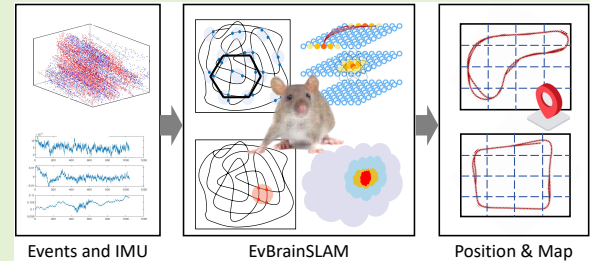


Brain-inspired Simultaneous Localization and Mapping via Integrating Event Camera and IMU

Jiarui Dou¹, Hui Zhao^{1,2}, Fangwen Yu³, Member, IEEE, Xianlei Long¹, Yan Li⁴, Chao Chen¹, Senior Member, IEEE, Xiangwei Zhu⁵, Fuqiang Gu^{1,*}, Member, IEEE

Abstract—Visual Simultaneous Localization and Mapping (SLAM) is a popular positioning method for robots, which has been successfully applied to a variety of tasks such as grasping and unmanned inspection. However, most existing visual SLAM methods suffer from low accuracy, poor robustness, and positioning failure in challenging environments such as texture-less or dim scenes. To address these challenges, we present EvBrainSLAM in this study, a novel brain-inspired visual SLAM that integrates event camera and Inertial Measurement Unit (IMU) for robust pose estimation. EvBrainSLAM converts asynchronous event streams into self-motion and local-view cues that can be effectively integrated by a Continuous Attractor Neural Network (CANN)-based navigation framework. Specifically, the frontend combines motion-compensated event frames with inertial measurements to estimate local motion through sliding-window nonlinear optimization, while the CANN-based backend performs attractor-based path integration, maintains a continuous pose representation, and suppresses localization drift caused by noisy or degraded measurements. Local-view information derived from event-intensity templates is further used to reactivate previously learned pose-cell states and provide loop-closure corrections through a brain-inspired cognitive map, thereby improving global trajectory consistency. By combining the high temporal resolution and wide dynamic range of event cameras with robust attractor-based state estimation, EvBrainSLAM achieves reliable localization under rapid motion, low illumination, and texture-less conditions. Experimental results on nine challenging sequences demonstrate that EvBrainSLAM consistently outperforms the evaluated baseline methods in terms of localization accuracy and robustness.

Index Terms—simultaneous localization and mapping (SLAM), visual navigation, biologically inspired navigation, event camera, intelligent robotics.



I. INTRODUCTION

VISUAL Simultaneous Localization and Mapping (SLAM) is widely applied in robotics and autonomous systems, where accurate and real-time localization is essential for tasks such as autonomous navigation, robotic manipulation, and unmanned inspection [1]–[3]. Representative systems, including ORB-SLAM3 [4] and Kimera [5], perform

well under favorable environmental conditions. However, conventional visual SLAM typically relies on frame-based cameras and may suffer significant performance degradation under severe illumination changes or motion blur.

Event cameras provide a promising sensing paradigm for overcoming these limitations and have attracted increasing attention in visual odometry (VO) and SLAM [6]–[10]. Unlike frame-based cameras, event cameras asynchronously capture brightness changes, offering high temporal resolution and dynamic range for robust localization. Existing event-based VO and SLAM methods generally employ filtering or nonlinear optimization to estimate system states from event measurements, often in combination with inertial data. Although these methods have achieved promising performance, their localization accuracy remains sensitive to measurement noise and insufficient feature correspondences, while accumulated drift can substantially affect long-term trajectory consistency. In parallel, brain-inspired SLAM has emerged as an alternative paradigm for improving localization robustness under uncertain or incomplete observations [11], [12].

Brain-inspired navigation models draw inspiration from the neural mechanisms of place cells in the hippocampus,

* Corresponding Authors.

