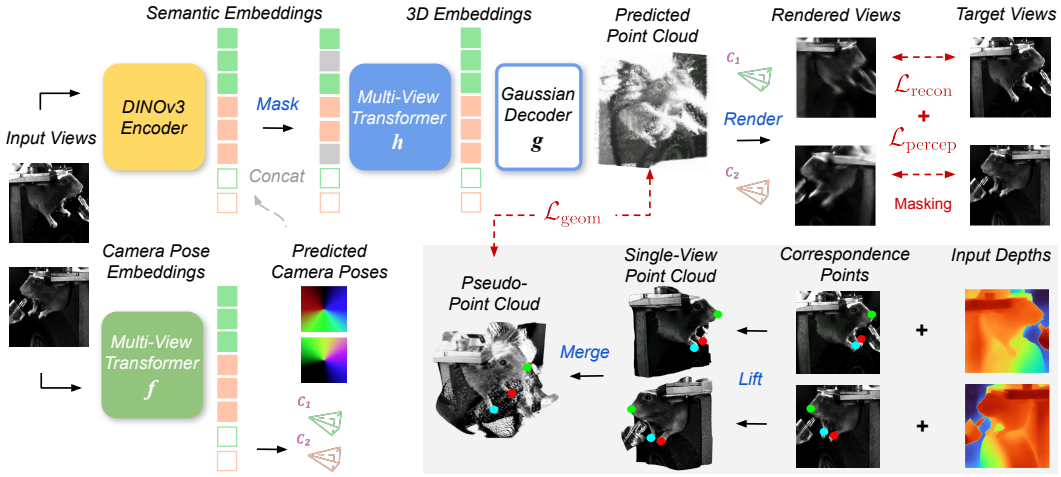

Figure 2: **SABLE framework.** A multi-view transformer predicts camera poses from two input views, which are converted into Plücker ray maps. These rays are concatenated with image tokens from a DINOv3 encoder that capture semantic information, masked with learnable mask tokens, and processed by a second multi-view transformer to produce 3D-structured embeddings that parameterize a Gaussian splatting decoder for 3D reconstruction. To prevent shortcut solutions, we impose geometric regularization via pseudo-point clouds derived from monocular depth and aligned across views using animal pose keypoints as correspondences. The predicted Gaussian point cloud is projected using the estimated camera poses and rendered into 2D images. Training is guided by a masked view reconstruction loss $\mathcal{L}_{\text{recon}}$ for pixel-level accuracy, a perceptual loss $\mathcal{L}_{\text{percep}}$ for perceptual similarity, and a geometric regularization term $\mathcal{L}_{\text{geom}}$ that constrains the 3D locations of the predicted point clouds.

Our insights. Self-supervised 3D learning from sparse, minimally overlapping views is ill-posed: reconstruction alone often collapses to trivial 2D solutions (Younis & Cheng, 2025). We address this by supervising predicted 3D point clouds with pseudo-point clouds derived from external depth and pose models. The distilled geometry provides an inductive bias that constrains the solution space and discourages trivial shortcuts and favors geometry-consistent representations for downstream neural analyses.

Overview. As shown in Fig. 2, SABLE predicts camera parameters with a multi-view transformer, then combines masked DINOv3 image tokens (Siméoni et al., 2025) with ray tokens from a second multi-view transformer to learn 3D-structured embeddings. These embeddings parameterize a Gaussian-splatting decoder that reconstructs 3D point clouds. Both transformers are initialized from E-RayZer (Zhao et al., 2026) to inherit multi-view and camera-pose priors. To prevent shortcut solutions, we regularize predictions using pseudo-point clouds derived from monocular depth, which are aligned across views using Lightning Pose keypoints (Aharon et al., 2026) as correspondences. A differentiable renderer projects the point clouds back into 2D views. After multi-animal pretraining, SABLE generalizes zero-shot to unseen animals without dataset-specific adaptation.

Notation. For each video frame, we consider V views of input images $I = \{I^{(v)}\}_{v=1}^{V}$, each with resolution $H \times W$ and C channels, and tokenize each view into $h \times w$ patches, where $h = H/p$ and $w = W/p$ denote the number of patches along the height and width, respectively, for patch size p. We denote tokens by $\mathbf{t}$, intermediate features by $\mathbf{e}$, and learned embeddings by $\mathbf{z}$. We further denote depth maps by $\mathbf{d}$ and point clouds by $\mathbf{p}$, which represent the 3D positions of points.

4.1 MODEL ARCHITECTURE

Multi-view transformer. SABLE comprises two E-RayZer–style multi-view transformers with alternating within-view and cross-view attention (Zhao et al., 2026). We tokenize all input views into patches to obtain image tokens $\mathbf{t}_{\text{view}} \in \mathbb{R}^{(Vhw)\times D}$, with D being the embedding dimension. A first transformer f encodes these tokens along with prepended camera pose tokens $\mathbf{t}_{\text{cam}}$ into intermediate

features

$$\mathbf{e}_{\text{cam}}, \mathbf{e}_{\text{view}} = f(\mathbf{t}_{\text{cam}}, \mathbf{t}_{\text{view}}), \quad \mathbf{e}_{\text{cam}} \in \mathbb{R}^{V\times D}, \ \mathbf{e}_{\text{view}} \in \mathbb{R}^{(Vhw)\times D}. \tag{1}$$

$\mathbf{e}_{\text{cam}}$ are used to predict camera intrinsics and extrinsics, which are converted into Plücker ray tokens $\mathbf{r}_{\text{plk}} \in \mathbb{R}^{(Vhw)\times D}$ (Hartley & Zisserman, 2003). We further process the image tokens using a DINOv3 encoder with a ViT-B backbone (Dosovitskiy et al., 2021), and then concatenate and fuse the image and ray tokens using a multi-layer perceptron (MLP) l. Per-view summary [CLS] tokens $\mathbf{e}^{\text{[CLS]}}_{\text{view}} \in \mathbb{R}^{V\times D}$, inherited from the DINOv3 [CLS] tokens, are also prepended. This sequence of tokens is masked with learnable mask tokens and passed to a second transformer h to produce 3D-aware embeddings:

$$(\mathbf{z}^{\text{[CLS]}}_{\text{view}}, \mathbf{z}_{\text{view}}) = h\big(\mathbf{e}^{\text{[CLS]}}_{\text{view}}, \, l(\mathbf{e}_{\text{view}}, \mathbf{r}_{\text{plk}})\big). \tag{2}$$

Unlike E-RayZer (Zhao et al., 2026), which masks entire views (often many at once) and reconstructs them from the remaining ones, SABLE is tailored to the extreme two-view setting. In this regime, removing one view leaves only a single view, providing insufficient geometric constraints for 3D recovery and leading to 2D collapse. Instead, SABLE applies patch-level masking across both views to preserve cross-view context and reduce visual ambiguity.

Gaussian splatting decoder and renderer. Given per-token image patch embeddings $\mathbf{z}_{\text{view}}$, we use a lightweight linear decoder g to predict per-pixel Gaussian parameters via upsampling:

$$\mathcal{G} = g(\mathbf{z}_{\text{view}}) \in \mathbb{R}^{(VHW)\times F}, \tag{3}$$

where F denotes the number of Gaussian parameters. Each element of $\mathcal{G}$ corresponds to a pixel-aligned 3D Gaussian primitive. For each Gaussian i, the parameters include 3D position $\mathbf{p}_i \in \mathbb{R}^3$, orientation represented as a quaternion $\mathbf{q}_i \in \mathbb{R}^4$, scale $\mathbf{s}_i \in \mathbb{R}^3$, opacity $\alpha_i \in \mathbb{R}$, and spherical harmonic coefficients $\mathbf{C}_i \in \mathbb{R}^{(d_{SH}+1)^2\times 3}$ (Kerbl et al., 2023). The predicted Gaussians are projected into each view using the camera intrinsics and extrinsics estimated by transformer f, and rendered into reconstructed 2D images $\hat{I}$ via a differentiable Gaussian splatting renderer.

Pseudo-point cloud generation. We obtain monocular depth maps $\mathbf{d}^{(v)} \in \mathbb{R}^{HW}$ for each view $v \in \{1, \ldots, V\}$ using Video Depth Anything (VDA) (Chen et al., 2025), and normalize them to $\tilde{\mathbf{d}}^{(v)} \in [-1, 1]^{HW}$. We further define normalized image-plane coordinates $(\tilde{x}_i^{(v)}, \tilde{y}_i^{(v)}) \in [-1, 1]^2$ for each pixel i. Each pixel is lifted to a 3D point in a pseudo-coordinate space $\tilde{\mathbf{p}}_i^{(v)} = (\tilde{x}_i^{(v)}, \tilde{y}_i^{(v)}, \tilde{d}_i^{(v)}) \in \mathbb{R}^3$. Aggregating all pixels yields a per-view pseudo-point cloud $\tilde{\mathbf{p}}^{(v)} \in \mathbb{R}^{HW\times 3}$. For two views, we denote the corresponding pseudo-point clouds as $\tilde{\mathbf{p}}^{(\text{src})}, \tilde{\mathbf{p}}^{(\text{tgt})} \in \mathbb{R}^{HW\times 3}$.