¹ Jiarui Dou, Xianlei Long, Fuqiang Gu and Chao Chen are with the College of Computer Science, Chongqing University, Chongqing, China. Emails: {doujr, xianlei.long, cschaochen, gufq}@cqu.edu.cn

² Hui Zhao is with the College of Computer Science, China University of Geosciences, Wuhan, China. Email: zhaohui@cug.edu.cn

³ Fangwen Yu is with the Department of Precision Instrument, Tsinghua University, Beijing, China. Email: yufangwen@tsinghua.edu.cn

⁴ Yan Li is with the School of Computing, Macquarie University, Sydney, Australia. Email: y.li@mq.edu.au

⁵ Xianwei Zhu is with the School of Electronics and Communication Engineering, Sun Yat-Sen University, Shenzhen, China. Email: zhuxw666@mail.sysu.edu.cn

This work is supported by National Key Research and Development Program for Young Scientists (No. 2025YFB4711500), National Natural Science Foundation of China (No. 42474027, 62403085, 62322601), Xiaomi Young Talents Program, and China Postdoctoral Science Foundation (No. 2026M791878).

grid cells in the entorhinal cortex, head-direction cells, and speed cells [13]–[17]. These neural cells jointly encode spatial position, movement direction, and motion velocity, providing biological inspiration for continuous spatial representation and path integration. Based on these mechanisms, systems such as RatSLAM [11], [18], BatSLAM [19], and NeuroSLAM [12] employ Continuous Attractor Neural Networks (CANNs) to integrate self-motion and environmental cues, maintain an internal representation of the robot pose, and correct accumulated localization errors through familiar-scene recognition.

CANNs maintain a localized activity packet through recurrent excitation and inhibition [20]. The position of the dominant activity packet represents the internal pose estimate, while external motion and visual information drive its propagation and correction. This attractor-based mechanism naturally smooths noisy inputs and resolves competing observations, thereby improving the stability of pose estimation under temporarily degraded sensing conditions. However, existing brain-inspired SLAM systems based on conventional cameras, LiDAR, GNSS, or sonar remain constrained by the characteristics of their sensing modalities [11], [12], [19], [21]–[23]. In particular, conventional visual sensors often become unreliable under rapid motion, low illumination, or motion blur.

In this work, we propose EvBrainSLAM, an event-driven brain-inspired SLAM system that integrates event and inertial measurements for robust localization in challenging environments. EvBrainSLAM adopts a hybrid frontend-backend architecture that combines optimization-based motion estimation with attractor-based pose representation. The frontend fuses event measurements and IMU data through sliding-window nonlinear optimization to estimate local self-motion. The estimated motion is then provided to the CANN-based backend, which performs attractor-based path integration, maintains a continuous pose representation, and incorporates local-view information to correct accumulated localization errors through a brain-inspired cognitive map. The recurrent dynamics of the CANN improve pose continuity and robustness against temporarily degraded frontend measurements, while the high temporal resolution and wide dynamic range of the event camera enable reliable perception under rapid motion, low illumination, and weak-texture conditions. To convert asynchronous event streams into navigation information, motion-compensated event frames are constructed for self-motion estimation, whereas event-intensity templates are generated for local-view matching and loop-closure detection. The resulting local-view information reactivates previously learned pose-cell states, reducing accumulated path-integration errors and improving the global consistency of the cognitive map. In summary, the main contributions of this study are as follows:

- We propose EvBrainSLAM, a novel event-driven and brain-inspired SLAM method that leverages event camera and the CANN to achieve robust performance in challenging environments like texture-less or dim environments.
- We integrate IMU motion priors with spatiotemporally coherent event frames to establish a dynamic feature-association mechanism for robust self-motion estimation, providing reliable motion information for the CANN-based backend.

- We propose an event-intensity-map matching mechanism to reconstruct the local-view cell activation model, enabling loop-closure detection for pose correction and bridging the representation gap between asynchronous event streams and brain-inspired navigation.
- We evaluate the proposed method through extensive experiments on nine challenging sequences. Experimental results demonstrate that the proposed method consistently outperforms state-of-the-art methods in term of positioning accuracy and robustness.

II. RELATED WORK

A. Visual SLAM

Visual SLAM methods have undergone significant development since the early 21st century. In 2007, Davison et al. introduced the first real-time Mono-SLAM [24] system, which utilized an extended Kalman filter to estimate the camera's motion and 3D elements in the world. In the same year, Klein et al. proposed the PTAM [25] framework, dividing visual SLAM into two independent threads of tracking and mapping. It introduced Bundle Adjustment for joint optimization of camera pose and 3D map, becoming a benchmark for subsequent work and a basis for categorizing pose optimization methods. Later on, various multi-source fusion solutions emerged, including the DTAM [26] framework introducing depth maps and SLAM++ [27] being the first to integrate semantic information. With the maturation of visual SLAM technology, researchers began focusing on enhancing system performance and accuracy. Forster et al. introduced the SVO [28] framework combining feature and direct methods, and Engel et al. presented the first sparse direct visual SLAM algorithm, DSO [29]. ORB-SLAM3 proposed [4] by Carlos et al. provides a significant update to the system, enabling it to better handle multiple types of sensor data and achieve more robust performance, making it applicable in a variety of environments. While visual SLAM methods have been widely applied in a variety of tasks, they cannot work well in highly dynamic scenarios due to the limitation of conventional cameras, such as susceptibility to light, motion blur, and low dynamic range.

To deal with the challenge of achieving accurate positioning in challenging scenarios, some event-based VO or SLAM methods have been proposed. An early work on event-based visual odometry is introduced in [30], which initially detects features from grayscale frames and subsequently asynchronously tracks these features using event streams. EVO [7] introduced a parallel tracking and mapping event-based monocular visual odometry system. It applies image-to-model alignment to tracking and utilizes event-based Multi-View Stereo (EMVS) [31] for mapping. In [32], an event-based visual-inertial odometry is developed to address the issue of incomplete scale estimation and provide accurate state estimation based on Extended Kalman Filtering. UltimateSLAM [8] combines event streams, image frames, and IMU measurements through non-linear optimization to achieve accurate pose estimation, which leverages the complementary advantages of event cameras and standard cameras. Apart

from the above works, there are other event-based VO or SLAM methods such as DEVO [33], ESVIO [9], and EDS [34]. Although plenty of event-based VO or SLAM methods have been proposed, these methods either use filtering-based methods or graph optimization methods for pose estimation, which may not well deal with uncertainty in the environment. In this work, we attempt to achieve robust positioning and mapping with event cameras and brain-inspired methods.

B. Bio-inspired SLAM

To cope with uncertainty in challenging environments, some bio-inspired SLAM methods have been proposed. Bio-inspired SLAM methods draw inspiration from biological systems, often mimicking the sensory and cognitive processes found in animals or humans. So far, many types of spatial navigational cells have gradually been discovered, including place cells [13], grid cells [14], head direction cells [15], speed cells [16], etc. Based on these discovered navigational cells, several bio-inspired navigation systems have been developed. An early bio-inspired SLAM system is RatSLAM [18] that draws inspiration from the neuroscientific understanding of the rodent brain. It employs a competitive attractor network (CAN) to seamlessly merge odometric data with landmark sensing to construct a coherent representation of the surrounding environment. Encouraged by the success or promise of RatSLAM in plenty of tasks such as autonomous exploration, search and rescue missions, many variants have been developed, including BatSLAM [19], DolphinSLAM [35], and NeuroSLAM [12].

Compared to conventional SLAM methods, bio-inspired SLAM methods offer several advantages. First, bio-inspired SLAM approaches often model the flexibility and adaptability observed in biological systems, making them more robust in dynamically changing environments. Besides, bio-inspired SLAM algorithms often incorporate mechanisms for handling uncertainties, such as sensor noise or changes in the environment, allowing for more adaptive and reliable navigation. In addition, bio-inspired SLAM often includes learning mechanisms and plasticity, allowing robots to improve their navigation performance over time. While some bio-inspired SLAM methods are attractive and have great potential for dynamic environments or resource-limited scenarios, existing works mainly use the conventional cameras to estimate the robot's pose. This makes it inevitably limited by the issues of conventional cameras such as motion blur, high time latency, and lower dynamic range. In this work, we investigate the feasibility of implementing a bio-inspired SLAM with event cameras.

III. PROPOSED METHOD: EVBRAINSLAM

The proposed method is inspired by the RatSLAM, which mimics the sensory and cognitive processes found in the rodent brain. Different from RatSLAM, the proposed EvBrainSLAM relies on event cameras to estimate the pose of the robot, which makes it capable to work accurately and robustly in dynamic and complex scenarios. To make the system more accurately and robustly, we also integrate IMU into EvBrainSLAM. The system architecture of the EvBrainSLAM is graphically

demonstrated in Figure 1. It consists of four main components: Motion Compensation module for data association, Self-Motion Cues, Local View Cues for loop detection, Pose Cell Network for path integration. EvBrainSLAM takes as input event streams and IMU measurements, and outputs the robot's position and map around the robot. Specifically, the Motion Compensation module integrates the IMU measurements with events to generate a synthesized event frame that is clearer than the event frame obtained by simply accumulating events. These synthesized event frames are then fed into the Self-motion Cues module and the Local View Cues module.