We use a pose tracking algorithm (Aharon et al., 2026) to identify $N \ll HW$ keypoints in each 2D view (e.g., nose, left paw, and right paw), and treat their lifted 3D locations in the pseudo-coordinate space as correspondences for view merging. Let $\{\tilde{\mathbf{u}}_n^{(\text{src})}\}_{n=1}^N$ and $\{\tilde{\mathbf{u}}_n^{(\text{tgt})}\}_{n=1}^N$ denote these corresponding points from both pseudo-point clouds. For each frame, we then estimate a rigid transformation via the Kabsch algorithm (Kabsch, 1976):

$$\min_{\mathbf{R}\in SO(3),\, \mathbf{T}} \sum_n \|\mathbf{R}\tilde{\mathbf{u}}_n^{(\text{src})} + \mathbf{T} - \tilde{\mathbf{u}}_n^{(\text{tgt})}\|_2^2. \tag{4}$$

The optimal rotation $\mathbf{R}$ and translation $\mathbf{T}$ are applied to map the source point cloud to the target coordinate space $\tilde{\mathbf{p}}^{(\text{src})'} = \mathbf{R}\tilde{\mathbf{p}}^{(\text{src})} + \mathbf{T}$. Finally, we concatenate the source and target point clouds represented in the same coordinate system into a single set to obtain a pseudo-point cloud for geometric regularization.

4.2 Training Objectives

Masked reconstruction and perceptual losses. For each frame, let $\hat{I}$ and I denote the rendered and target images and $\mathcal{M}$ the set of masked pixels across all views. We compute:

$$\mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{percep}} = \frac{1}{|\mathcal{M}|} \sum_{i\in\mathcal{M}} \|\hat{I}_i - I_i\|_2^2 + \sum_l \frac{1}{|\phi_l|} \|\phi_l(\hat{I}) - \phi_l(I)\|_1, \tag{5}$$

where $\phi_l(\cdot)$ denotes features from layer l of a pretrained VGG-19 network (Simonyan & Zisserman, 2015).

Geometric regularization. Let $\hat{\mathbf{p}}, \tilde{\mathbf{p}} \in \mathbb{R}^{(VHW)\times 3}$ denote the predicted and pseudo point clouds for each frame. We first normalize both point clouds to remove global translation and scale differences before enforcing geometric consistency via:

$$\mathcal{L}_{\text{geom}} = \frac{1}{VHW} \sum_{i=1}^{VHW} \|\hat{\mathbf{p}}_i - \tilde{\mathbf{p}}_i\|_2^2. \tag{6}$$

The final objective is a weighted sum of losses, with λ denoting the weights (Appendix E.1):

$$\mathcal{L} = \lambda_{\text{recon}} \cdot \mathcal{L}_{\text{recon}} + \lambda_{\text{percep}} \cdot \mathcal{L}_{\text{percep}} + \lambda_{\text{geom}} \cdot \mathcal{L}_{\text{geom}}. \tag{7}$$

5 RESULTS

5.1 3D ANIMAL BEHAVIOR RECONSTRUCTION

Learning 3D animal pose is essential for capturing body–environment interactions beyond ambiguous 2D projections and for studying their neural basis. On IBL data, we show that our method reconstructs accurate animal point clouds from only two minimally overlapping views, whereas state-of-the-art 3D models collapse to degenerate solutions. We further validate this on Cheese3D (Daruwalla et al., 2026); see Appendix A.2.5 for results.

Models. We benchmark point cloud reconstruction quality in sparse-view settings against the following large-scale 3D models: (1) *VGGT* employs a multi-view transformer with alternating local–global attention to directly infer 3D attributes of a scene from multiple views and is trained with 3D supervision (Wang et al., 2025). (2) *DA3* uses a DINO-style transformer with cross-view attention to predict depth via teacher–student training and ray maps using a dual-DPT head; point clouds are then obtained by back-projecting pixels along the predicted rays using the estimated depth values (Lin et al., 2025). (3) *E-RayZer* is a self-supervised model that adopts VGGT-style multi-view transformers for camera pose estimation and Gaussian splatting for cross-view reconstruction (Zhao et al., 2026).

Evaluation. We qualitatively compare predicted 3D point clouds across baseline models on IBL data captured from two camera views. Since ground-truth 3D labels are unavailable, models that rely on 3D supervision, such as VGGT, are evaluated only in a zero-shot setting using pretrained checkpoints. We also report zero-shot results for *DA3* and E-RayZer. Because E-RayZer is self-supervised and does not require 3D labels, we additionally finetune it on the IBL two-camera dataset and visualize the resulting point clouds. SABLE is trained under the same setting as E-RayZer for a direct comparison.

Results. As shown in Fig. 1, 3D point cloud reconstruction from IBL data is particularly challenging due to minimal cross-view overlap (e.g., nose, paws, and wheel), distribution shift from standard 3D datasets, and the absence of ground-truth 3D supervision. Under these conditions, existing large-scale 3D models fail to recover coherent geometry. DA3 leverages monocular depth cues to produce plausible single-view point clouds, but fails to estimate accurate camera poses, so that the final point cloud is misaligned across views. VGGT collapses to a degenerate single-view 2D solution, and E-RayZer, when applied zero-shot, exhibits a similar failure mode; even after finetuning, E-RayZer produces overlapping, flattened point clouds due to inaccurate camera pose estimation. In contrast, SABLE learns consistent camera parameters and reconstructs coherent 3D mouse structure using a geometry loss based on pseudo-point clouds, which provides a geometric inductive bias. Without this loss, the reconstruction degenerates, demonstrating its importance for resolving ambiguity in sparse-view 3D reconstruction. Importantly, these interpretable 3D representations enable analysis of animal pose and movement dynamics while supporting downstream neural encoding and decoding.

5.2 NEURAL DECODING

Neural decoding measures how well neural activity predicts behavior and underlies brain–computer interfaces such as motor and speech prosthetics (Silva et al., 2024). Here, we show that our method learns behavioral representations that are more accurately decoded from neural activity, enabling reconstruction of 3D animal behavior. We focus on IBL, which provides a stronger decoding signal with at least 708 neurons per session, versus 8 in Cheese3D.

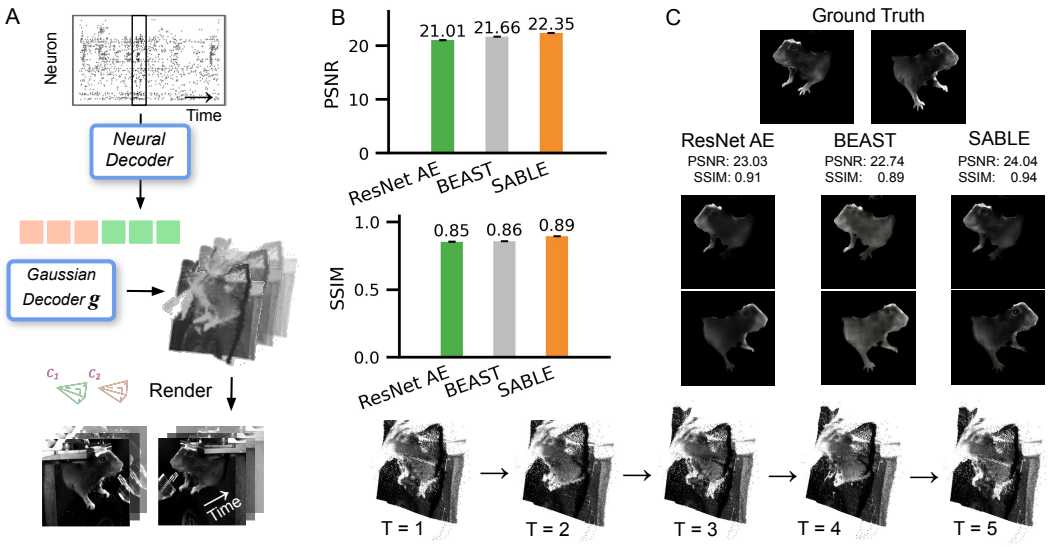

Figure 3: **Benchmarking neural decoding of 2D video frames and 3D point clouds from neural activity.** (A) For each video frame, we train a TCN decoder to predict image patch embeddings from neural activity, which are then passed to a Gaussian decoder to generate 3D point clouds and render 2D images. (B) Decoding performance is evaluated across baseline models that learn video-derived behavioral embeddings. ResNet AE compresses image features extracted with a ResNet-18 backbone into latent vectors, while BEAST and SABLE use image patch embeddings produced by a ViT-MAE and a multi-view transformer, respectively. SABLE outperforms all baselines in video frame decoding quality, as measured by PSNR and SSIM, which evaluate pixel-level accuracy and perceptual quality. Error bars indicate the standard error of the mean across 14,400 frame pairs from held-out sessions. (C) Comparison of decoded video frames from ResNet AE, BEAST, and SABLE against the ground truth. (D) The Gaussian point clouds decoded from neural activity capture fine-grained paw movements.

Models. We benchmark the following nonlinear baselines on decoding video frames from neural activity: (1) *ResNet autoencoder (AE)*: We use a ResNet-18 backbone to extract frame features (He et al., 2016), compress them into nonlinear latents via an MLP encoder, and reconstruct images with an MLP decoder and another ResNet-18 (Batty et al., 2019). Video embeddings are constructed by concatenating latents from both views. (2) *BEAST*: We finetune an ImageNet-pretrained ViT-MAE (He et al., 2022) with a ViT-B backbone (Dosovitskiy et al., 2021) on IBL data to learn per-view image patch embeddings, which are fed into a ViT-MAE decoder for frame reconstruction (Wang et al., 2026b). (3) SABLE: The multi-view transformers f and h are initialized from a pretrained E-RayZer model (Appendix E). We feed the image patch embeddings for each frame, $\mathbf{z}_{\text{view}}$ (Eq. 2), into a Gaussian splatting decoder to predict point clouds rendered into 2D images.

Video frame and point cloud reconstruction. For each baseline, we follow the procedure below to decode video frames. For notational simplicity, we denote the video embeddings learned by each baseline as $\mathbf{z} \in \mathbb{R}^{(KT)\times(VSD)}$, where K is the number of trials, T the number of time bins, V the number of views, S the sequence length, and D the embedding dimension. For ResNet AE, $S = 1$, as each view is compressed into a single D-dimensional vector. For BEAST and SABLE, $S = hw$ corresponds to the number of image patches (see Section 4.1). We set $D = 768$ for all baselines to match the embedding dimension of SABLE. Since the mouse is head-fixed and the background is static, we subtract the frame average from $\mathbf{z}$ to obtain $\tilde{\mathbf{z}}$, which captures only frame-specific behavioral variations. Since $\tilde{\mathbf{z}}$ is high-dimensional, we apply Principal Component Analysis (PCA) to obtain a compact latent vector $\tilde{\mathbf{z}}_{\text{compact}} \in \mathbb{R}^{(KT)\times P}$, where $P \ll D$, for decoding. We then predict $\hat{\mathbf{z}}_{\text{compact}}$ from neural activity and reconstruct $\hat{\mathbf{z}} \in \mathbb{R}^{(KT)\times(VSD)}$ via PCA inversion, followed by adding back the frame average. The reconstructed $\hat{\mathbf{z}}$ is then passed through the (pretrained) decoder of the corresponding model, namely ResNet AE, BEAST, or SABLE, for frame reconstruction (Whiteway et al., 2021). In addition, SABLE uniquely enables decoding of 3D animal point clouds across video frames.

Evaluation. We pretrain all baselines and SABLE on seven sessions, then apply them zero-shot to extract embeddings from frames corresponding to the 400 sampled trials in each of the three

held-out sessions (Section 3). These trials are split into 70% training, 10% validation, and 20% test sets. Following the aforementioned procedure, we compress the extracted video embeddings from each baseline into a P-dimensional latent vector ($P = 6$, with 3 principal components (PCs) per view). Given neural activity, we then train a temporal convolutional network (TCN) model (Syeda et al., 2024) to predict these latents for each baseline (Appendix E.4). TCN decoders are trained on the training set, with hyperparameters tuned and checkpoints selected using the validation set. Performance is evaluated on the test set using foreground-only peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM) (Appendix B.3-B.4), computed over the foreground animal segmented using SAM3 (Carion et al., 2026), where higher values indicate better frame reconstruction quality. See Appendix B.5 for details on the rationale and computation of the foreground-only metrics. Hyperparameters are optimized with Ray Tune by sampling 30 configurations (Appendix F.4). This procedure is repeated across all held-out sessions.

Results. In Fig. 3, we show that neural activity contains sufficient information to decode high-resolution video frames from the IBL dataset. BEAST outperforms ResNet AE in video frame decoding quality, while SABLE further surpasses BEAST on both PSNR and SSIM, indicating higher pixel-level fidelity and perceptual similarity to the ground truth (Fig. 3B). All baseline models capture frame-to-frame behavioral variability after accounting for static background information (Fig. 3C). Beyond 2D video reconstruction, SABLE also decodes 3D point clouds that capture frame-specific behavioral dynamics over time. By recovering 3D behavioral structure containing information about posture and movement, our approach opens a new avenue for studying brain–behavior relationships.

5.3 Neural Encoding

Neural encoding complements decoding by quantifying how much variance in neural activity is explained by behavioral variables and also underlies neural prosthetics, such as retinal implants (Paninski et al., 2007). Here, we evaluate how well behavioral representations from different approaches explain neural activity. We focus on the IBL dataset here; Cheese3D results follow the same trend and are provided in Appendix A.2.6.

Models. We compare against the following baselines that extract behavioral features from video: (1) *Behavioral keypoints*, including wheel velocity, whisker motion energy, nose speed, and left and right paw speeds from each camera view. (2) *PCA*: We apply PCA to compress pixel values into linear embeddings and concatenate the PCs from both views as the final representation. (3) *ResNet AE*: We concatenate the autoencoder latents from both views as the behavioral representations. (4) *BEAST*: We use the ViT-MAE `[CLS]` tokens from both views as frame-level behavioral features (Wang et al., 2026b). (5) SABLE: For each frame, we concatenate the DINOv3 `[CLS]` tokens from both views to summarize the behavioral information. (6) *Random*: a random-embedding baseline, obtained by shuffling SABLE embeddings across trials before predicting neural activity (Appendix B.2).

Evaluation. We benchmark neural encoding across baselines by predicting neural activity from behavior representations extracted by each model. Except for behavioral traces, all baselines are first pretrained across seven sessions and then applied zero-shot to extract embeddings from frames in each of the three held-out sessions corresponding to the 400 sampled trials (Section 3). We train a TCN encoder (Syeda et al., 2024) to predict neural activity from behavioral traces or video embeddings (Appendix E.4). Training, model selection, and hyperparameter tuning follow the same protocol as in the neural decoding setup (Section 5.2). The best encoder is then evaluated on the test set using bits per spike (BPS) (Appendix B.1), where higher values indicate better encoding performance. Results are reported across 2,619 neurons from the held-out sessions.

Results. In Fig. 4, PCA representations surpass keypoint-based approaches, suggesting that behavioral videos contain richer information than pose estimation alone. ResNet AE further outperforms PCA, indicating that nonlinear embeddings capture behavioral structure beyond linear representations with the same latent dimensionality (Batty et al., 2019). We additionally evaluate BEAST as a transformer-based video baseline. Across these conventional 2D video representations, SABLE achieves the best prediction performance. Although the average improvement in the bar plot is modest, SABLE consistently improves prediction quality for most neurons across all held-out sessions

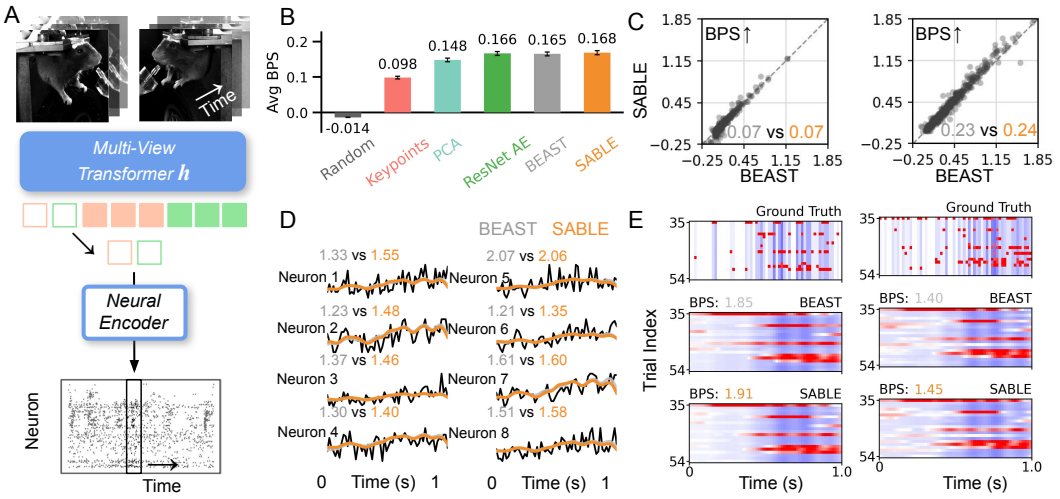

Figure 4: **Quantitative and qualitative comparison of neural activity prediction using behavioral features extracted from IBL video data.** (A) At each time point, we concatenate the DINOv3 [CLS] tokens from the two camera views, then use a TCN over time to predict neural activity. (B) Encoding performance is evaluated across baseline models that learn video-derived behavioral embeddings. Behavioral keypoints provide interpretable summaries derived from sensors or animal pose tracking. PCA learns linear latent representations from video, while ResNet AE learns nonlinear latents. BEAST uses [CLS] tokens from the ViT-B backbone of a ViT-MAE. SABLE outperforms all baselines in neural prediction quality measured by bits per spike (BPS). We also include a random-embedding baseline as a control, confirming it yields near-zero BPS. Error bars indicate the standard error of the mean. (C) Scatterplot comparison of SABLE vs BEAST performance in two example sessions. Each dot corresponds to an individual neuron. The values in the bottom-right corner represent the session-averaged BPS. Although the improvement in BPS is modest, SABLE outperforms BEAST in prediction quality for most neurons. (D) Comparison of predicted firing rates averaged over randomly sampled 1s recording segments for eight example neurons using SABLE and BEAST. The values in the top-left corner represent per-neuron BPS. (E) Comparison of residual firing-rate variability (heatmaps), obtained by subtracting each neuron's mean firing rate across recording segments, for two example neurons (columns) using SABLE and BEAST.