The Self-motion Cues module, analogous to the frontend of traditional SLAM, employs the synthesized event frames and IMU measurements via a sliding window non-linear optimization method to estimate the velocity of a robot. Such velocity information is then fed to the Pose Cell Network to estimate the position of the robot and construct the surrounding map.

The Local View Cues module utilizes the synthesized event frames to create visual templates for loop closure using the scanline intensity profile algorithm [36]. This is done by checking for the matching between the current template and historical templates. When the matching score falls into a predefined threshold, a loop closure is detected. This activates specific pose cells associated with the corresponding local view cells, injecting activity to provide local view cues. In the way, the accumulated positioning error can be corrected.

The Pose Cell Network continuously encodes self-motion cues (translational velocity and rotational velocity) to update cell activities, simulating the behavior of neurons in the brain of rodents to facilitate spatial perception. The Pose Cell Network is implemented by a continuous attractor neural network. The process of encoding self-motion information is also known as path integration, where the drive of external information leads to the propagation of activity packets among pose units in the network. After the encoding process is completed, the most active centroid in a region represents the robot's best internal estimate of its current pose. By integrating continuous spatial positions and scene information, an experience map with a topological structure is formed.

IV. DETAILS OF EVBRAINSLAM

In this section, we provide a detailed description of the proposed method and elaborate on each of the four key components: Motion Compensation, Self-Motion Cues, Local View Cues, and Pose Cell Network.

A. Motion Compensation

The motion compensation module correlates event data for self-motion cue estimation and local view cue extraction. Given that planar motion triggers uneven and sparser event data and local view cells require event view templates, we adopt a more stable motion compensation method in [37] to align and accumulate event data into event frames within a time window. If we directly simply accumulate events to generate event frames without motion compensation, the performance of estimating velocities and loop closure will be significantly affected. Figure 2(a) displays the instantaneous event

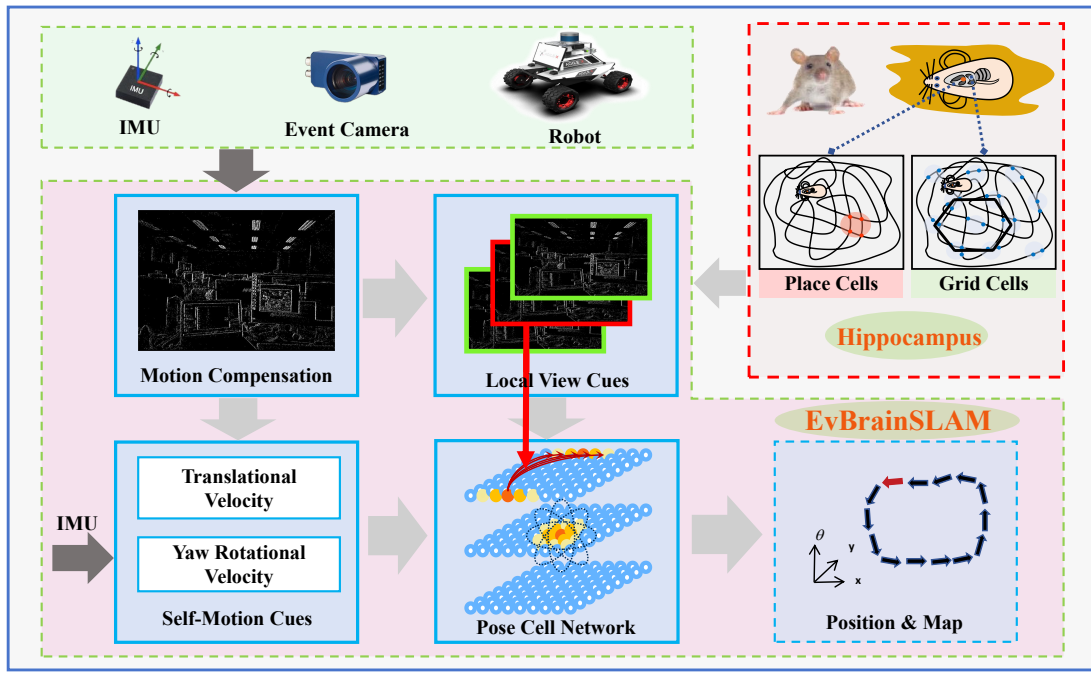


Fig. 1. Overview of the proposed EvBrainSLAM. It is implemented using an event-driven computational model of navigation cells.

accumulation image, while Figure 2(b) shows the synthesized event frame obtained through motion compensation. From the Figure 3, we can see that the synthesized event frame is clearer than the event frame generated by simply accumulating events. It can hence result in a better performance for self-motion cue estimation and local view cue estimation.