(Fig. 4 C). These results suggest that explicitly modeling 3D behavioral structure provides additional information beyond conventional 2D video embeddings for explaining neural activity.

6 Discussion

We introduce SABLE, a self-supervised framework that learns interpretable 3D animal behavior representations from sparse multi-view video. Its geometric inductive biases reconstruct reliable 3D structure where existing methods fail, improving neural encoding and enabling 3D behavior decoding from neural activity. This work has several limitations. First, although SABLE can be pretrained on multi-animal data and generalize zero-shot to unseen animals, we have only applied it to head-fixed mouse datasets. Extending SABLE to freely moving animals and other species is an important direction for future work. Second, while SABLE supports novel-view synthesis, future work could use these synthesized views to extract additional behavioral keypoints and potentially improve neural encoding performance.

AI use statement

We used generative AI tools for software debugging, grammar checking, and improving the writing. All AI-assisted content was reviewed by the authors, who take responsibility for the final work.

Reproducibility statement

Details on the model hyperparameters, training procedures, and experimental settings are provided in the main text and appendix for reproducibility. The appendix also includes details on the features used, data splits, and preprocessing for both video and neural recordings.

REFERENCES

Lenny Aharon, Matthew R Whiteway, Karan Sikka, Keemin Lee, Yanchen Wang, Selmaan Chettih, Benjamin Midler, Ilana B Witten, Dmitriy Aronov, International Brain Laboratory, et al. Lightning pose 3d: an uncertainty-aware framework for data-efficient multi-view animal pose estimation. *bioRxiv*, pp. 2026–04, 2026.

Eleanor Batty, Matthew Whiteway, Shreya Saxena, Dan Biderman, Taiga Abe, Simon Musall, Winthrop Gillis, Jeffrey Markowitz, Anne Churchland, John Cunningham, et al. Behavenet: nonlinear embedding and bayesian neural decoding of behavioral videos. *Advances in neural information processing systems*, 32, 2019.

Gordon J Berman. Measuring behavior across scales. *BMC biology*, 16(1):23, 2018.

Dan Biderman, Matthew R Whiteway, Cole Hurwitz, Nicholas Greenspan, Robert S Lee, Ankit Vishnubhotla, Richard Warren, Federico Pedraja, Dillon Noone, Michael M Schartner, et al. Lightning pose: improved animal pose estimation via semi-supervised learning, bayesian ensembling and cloud-native open-source tools. *Nature Methods*, 21(7):1316–1328, 2024.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris Coll-Vinent, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. In *International conference on learning representations*, volume 2026, pp. 138846–138923, 2026.

Sili Chen, Hengkai Guo, Shengnan Zhu, Feihu Zhang, Zilong Huang, Jiashi Feng, and Bingyi Kang. Video depth anything: Consistent depth estimation for super-long videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22831–22840, 2025.

Kyle Daruwalla, Irene Nozal Martin, Linghua Zhang, Diana Naglič, Andrew Frankel, Catherine Rasgaitis, Rubin Zhao, Xinyan Zhang, Zainab Ahmad, Jeremy C Borniger, et al. Cheese3d enables sensitive detection and analysis of whole-face movement in mice. *Nature Neuroscience*, pp. 1–12, 2026.

Sandeep Robert Datta, David J Anderson, Kristin Branson, Pietro Perona, and Andrew Leifer. Computational neuroethology: a call to action. *Neuron*, 104(1):11–24, 2019.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Timothy W Dunn, Jesse D Marshall, Kyle S Severson, Diego E Aldarondo, David GC Hildebrand, Selmaan N Chettih, William L Wang, Amanda J Gellis, David E Carlson, Dmitriy Aronov, et al. Geometric deep learning enables 3d kinematic profiling across species and environments. *Nature methods*, 18(5):564–573, 2021.

Jack Goffinet, Youngjo Min, Carlo Tomasi, and David E Carlson. Pose splatter: A 3d gaussian splatting model for quantifying animal pose and appearance. *arXiv preprint arXiv:2505.18342*, 2025.

Alex Gomez-Marin, Joseph J Paton, Adam R Kampff, Rui M Costa, and Zachary F Mainen. Big behavioral data: psychology, ethology and the foundations of neuroscience. *Nature neuroscience*, 17(11):1455–1462, 2014.

Harrison A. Grier, Sohrab Salimian, and Matthew T. Kaufman. Mouse sensorimotor cortex reflects complex kinematic details during reaching and grasping. *eLife*, 14:RP106270, February 2026. ISSN 2050-084X. doi: 10.7554/eLife.106270.

Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009, 2022.

IBL, Dora Angelaki, Brandon Benson, Julius Benson, Daniel Birman, Niccolò Bonacchi, Kcénia Bougrova, Sebastian A Bruijns, Matteo Carandini, Joana A Catarino, et al. A brain-wide map of neural activity during complex behaviour. *Nature*, 645(8079):177–191, 2025.

James J Jun, Nicholas A Steinmetz, Joshua H Siegle, Daniel J Denman, Marius Bauza, Brian Barbarits, Albert K Lee, Costas A Anastassiou, Alexandru Andrei, Çağatay Aydın, et al. Fully integrated silicon probes for high-density recording of neural activity. *Nature*, 551(7679):232–236, 2017.

Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 32(5):922–923, 1976.

Pierre Karashchuk, Katie L Rupp, Evyn S Dickinson, Sarah Walling-Bell, Elischa Sanders, Eiman Azim, Bingni W Brunton, and John C Tuthill. Anipose: A toolkit for robust markerless 3d pose estimation. *Cell reports*, 36(13), 2021.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023.

John W Krakauer, Asif A Ghazanfar, Alex Gomez-Marin, Malcolm A MacIver, and David Poeppel. Neuroscience needs behavior: correcting a reductionist bias. *Neuron*, 93(3):480–490, 2017.

Yi Li, Xu An, Patrick J. Mulcahey, Yongjun Qian, X. Hermione Xu, Shengli Zhao, Hemanth Mohan, Shreyas M. Suryanarayana, Ludovica Bachschmid-Romano, Nicolas Brunel, Ian Q. Whishaw, and Z. Josh Huang. Cortico-thalamic communication for action coordination in a skilled motor sequence, October 2024. ISSN 2692-8205.

Haotong Lin, Sili Chen, Junhao Liew, Donny Y Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025.

Jesse D Marshall, Tianqing Li, Joshua H Wu, and Timothy W Dunn. Leaving flatland: Advances in 3d behavioral measurement. *Current Opinion in Neurobiology*, 73:102522, 2022.

Alexander Mathis, Pranav Mamidanna, Kevin M Cury, Taiga Abe, Venkatesh N Murthy, Mackenzie Weygandt Mathis, and Matthias Bethge. Deeplabcut: markerless pose estimation of user-defined body parts with deep learning. *Nature neuroscience*, 21(9):1281–1289, 2018.

Mackenzie Weygandt Mathis and Alexander Mathis. Deep learning tools for the measurement of animal behavior in neuroscience. *Current opinion in neurobiology*, 60:1–11, 2020.

Simon Musall, Matthew T. Kaufman, Ashley L. Juavinett, Steven Gluf, and Anne K. Churchland. Single-trial neural dynamics are dominated by richly varied movements. *Nature Neuroscience*, 22(10):1677–1686, October 2019. ISSN 1546-1726. doi: 10.1038/s41593-019-0502-4.

Tanmay Nath, Alexander Mathis, An Chi Chen, Amir Patel, Matthias Bethge, and Mackenzie Weygandt Mathis. Using deeplabcut for 3d markerless pose estimation across species and behaviors. *Nature protocols*, 14(7):2152–2176, 2019.

Liam Paninski, Jonathan Pillow, and Jeremy Lewi. Statistical models for neural encoding, decoding, and optimal stimulus design. *Progress in Brain Research*, 165:493–507, 2007. ISSN 0079-6123. doi: 10.1016/S0079-6123(06)65031-0.

Talmo D Pereira, Diego E Aldarondo, Lindsay Willmore, Mikhail Kislin, Samuel S-H Wang, Mala Murthy, and Joshua W Shaevitz. Fast animal pose estimation using deep neural networks. *Nature methods*, 16(1):117–125, 2019.

Talmo D Pereira, Joshua W Shaevitz, and Mala Murthy. Quantifying behavior to understand the brain. *Nature neuroscience*, 23(12):1537–1549, 2020.

Talmo D Pereira, Nathaniel Tabris, Arie Matsliah, David M Turner, Junyu Li, Shruthi Ravindranath, Eleni S Papadoyannis, Edna Normand, David S Deutsch, Z Yan Wang, et al. Sleap: A deep learning system for multi-animal pose tracking. *Nature methods*, 19(4):486–495, 2022.

Alexander B. Silva, Kaylo T. Littlejohn, Jessie R. Liu, David A. Moses, and Edward F. Chang. The speech neuroprosthesis. *Nature Reviews Neuroscience*, 25(7):473–492, July 2024. ISSN 1471-0048. doi: 10.1038/s41583-024-00819-9.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations (ICLR)*, 2015.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. Dinov3, 2025. URL https://arxiv.org/abs/2508.10104.

Atika Syeda, Lin Zhong, Renee Tung, Will Long, Marius Pachitariu, and Carsen Stringer. Facemap: a framework for modeling neural activity based on orofacial tracking. *Nature neuroscience*, 27 (1):187–195, 2024.

Jianyuan Wang, Nikita Karaev, Christian Rupprecht, and David Novotny. Vggsfm: Visual geometry grounded deep structure from motion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 21686–21697, 2024.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 5294–5306, 2025.

Yanchen Wang, Lenny Aharon, Wangshu Zhu, Kyle Daruwalla, Linghua Zhang, Jiaru Zou, Selmaan Chettih, Helen Hou, Liam Paninski, and Matthew R Whiteway. Beast3d: Animal behavioral analysis and neural encoding from multi-view video via gaussian splatting, June 2026a. URL https://arxiv.org/abs/2606.02937.

Yanchen Wang, Han Yu, Ari Blau, Yizi Zhang, International Brain Laboratory, Liam Paninski, Cole Lincoln Hurwitz, and Matthew R Whiteway. Animal behavioral analysis and neural encoding with transformer-based self-supervised pretraining. In *The Fourteenth International Conference on Learning Representations*, April 2026b. URL https://openreview.net/forum?id=AeqPIRKUni.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. doi: 10.1109/TIP.2003.819861.

Richard A Warren, Qianyun Zhang, Judah R Hoffman, Edward Y Li, Y Kate Hong, Randy M Bruno, and Nathaniel B Sawtell. A rapid whisker-based decision underlying skilled locomotion in mice. *eLife*, 10:e63596, January 2021. ISSN 2050-084X. doi: 10.7554/eLife.63596.

Matthew R Whiteway, Dan Biderman, Yoni Friedman, Mario Dipoppa, E Kelly Buchanan, Anqi Wu, John Zhou, Niccolò Bonacchi, Nathaniel J Miska, Jean-Paul Noel, et al. Partitioning variability in animal behavioral videos using semi-supervised variational autoencoders. *PLoS computational biology*, 17(9):e1009439, 2021.

Chun-Han Yao, Wei-Chih Hung, Yuanzhen Li, Michael Rubinstein, Ming-Hsuan Yang, and Varun Jampani. Lassie: Learning articulated shapes from sparse image ensemble via 3d part discovery. *Advances in Neural Information Processing Systems*, 35:15296–15308, 2022.

Tanveer Younis and Zhanglin Cheng. Sparse-view 3d reconstruction: Recent advances and open challenges. *arXiv preprint arXiv:2507.16406*, 2025.

Qitao Zhao, Hao Tan, Qianqian Wang, Sai Bi, Kai Zhang, Kalyan Sunkavalli, Shubham Tulsiani, and Hanwen Jiang. E-rayzer: Self-supervised 3d reconstruction as spatial visual pre-training. In *CVPR*, 2026.

Yuan Zhong, Jingxiang Sun, Zhongbin Zhang, Liang An, and Yebin Liu. Moremouse: Monocular reconstruction of laboratory mouse. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 13548–13556, 2026.

A DATASETS

A.1 IBL

A.1.1 PCA + k-MEANS FRAME SELECTION

Since the animal and experimental apparatus remain largely static in IBL videos, uniformly sampling video frames for model training over-represent inactive states and under-represent rare postures. Therefore, we adopt a PCA + k-means–based frame-selection scheme to construct a compact, behaviorally diverse training set (Wang et al., 2026b). We adopt the same procedure and select approximately 1,000 frames per session for model training. For each session, frames from the left-camera video are resized to a low spatial resolution, vectorized, and used to compute frame-wise motion energy via absolute pixel-wise differences between consecutive frames. We retain frames above a motion-energy percentile threshold, fit PCA within each session on the resulting high-motion frame vectors, and apply k-means clustering to identify anchor frames. Each anchor index i is then expanded to $\{i-1, i, i+1\}$ to incorporate short temporal context. The resulting frame indices are reused for the right camera to ensure that the extracted video frames are temporally synchronized across the left and right views. In total, 1,000 synchronized frame pairs are selected from each session for training. For model evaluation (neural encoding and decoding), we extract video frames temporally aligned with the neural data (Section 3). Each session contains 400 randomly sampled 1-second time segments, with videos sampled at 60 Hz, yielding 24,000 aligned frames from each view. This separation allows the baseline models to be trained on a compact yet diverse set of unlabeled video frames, while neural analyses are performed on frames corresponding to the sampled neural time segments.

A.1.2 LIGHTNING POSE KEYPOINTS

Keypoints are obtained using the Lightning Pose 3D (LP3D) framework (Aharon et al., 2026). We train an ensemble of three LP3D models on 200 manually labeled frames per session using distinct random seeds for the train/validation split. Each model uses a ViT-S/DINO vision transformer backbone within a multi-view transformer architecture, with patch masking and a 3D reprojection loss that leverages camera calibration parameters to enforce geometric consistency across views during training. We then run inference on all video frames for each ensemble member.

To obtain post-processed predictions with calibrated uncertainty estimates, we apply the nonlinear multi-view Ensemble Kalman Smoother (mvEKS) with variance inflation to the ensemble outputs. This post-processing step jointly exploits temporal continuity and multi-view geometric constraints, inflating observation noise for views whose predictions are inconsistent with the remaining views and thereby downweighting overconfident erroneous predictions.

To produce final keypoint trajectories suitable for large-scale analysis, we apply a distillation procedure (Aharon et al., 2026). Frames are first filtered by retaining the $\sim$60% with the lowest posterior predictive variance from the nonlinear mvEKS, keeping only high-confidence predictions. The retained frames are then filtered for pose diversity using k-means clustering in 3D pose space ($k = 3000$). This procedure generates high-quality and diverse pseudo-labels. The resulting pseudo-labeled frames are combined with the original labeled frames across all seeds to form an augmented training set. A single LP3D model is then retrained on this combined dataset and applied to the full videos to generate the final keypoint trajectories used in all downstream analyses.

A.1.3 VIDEO DEPTH EXTRACTION

SABLE uses monocular depth estimates as an auxiliary geometric signal for sparse-view 3D reconstruction (Section 4.1). Monocular relative depth is extracted using Video Depth Anything (VDA) (Chen et al., 2025), which is used as a frozen data preprocessing model rather than a trainable model component. We choose VDA for depth estimation, because it produces temporally consistent depth

predictions across video frames. We use the publicly released ViT-B checkpoint and apply it independently to the extracted left- and right-camera RGB videos for each session to generate dense single-channel relative depth maps. We use relative rather than metric depth, since absolute scene scale is not required for our reconstruction objective and relative depth provides sufficient geometric structure for regularization. In the main experiments, RGB images are resized to 320×320, and the corresponding depth maps are resized to the same spatial resolution to preserve pixel alignment between RGB and depth representations.

A.2 CHEESE3D: NOVEL-VIEW SYNTHESIS AND NEURAL ENCODING

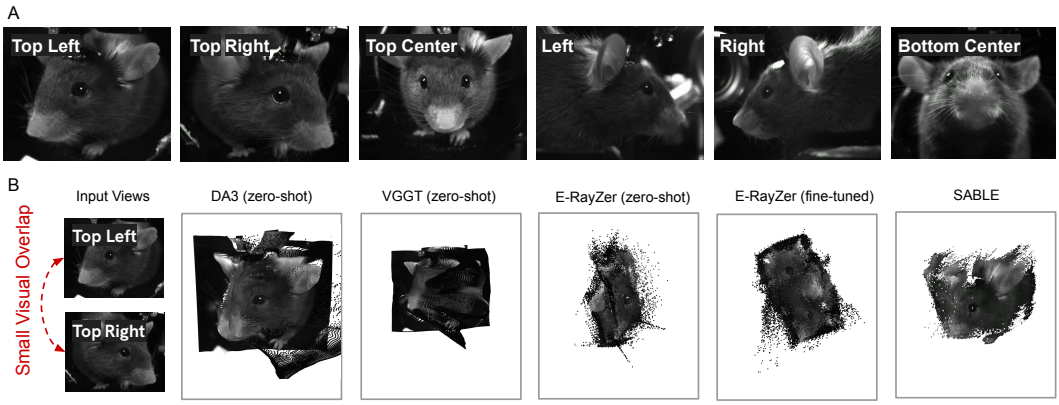

Figure 5: **Cheese3D video frames and 3D reconstruction comparison.** (A) Six camera views from the Cheese3D dataset. (B) 3D reconstruction from the selected sparse view pair (top-right and top-left) of Cheese3D. SABLE reliably reconstructs 3D animal behavior, while existing 3D foundation models exhibit distinct failure modes, consistent with the results on IBL (Fig. 1). DA3 (Lin et al., 2025) and VGGT (Wang et al., 2025) produce plausible per-view point clouds; however, DA3 fails to consistently align the two point clouds, with noticeable misregistration around the nose, and VGGT fails to disentangle foreground and background in depth, leaving background geometry intermingled with the mouse. Both zero-shot and fine-tuned E-RayZer (Zhao et al., 2026) collapse to degenerate, flattened 3D structures. In contrast, SABLE recovers consistent cross-view geometry, separates foreground from background in depth, and avoids degenerate solutions, producing a coherent 3D reconstruction of the mouse.