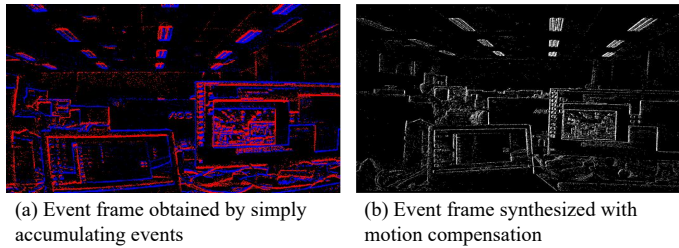


Fig. 2. The left side displays the event frame generated by rapidly accumulated events, while the right side shows the synthesized event frame with motion compensation.

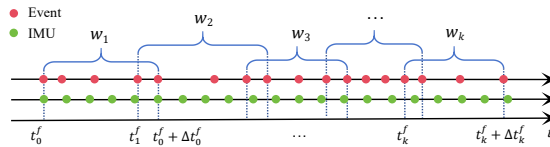


Fig. 3. The triggered events (denoted by red solid circles) are partitioned into temporal windows w_k at different timestamps. Each window w_k starts from t_k^f and ends at $t_k^f + x$, where the window size is adaptively adjusted based on the density of triggered events.

Specifically, for timestamped event stream and IMU measurements, we segment the data streams using overlapping temporal windows as illustrated in Figure 3. The window size is dynamically adjusted based on the event-triggering rate, with a predefined maximum event count threshold. To

align discrete events at unified timesteps, we project the pixel coordinates of all events within each window to the coordinate system corresponding to the window's initial timestamp using linear interpolation of IMU data. By accumulating these corrected events, a synchronized composite event frame is ultimately generated.

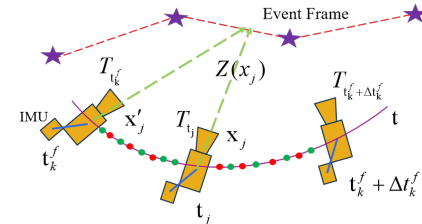


Fig. 4. Motion compensation process of events. Green dots are collected events, red dots are IMU measurements, stars are the event frames.

Then, the motion compensation process of events and IMU measurements is shown in Figure 4. Let t_k^f and $t_k^f + \Delta t_k^f$ represent the start and end time of the k_{th} time window W_k . Events within the k_{th} window are denoted by $W_k = \{e_{k_i}, \dots, e_{k_{i+N-1}}\}$. In the time window W_k , each event e_k at different moments will be aligned with that at the first moment, and the aligned variable is the pixel coordinates. The event correction using IMU measurements is formulated as follows:

$$x'_j = \pi(T_{t_k^f, t_j}^f(Z(x_j)\pi^{-1}(x_j))), \quad (1)$$

where x_j and x'_j represent the pixel coordinates before and after the event correction, respectively. The function $\pi(\cdot)$ represents the camera projection model, while $T_{t_k^f, t_j}^f$ denotes the SE(3) transformation capturing rigid body changes over time, interpolating between the IMU measurements $T_{t_k^f}^f$ and

$T_{t_k^f + \Delta t_k^f}$. $Z(x_j)$ represents the median depth of the current landmarks. Thus, we can aggregate the corrected events within the window W_k to construct an event frame.

To better extract robust features from event frames, we apply a median filter on the synthesized event frames to remove event noises. We found in the experiment that the filtering process can improve the stability of feature point tracking. Figure 5 visualizes the results before and after using the filtering method on the feature points.