A.2.1 DATASET

The Cheese3D dataset consists of six synchronized camera views of head-fixed mice, capturing whole-face movements at 100 Hz with a spatial resolution of 1280×1024 pixels (see Fig. 5 panel A). For our experiments, we downsample the videos to 60 Hz. The dataset contains 11 video-recording sessions, one of which includes simultaneously recorded neural activity. We use the 10 sessions without neural recordings for model training, using approximately 5,500 synchronized frame pairs, and use the session with neural recordings for neural encoding evaluation.

To evaluate whether SABLE's sparse-view reconstruction capability generalizes beyond the IBL dataset, we construct an analogous two-view setting from Cheese3D. Specifically, we select the top-left and top-right camera views, which have limited overlap and therefore provide a challenging sparse-view configuration similar to that used for IBL. SABLE reconstructs coherent 3D mouse structure from the selected sparse view pair. In contrast, existing 3D foundation models produce inconsistent or degenerate reconstructions, exhibiting distinct failure modes (Fig. 5, panel B).

A.2.2 LIGHTNING POSE KEYPOINTS

Similar to IBL frame extraction, to select pose-diverse video frames from Cheese3D, we adopt a keypoints + k-means strategy (Wang et al., 2026a). We first obtain geometrically consistent keypoints across all six camera views using a frame-filtering and triangulation procedure. To be more specific, we train an ensemble of three single-view Lightning Pose 3D (LP3D) models (Aharon et al., 2026) on 665 manually labeled instances. Each instance corresponds to a time point and contains labels from a subset of the camera views in which the facial keypoints are anatomically visible, out of up to six camera views. We then run the ensemble on the full-length videos from all

views and retain only frames for which the median keypoint likelihood exceeds 0.6 in at least two views. For each retained frame, we triangulate the keypoints across the available views using the calibrated camera parameters to obtain their 3D locations. Finally, we project the triangulated 3D keypoints back into all six camera views, yielding geometrically consistent 2D keypoint estimates for subsequent pose-diverse frame selection and quality control.

A.2.3 KEYPOINTS + k-MEANS

We next select a pose-diverse subset of frames. For each session, the triangulated 3D keypoints are flattened into a single pose vector, and k-means clustering is applied in this pose space. For each cluster, we select the frame whose 3D pose is closest to the cluster centroid, yielding approximately $50 - 55$ representative frames per session.

We then perform an additional geometric quality-control step based on reprojection error. Specifically, for each selected frame, we compare the median LP3D 2D keypoint predictions with the 2D locations obtained by projecting the triangulated 3D keypoints back into each camera view. The reprojection error is defined as the Euclidean distance between these two estimates. We discard the 25% of selected frames with the largest reprojection errors, thereby removing frames whose multi-view predictions are least geometrically consistent. The remaining frames are used to construct the final training and test sets, containing 450 and 150 instances, respectively.

A.2.4 VIDEO DEPTH EXTRACTION

To obtain monocular depth estimates used by SABLE, we apply Video Depth Anything (VDA) using the same preprocessing procedure described for IBL in Section A.1.3. Specifically, VDA is applied independently to the Cheese3D RGB videos, with input images non-uniformly resized to 320×320 pixels without cropping, producing temporally consistent relative-depth maps aligned with the corresponding RGB frames.

A.2.5 NOVEL VIEW SYNTHESIS

In Fig. 5, we show qualitative 3D reconstructions from sparsely sampled Cheese3D views. Because Cheese3D lacks ground-truth (GT) 3D labels, direct 3D evaluation is not possible. We therefore reduce the problem to a 2D novel-view synthesis (NVS) task: if a model's reconstructed 3D structure is accurate, then rendering it from a new, known camera pose should closely match the real image from that viewpoint.

Evaluation. We benchmark SABLE against E-RayZer (Zhao et al., 2026) on this task, in which each model receives only the top-left and top-right views and predicts a 3D point cloud together with the corresponding camera poses. For each model, we hold out the top-center view, render it using its GT camera pose, and compare the rendering to the GT top-center frame via PSNR and SSIM (mean $\pm$ 1 SE over 100 test frames). This comparison isolates reconstruction quality only if pose estimation error is small; otherwise, a poor rendering could reflect inaccurate camera poses rather than inaccurate 3D structure, confounding the two. We verify that both models' predicted input camera poses closely match the GT camera poses, so held-out view error is attributable primarily to reconstruction quality rather than pose estimation error. Under this condition, better 3D reconstruction yields a held-out rendering closer to GT.

Results. Table 1 reports PSNR and SSIM on the held-out view, alongside the two input views (as a reference upper bound) and an all-black image (as a lower-bound control). The large gap over the all-black control confirms the rendered held-out views capture genuine animal structure rather than trivial content. SABLE outperforms E-RayZer on the held-out view on both metrics, consistent with the qualitative comparisons in Fig. 6, where SABLE's point clouds are visually more accurate. Together, these results indicate SABLE reconstructs 3D mouse behavior more faithfully than E-RayZer from sparse camera views.

A.2.6 NEURAL ENCODING

Previous results show that SABLE outperforms baseline models in the neural encoding experiment on the IBL dataset (Section 5.3). To test whether the neural encoding advantage of SABLE general-

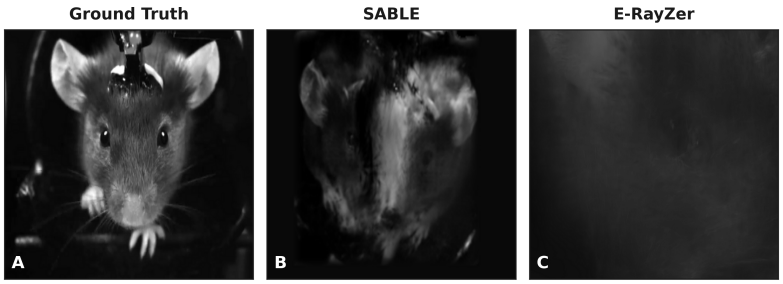

Figure 6: **Novel-view synthesis on Cheese3D.** Both models take only the top-left and top-right views as input and are evaluated on the held-out top-center view. (A) Ground truth. (B) SABLE rendering. (C) E-RayZer rendering. SABLE better matches the ground truth, indicating more accurate 3D reconstruction.

Table 1: Novel-view synthesis results on Cheese3D, comparing E-RayZer and SABLE on input views and a held-out top-center view.

PSNR			
View	All Black Image	E-RayZer	SABLE
Top-Left	11.31	23.69 ± 0.33	$\mathbf{29.74 \pm 0.06}$
Top-Right	11.88	28.10 ± 0.36	$\mathbf{28.78 \pm 0.09}$
Top-Center (Held-Out)	7.35	10.74 ± 0.10	$\mathbf{12.74 \pm 0.06}$
SSIM			
View	All Black Image	E-RayZer	SABLE
Top-Left	0.004	0.812 ± 0.009	$\mathbf{0.892 \pm 0.001}$
Top-Right	0.004	0.863 ± 0.008	$\mathbf{0.898 \pm 0.002}$
Top-Center (Held-Out)	0.001	0.310 ± 0.004	$\mathbf{0.362 \pm 0.002}$

izes beyond the IBL dataset in a minimal-overlap sparse-view setting, we evaluate SABLE and the baseline models on Cheese3D. We train each model using two camera views and compare neural encoding performance using embeddings extracted from the resulting models.

View selection. We select the top-left and top-right Cheese3D views, which have limited visual overlap, analogous to the two-view configuration used for IBL.

Foreground mask extraction. When merging the per-view point clouds directly, background and mouse surfaces can intersect. To isolate the animal surface from the background, we generate a foreground mouse mask using SAM3 (Carion et al., 2026) following (Wang et al., 2026a). We use this mask in two ways. First, we assign background pixels the farthest normalized depth before constructing the two-view point cloud, thereby preventing background pixels from producing spurious points near the mouse surface. Second, we compute $\mathcal{L}_{\text{recon}}$ and $\mathcal{L}_{\text{geom}}$ only over foreground pixels. This prevents the background from contributing to these objectives and focuses the optimization on reconstructing the mouse.

Rigid Transformation For Cheese3D, we manually determine a single rigid transformation that is applied consistently across sessions to align the two-view point clouds and construct the pseudo point cloud used for loss regularization. No LightningPose keypoints are used for this alignment. Despite this simplified alignment procedure, SABLE recovers the underlying 3D structure (Fig. 5).

Neural Encoding Model Training. For neural encoding, models are trained on 10 sessions using approximately 5,500 synchronized frame pairs and evaluated on a held-out session. The held-out session contains simultaneously recorded electrophysiological activity measured with a 32-channel single-shank silicon probe in the brainstem. We randomly sample 1,233 one-second trials from the recording session. Neural activity is available from 8 neurons and is subsequently binned at 60 Hz, matching the downsampled video frame rate, resulting in one neural activity vector per video frame.

Table 2: Neural encoding results on Cheese3D.

	ResNet AE	BEAST	SABLE
Encoding (bps)	0.603 ± 0.144	0.452 ± 0.088	$\mathbf{0.655 \pm 0.146}$

Neural Encoding Results. Neural encoding results on the held-out session across baseline models are summarized in Table 2, which shows the mean bps with standard error across 8 neurons. SABLE

outperforms ResNet AE and BEAST in neural prediction quality measured by bits per spike (BPS). The neural encoding procedure and embedding extraction are the same as those used in the neural encoding experiment on the IBL dataset in Section 5.3.

B METRICS AND BASELINES

B.1 BITS PER SPIKE

Bits per spike (BPS) quantifies the improvement in log-likelihood of a predictive model over a null (mean-rate) model, normalized by the number of observed spikes and expressed in base-2 units. For neuron n, it is defined as

$$\mathrm{BPS} = \frac{\mathcal{L}_{\mathrm{null}} - \mathcal{L}_{\mathrm{pred}}}{N_{\mathrm{sp}} \ln 2}. \tag{8}$$

A higher BPS indicates that the model assigns higher likelihood to the observed spikes than the null model, with $\mathrm{BPS} = 0$ corresponding to the mean-rate baseline.

B.2 RANDOM-EMBEDDING BASELINE FOR NEURAL ENCODING

A BPS value of 0 corresponds to a model that collapses to predicting the trial-averaged neural activity and fails to capture variability across individual trials. We expect a random model to perform at or near this trial-averaged baseline because it does not contain information about the animal behavior. The models' positive BPS values suggest that they capture behavior-related information from videos that is predictive of neural activity, rather than simply reproducing the average neural response.

As a null baseline, we pool embeddings from DINOv3 [CLS] tokens of SABLE across all trials and time bins and randomly permute them, thereby destroying the temporal structure while preserving the embedding content within each frame. As shown in figure 4, the random baseline only achieves near chance-level (0 bps) encoding performance, and therefore, proves SABLE and beseline models capture variability across trials.

B.3 PEAK SIGNAL-TO-NOISE RATIO (PSNR)

Peak signal-to-noise ratio (PSNR) measures pixel-level reconstruction fidelity. For a ground-truth image $x \in [0, M]^{H \times W \times C}$ and a reconstruction $\hat{x}$, we first compute the mean squared error

$$\mathrm{MSE}(x, \hat{x}) = \frac{1}{HWC} \sum_{h=1}^{H} \sum_{w=1}^{W} \sum_{c=1}^{C} (x_{hwc} - \hat{x}_{hwc})^2 . \tag{9}$$

The PSNR is then

$$\mathrm{PSNR}(x, \hat{x}) = 10 \log_{10} \left(\frac{M^2}{\mathrm{MSE}(x, \hat{x})} \right), \tag{10}$$

where M denotes the maximum possible pixel value. In our experiments, we normalize images to $[0, 1]$, yielding $M = 1$. Higher PSNR corresponds to lower reconstruction error in the mean-squared sense. Since PSNR is a monotonic transformation of MSE, it primarily reflects pixel-wise accuracy and is most appropriate for assessing low-level reconstruction quality.

B.4 STRUCTURAL SIMILARITY INDEX MEASURE (SSIM)

Structural similarity index measure (SSIM) evaluates reconstruction quality by comparing local luminance, contrast, and structural statistics (Wang et al., 2004). Given a local window extracted from a ground-truth image x and a reconstruction $\hat{x}$, SSIM is defined as

$$\mathrm{SSIM}(x, \hat{x}) = \frac{(2\mu_x \mu_{\hat{x}} + C_1)(2\sigma_{x\hat{x}} + C_2)}{(\mu_x^2 + \mu_{\hat{x}}^2 + C_1)(\sigma_x^2 + \sigma_{\hat{x}}^2 + C_2)}, \tag{11}$$

where μ_x and $\mu_{\hat{x}}$ denote local means, σ_x^2 and $\sigma_{\hat{x}}^2$ are local variances, $\sigma_{x\hat{x}}$ is the local covariance, and C_1, C_2 are stabilization constants.

We compute SSIM using 11×11 Gaussian windows and average over spatial locations and channels. Compared to PSNR, SSIM is designed to better align with perceptual image quality, assigning higher scores to reconstructions that preserve local structure, edges, and contrast, even when pixel-wise errors are similar.

B.5 Foreground-only PSNR and SSIM

In the neural decoding experiment, we provide a direct evaluation of behavior-related visual reconstruction quality. Nevertheless, among baseline models' decoders may affect reconstruction quality in static background regions, which could confound PSNR and SSIM. Therefore, we compare the reconstruction image quality by computing both metrics exclusively on the mouse foreground.

Specifically, we use SAM3 (Carion et al., 2026) to generate mouse foreground masks of the ground truth frames using the text prompt 'mouse', and then we apply the foreground masks on ground truth frames and the reconstructed frames, yielding frames with uniformly black background for all baseline reconstructions. We then compute PSNR and SSIM on the masked frames.

C Generalization Across Varying Numbers of Camera Views

SABLE is not inherently limited to 2 input views and naturally supports varying numbers of views depending on the camera setup. To demonstrate this flexibility, we considered 2 additional settings: (1) single-view reconstruction on the IBL dataset, and (2) sparse-view reconstruction on the 6-view Cheese3D dataset using only 3 input views.

For IBL, we initialize SABLE from the off-the-shelf model pretrained with 2 input views, as described in Appendix F.1, and fine-tune it on a new animal using a cross-view prediction objective. At inference time, however, we provide only a single input view. Despite the reduction in the number of observed views, the fine-tuned SABLE produces consistent 3D behavioral point clouds from a single image, as visualized in Fig. 7 (A). Thus, the pretrained SABLE model can be adapted to a single-view setting without requiring a new architecture.

For Cheese3D, 6-view reconstruction is straightforward due to the large camera overlap, and we found that the original E-RayZer already performs well. Since SABLE targets sparse-view reconstruction, we intentionally use only 3 of the 6 views, where it still reconstructs 3D point clouds, as visualized in Fig. 7 (B). Together, these experiments demonstrate that SABLE can operate with varying numbers of input views, including both fewer and more than the default 2-view setting.

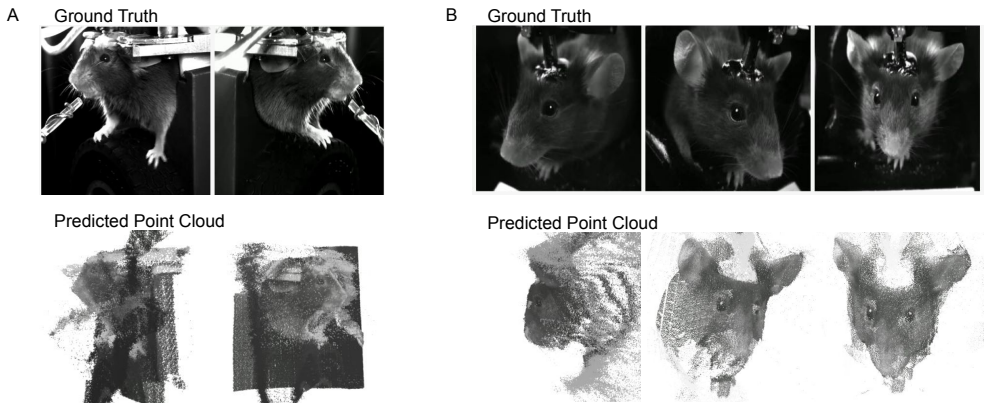

Figure 7: **SABLE with varying numbers of input views.** (A) On IBL, SABLE is fine-tuned using a cross-view prediction objective, where one view is used as input to predict the other view and itself. At inference time, SABLE reconstructs both views from a single input view. (B) On Cheese3D, we restrict the input to 3 of the available 6 views. For both settings, the second row visualizes the reconstructed 3D point clouds from different viewing angles.

D Ablation Study

D.1 Model Size

We benchmark SABLE against diverse representation learning approaches, including a keypoint-based model, a lightweight CNN autoencoder (ResNet-AE-18), and a ViT masked autoencoder (BEAST) (Section 5.2 and 5.3). Wang et al. (2026b) showed that BEAST outperforms the larger ResNet-AE-50 on downstream pose estimation, motivating our choice of BEAST as the primary high-capacity baseline.

However, model size may independently affect performance. We therefore perform a model-size ablation on the IBL dataset, evaluating larger variants of both ResNet-AE (ResNet-152) and BEAST (ViT-Large) using the same encoding and decoding benchmarks. As shown in Table 3, increasing model size does not improve performance for either baseline. Moreover, SABLE outperforms both the best ResNet-AE and BEAST variants across the encoding and decoding metrics. These results indicate that SABLE's performance advantage is not simply explained by scaling the backbone, highlighting the importance of the representation learning architecture in this setting.

Table 3: Neural encoding (bits per spike) and decoding (PSNR, SSIM) across model sizes on the IBL dataset.