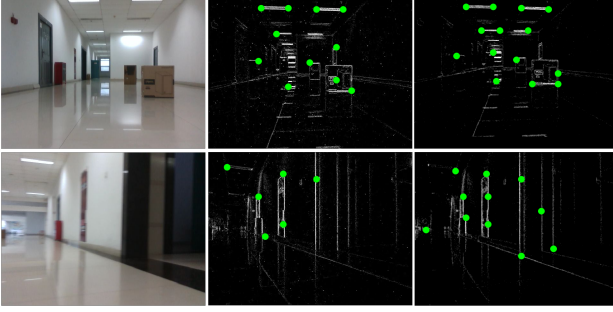


Fig. 5. The left column is the frame image. The middle column is the noisy event frame with some salt noise. The right column is the processed event frame after filtering. Green dots denote feature points.

B. Self-Motion Cues

The self-motion cues module utilizes synthesized event frames to estimate visual self-motion cues containing translation and rotation velocity information, which serve as inputs to the pose cell network and drive them to perform path integration. To accelerate the calculation of self-motion cues, we adopt the commonly-used sliding-window nonlinear optimization method to construct the reprojection error terms of visual-inertial odometry within a local window. Specifically, we first conduct feature extraction and tracking on the event frame, and apply the RANSAC algorithm to remove outliers and evaluate whether the current frame can be used as a keyframe. Then, these features with good tracking quality are triangulated to form landmarks. After that, we then use the OKVIS [38] to combine weighted reprojection error e_r and IMU residuals e_s . Mathematically, the keyframe-based minimization problem can be formulated as (2).

$$J := \sum_{k=1}^K \sum_{j \in \mathcal{J}(k)} e_r^{j,kT} W_r^{j,k} e_r^{j,k} + \sum_{k=1}^{K-1} e_s^{kT} W_s^k e_s^k, \quad (2)$$

where k and j denote the frame and landmark indices, $W_r^{j,k}$ and W_s^k represent the information matrix for the j^{th} landmark in the k^{th} frame and the k^{th} IMU residuals, respectively.

C. Local View Cues

Local view cues play an important role in loop closure detection, which can correct the accumulated error of odometry. During movement, the robot continuously records historical scenes as event frames stored in local view cell units, which serve as local view cue information. As shown in Figure 6, when a novel scene A is observed familiar with previously-visited scenes A', the current local view will match it with the

historical views to activate the corresponding pose cells, as shown in Figure 6(b). After completing one round of attractor competition in the pose cell network, loop closure correction can be performed on the trajectory. Specifically, the incoming event frames are first translated into a visual template, then the template is compared with these templates that were seen and stored in a database through the visual template matching algorithm [11].

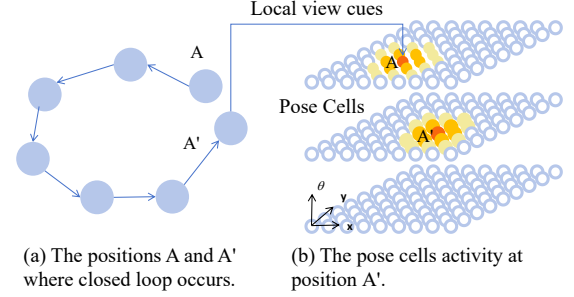


Fig. 6. When loop closure occurs in the current scene A', local view cues are sent to the pose cell network to activate the cellular activity of the historical scene A.

The pseudo-code of the algorithm is given in Algorithm 1. The algorithm takes as input the synthesized event frame ν , and outputs the local view cues. It first initializes a set V that stores local view cells, and sets a matching threshold ε . Then, the input event frame ν is cropped into multiple sub-sampling regions ν' to eliminate redundant information such as visual monotonic features such as roads. Next, we count the total number of pixels in each subregion as pixel intensity, and then normalize the pixel intensity to maintain the contrast of each subregion, taking into account local and global effects. This process corresponds to the templating in line 3.

Algorithm 1: Local view cues generation

Input: Event frame ν
Output: Local view cues V^*

- 1 Initialize the current local view cell set $V \leftarrow \emptyset$, and threshold ε ;
- 2 Crop ν into multiple sub-sampling regions ν' ;
- 3 Templating $\nu' \rightarrow V'$;
- 4 **for** $V_i \in \{1, \dots, V\}$ **do**
- 5 value = SAD(V_i, V');
- 6 **if** value $< \varepsilon$ **then**
- 7 $V^* \leftarrow V_i$;
- 8 **return** V^* ;
- 9 **else** $V = V \cup V'$;

Finally, we calculate the Sum of Absolute Differences (SAD) between the visual template of the current event frame and all previous visual templates. If the value is below a threshold ε , the corresponding learning template is selected. Otherwise, the current visual template is added to the template database. The local view cues V^* represents the output based on selected template. This activates specific pose cells associated with the corresponding local view cell and injects

its activity, which is then passed to the pose cell network for further processing.