Method	Decoding (PSNR) ↑	Decoding (SSIM) ↑	Encoding (bps) ↑
ResNet-AE-18	21.013 ± 0.015	0.853 ± 0.000	0.166 ± 0.005
ResNet-AE-152	19.520 ± 0.017	0.840 ± 0.000	0.152 ± 0.005
BEAST (ViT Base)	21.659 ± 0.009	0.858 ± 0.000	0.165 ± 0.005
BEAST (ViT Large)	21.210 ± 0.008	0.852 ± 0.000	0.102 ± 0.003
SABLE	$\mathbf{22.355 \pm 0.014}$	$\mathbf{0.893 \pm 0.000}$	$\mathbf{0.168 \pm 0.006}$

D.2 Loss Weighting

The loss weights were selected through preliminary hyperparameter tuning and consistently produced high-quality 2D renderings and 3D reconstructions. For fair comparison, we similarly tuned the ResNet-AE and BEAST baselines to achieve strong 2D reconstruction performance before downstream evaluation.

We ablated the three loss weights by varying $\mathcal{L}_{\text{recon}}$, $\mathcal{L}_{\text{percep}}$, and $\mathcal{L}_{\text{geom}}$ with all other settings fixed. In the table, we report the PSNR of the reconstructed view. The model performance is generally robust to moderate changes in the loss weights. Although removing $\mathcal{L}_{\text{geom}}$ or increasing $\mathcal{L}_{\text{recon}}$ slightly improves PSNR, the former causes the reconstructed 3D point clouds to collapse, while the latter introduces noticeable geometric distortions by overemphasizing 2D reconstruction. Among the loss combinations that preserve coherent 3D geometry, our chosen weighting achieves the highest PSNR.

Table 4: Ablation on loss weighting.

Variant	$\mathcal{L}_{\text{recon}}$	$\mathcal{L}_{\text{percep}}$	$\mathcal{L}_{\text{geom}}$	PSNR ↑
Ours	**1.0**	**0.3**	**1.0**	**26.0**
No $\mathcal{L}_{\text{geom}}$	1.0	0.3	0.0	26.9
High $\mathcal{L}_{\text{recon}}$	2.0	0.3	1.0	26.1
Low $\mathcal{L}_{\text{percep}}$	1.0	0.1	1.0	25.9
No $\mathcal{L}_{\text{percep}}$	1.0	0.0	1.0	25.7
High $\mathcal{L}_{\text{percep}}$	1.0	1.0	1.0	24.7
High $\mathcal{L}_{\text{geom}}$	1.0	0.3	2.0	25.6
Low $\mathcal{L}_{\text{recon}}$	0.5	0.3	1.0	25.2

E Model and hyperparameter details

E.1 SABLE

SABLE builds upon the E-RayZer backbone (Zhao et al., 2026), and we use the `erayzer_multi.pt` pretrained checkpoint available on Hugging Face. Unlike the original E-RayZer model, which adopts cross-view masking, SABLE applies patch-level masking across both

views. We set the loss weights in Eq. 7 to $\lambda_{\text{recon}} = 1$, $\lambda_{\text{percep}} = 0.3$ and $\lambda_{\text{geom}} = 1$. Table 5 summarizes the model hyperparameters used in the main experiments.

Table 5: SABLE model hyperparameters.

Hyperparameter	Value
Input image size	320×320
Patch Size	16
Number of Image Tokens	400
Embedding Dimension	768
Number of Attention Heads	12
Image Encoder Depth	16
Geometry Encoder Depth	16
Decoder Depth	8
Neck Depth	1
Mask Ratio	0.5 for IBL; 0.75 for Cheese3D
Gaussian Spherical Harmonics Degree	3
Near Plane	0.2
Depth Range	$(0, 8)$
Scene Scale Factor	1.35

E.2 RESNET AUTOENCODER

The ResNet autoencoder baseline (He et al., 2016) compresses each video frame into a low-dimensional latent vector and reconstructs the input image from this latent representation. The model uses a ResNet-based convolutional encoder followed by an MLP bottleneck and a symmetric decoder for image reconstruction. Table 6 summarizes the model hyperparameters used in the main experiments.

Table 6: ResNet autoencoder model hyperparameters.

Hyperparameter	Value
Input image size	224×224
Image preprocessing	ImageNet normalization
Backbone	ResNet-18
Residual block configuration	$(2, 2, 2, 2)$
Embedding Dimension	768
Bottleneck mapping	$512 \times 7 \times 7 \to 768 \to 512 \times 7 \times 7$
Batch Size	32

E.3 BEAST

The BEAST baseline is built on a ViT-MAE architecture with a ViT-B backbone pretrained on ImageNet and learns video embeddings through masked image reconstruction and temporal contrastive learning. The model tokenizes each frame into image patches with a patch size of 16, applies random patch masking with a standard mask ratio of 75%, and reconstructs the masked regions using a mean squared error (MSE) reconstruction loss. In addition, a frame-level contrastive (InfoNCE) loss encourages temporal consistency between nearby video frames. Table 7 summarizes the model architecture and hyperparameters used in our experiments.

E.4 TCN

For neural encoding and decoding, we use the temporal convolutional network (TCN) architecture from the Facemap study (Syeda et al., 2024). The model processes inputs (behavioral features for encoding or neural activity for decoding) through a hierarchy of temporal convolutional layers with multiple receptive-field scales to capture temporal dynamics across different timescales. The resulting temporal features are passed through feedforward layers to produce compact latent representations, followed by a linear readout layer that predicts neural activity for encoding or behavioral features for decoding. Table 8 summarizes the model architecture and hyperparameters used in our experiments.

Table 7: BEAST model hyperparameters.

Hyperparameter	Value
Input image size	224×224
Patch size	16
Embedding Dimension	768
Encoder layers	12
Attention heads	12
MLP intermediate size	3072
Activation	GELU
QKV bias	Enabled
Decoder hidden dimension	512
Decoder layers	8
Decoder attention heads	16
Decoder intermediate size	2048
MAE mask ratio	0.75
Projection / embedding size	768
Batch Size	32

Table 8: TCN model hyperparameters.

Hyperparameter	Value
Input dimension	Neuron count (decoding) or behavioral feature dimension (encoding)
Temporal filters	10
Temporal kernel size	201
Core layers	2
Intermediate dimension	50
Latent dimension	256
Readout layers	1
Output dimension	Neuron count (encoding) or behavioral feature dimension (decoding)
Temporal filter initialization	Gabor Wavelets
Wavelet nonlinearity	ReLU
Latent nonlinearity	ReLU

F TRAINING DETAILS

F.1 SABLE

SABLE is initialized from a pretrained E-RayZer checkpoint and further pretrained on the IBL two-camera dataset (approximately 7,000 frame pairs from 7 sessions) and the Cheese3D dataset (5,500 frame pairs from 10 sessions), respectively. Unless otherwise stated, each training sample consists of two synchronized camera views and corresponding reconstruction targets. The model jointly infers camera geometry and reconstructs 3D structure through a Gaussian splatting decoder conditioned on multi-view transformer embeddings. Training data from each session are split into training and validation subsets using a 90% and 10% split. The SABLE training objective combines image reconstruction, perceptual, and geometric consistency losses. Optimization is performed using AdamW with a learning rate of 2×10^{-5}, weight decay of 0.05, batch size 24 for IBL and 16 for Cheese3D, and a cosine learning-rate schedule with 15% warmup steps and a cosine division factor of 10. Training is performed for 100 epochs on a single NVIDIA A40 or A100 GPU and takes approximately 17 hours per session.

F.2 RESNET AUTOENCODER

The ResNet autoencoder is trained on the same video frame pairs from the IBL and Cheese3D datasets, as described in Appendix A.1.1 and Appendix A.2.6, respectively. We use an Adam optimizer with a learning rate of 2×10^{-5}, batch size 24, weight decay of 0.05, and a cosine learning-rate schedule with 15% warmup steps and a cosine division factor of 10. Training uses the default image augmentation pipeline from the BEAST code repository and runs for 160 epochs for IBL and 400 for Cheese3D. Training is performed on a single NVIDIA A100 GPU and takes approximately 0.5 hours per session.

F.3 BEAST

The BEAST backbone baseline is trained on the same video frame pairs from the IBL and Cheese3D datasets, as described in Appendix A.1.1 and Appendix A.2.6, respectively. We use a standard mask ratio of 0.75 for training the ViT-MAE backbone. Optimization is performed using Adam with a learning rate of 2×10^{-5}, weight decay of 0.05, batch size 24, and a cosine learning-rate schedule with 15% warmup steps and a cosine division factor of 10. We adopt the default image augmentation pipeline from the BEAST code repository. Training is performed for 160 epochs for IBL and 400 for Cheese3D on a single NVIDIA A100 GPU and takes approximately 1 hour per session.

F.4 TCN

For the TCN encoder and decoder, we perform hyperparameter searches using Ray Tune to identify the best-performing configurations. Each TCN is trained using AdamW for 300 epochs with an MSE loss and a temporal-filter smoothing penalty of 0.5. We sample 30 random hyperparameter configurations, with learning rates drawn log-uniformly from 5×10^{-5} to 2×10^{-3}. We apply a weight decay of 10^{-4} for model training. Model selection is performed using validation metrics (BPS for encoding and R^2 for decoding). We then report results on the test set. Training and hyperparameter search are performed on 2 NVIDIA A40 or A100 GPUs, with the full search taking approximately 0.5 hours.