D. Pose Cell Network

The Pose Cell Network is responsible for providing robust position estimation, which is implemented as a CANN that converges to a stable pattern of activation across its units. Each unit excites nearby units while inhibiting those further away. This pattern leads to a concentration of activity, resulting in an activity packet eventually dominating. The centroid of an activity packet represents the robot's best internal estimate of its current pose. The network structure is shown in Figure 7, corresponding to the robot's coordinate $x, y \in \mathbb{Z}$ and direction $\theta \in \mathbb{Z}$.

The renewal of cell activity is based on competition of internal attractor dynamics, as well as stimulation of external cues. Attractor dynamics maintain the renewal of the network activity through local excitation, global inhibition, and normalization of cells in turn. The pose network takes as input self-motion cues and local view cues, and outputs the position of the robot. Specifically, it performs path integration to update cell activity based on self-motion cues from the event camera, and executes loop closure to reduce cumulative errors by encoding local view cues from visual templates. These encoding process all transmits cell data activity through inhibitory or excitatory connections. In addition, the pose cell network will gradually form a cognitive map in conjunction with the local view cues to represent the robot's historical trajectory information.

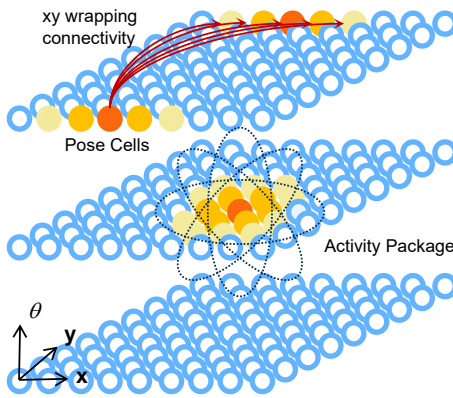


Fig. 7. The pose cell network model is implemented by a continuous attractor network of a three-dimensional matrix (coordinate x , y and direction θ), which is able to maintain the position of the robot on a two-dimensional plane, and the boundary of each network layer is connected to the other side.

1) Attractor dynamics: The pose cell network goes through the competitive attractor dynamic process. There are three stages to the dynamics: local excitation stage, global inhibition stage, and normalization stage.

In the local excitation stage, we use a weight matrix ε that conforms to a 3D Gaussian distribution to represent the range of cell activation in three dimensions. The weight ε is

calculated by the following formula:

$$\varepsilon_{i,j,k} = \frac{1}{\delta\sqrt{2\pi}} e^{(-u^2-v^2-w^2)/(2\delta^2)} \quad (3)$$

where δ represents the covariance constant, u, v and w denote the relative positions of the two active cell units, respectively. The activity change ΔP in this stage is calculated by:

$$\Delta P_{x,y,\theta} = \sum_{i=0}^{n_x} \sum_{j=0}^{n_y} \sum_{k=0}^{n_\theta} \varepsilon_{i,j,k} P_{(x+i)(y+j)(\theta+k)}, \quad (4)$$

where P is the activity level of the cell, n_x, n_y, n_θ are the dimensions of the matrix ε , respectively.

In the global inhibition stage, each pose cell unit inhibits the surrounding cells through a process of local inhibition. We use a local inhibition weight matrix γ (similar to ε mentioned above) to reduce the activity level of the cells during the local inhibition stage. Then, all pose cells are uniformly renewed by global inhibition φ . The local and global inhibition processes cause changes in cell activity, and the results are calculated as follows:

$$\Delta P_{x,y,\theta} = \max\left[\sum_{i=0}^{n_x} \sum_{j=0}^{n_y} \sum_{k=0}^{n_\theta} \gamma_{i,j,k} P_{(x+i)(y+j)(\theta+k)} - \varphi, 0\right]. \quad (5)$$

In the normalization stage, in order to maintain the total activity of the cell network to one, we perform normalization operations, and the activity level of each cell is calculated using the following formula:

$$P_{x,y,\theta} = \frac{P_{x,y,\theta}}{\sum_{x=0}^{N_x} \sum_{y=0}^{N_y} \sum_{z=0}^{N_\theta} P_{x,y,\theta}}, \quad (6)$$

where N_x, N_y and N_θ represent the number of cells in the three dimensions of the network, respectively.

2) Path integration: After the dynamic process of competing attractors, the pose cell network performs the path integration based on the self-motion cues, projecting the activity of each pose cells into surrounding cells. The magnitude of cell activity transferred in the dimension (x, y, θ) depends on the velocity (v_x, v_y, ω) . The variable ΔP for each cell in the network can be formulated as:

$$\Delta P_{x,y,\theta} = \sum_{i=\delta_{x_0}}^{\delta_{x_0}+1} \sum_{j=\delta_{y_0}}^{\delta_{y_0}+1} \sum_{k=\delta_{\theta_0}}^{\delta_{\theta_0}+1} \eta_{i,j,k} P_{(x+i)(y+j)(\theta+k)}. \quad (7)$$

where P is the activity level of the cell sending the packet, η is a residual that depends on the fractional parts of the offsets $\delta_{x_0}, \delta_{y_0}, \delta_{\theta_0}$. The residual η can be calculated as follows:

$$\eta_{i,j,k} = f(\delta_{x_f}, x - \delta_{x_0}) f(\delta_{y_f}, y - \delta_{y_0}) f(\delta_{\theta_f}, \theta - \delta_{\theta_0}), \quad (8)$$

$$f(a, b) = \begin{cases} a, & \text{if } b = 1 \\ 1 - a, & \text{if } b = 0 \end{cases}.$$

The fractional parts of the offsets $\delta_{x_0}, \delta_{y_0}$ and δ_{θ_0} can be calculated as follows:

$$\begin{bmatrix} \delta_{x_0} \\ \delta_{y_0} \\ \delta_{\theta_0} \end{bmatrix} = \begin{bmatrix} [k_x v_x] \\ [k_y v_y] \\ [k_\theta \omega] \end{bmatrix}, \quad (9)$$

$$\begin{bmatrix} \delta_{x_f} \\ \delta_{y_f} \\ \delta_{\theta_f} \end{bmatrix} = \begin{bmatrix} k_x \delta_x / \cos\theta - \delta_{x_0} \\ k_y \delta_y / \sin\theta - \delta_{y_0} \\ k_\theta \omega - \delta_{\theta_0} \end{bmatrix},$$

where k_x, k_y and k_ω are constants.

3) *Local view calibration*: Since the error of path integration increases gradually over time, we need to calibrate the error periodically. Hence, we use local view cues to correct the accumulated error and achieve a closed loop. Once the robot encounters a similar scene, the corresponding local view cell is activated, transmitting activity packets to the pose cell through the link τ that is a learning connection between local view cells and pose cells obtained through the Hebb's rule [39]. The connection matrix ψ stores the learned connections between local view cells and pose cells. The activity change $\Delta P_{x,y,\theta}$ of pose cells can be described as (10), where δ is a local scene correction intensity and V_i is the local view cell activity, which is usually set to a constant.

$$\begin{aligned}\psi_{i,x,y,\theta}^{t+1} &= \max(\tau V_i P_{x,y,\theta}, \psi_{i,x,y,\theta}^t), \\ \Delta P_{x,y,\theta} &= \delta \psi_{i,x,y,\theta}^{t+1} V_i.\end{aligned}\quad (10)$$

4) *Cognitive map*: The establishment of an experience map is a cognitive process that records the historical trajectory of the robot, forming a topologically structured map. Each node in the experience map is a tuple consisting of a pose cell, a local view cell, and a position in experience map, which is defined as follows:

$$e_i = (P^i, V^i, p_i^{exp}), \quad (11)$$

where P^i and V^i represent the pose cells and local view cells at that time, respectively, and $p_i^{exp} = (x, y, \theta)^T$ is the location in the experience map. The creation of an experiential map node e_i to a new experiential map node e_j contains pose changes Δp^{ij} , local view cue V^j , and pose cell P^j . An edge l_{ij} is also formed between e_i and e_j that records the change of position Δp^{ij} and time Δt^{ij} . This process can be defined as:

$$\begin{aligned}e_j &= (P^j, V^j, p^j + \Delta p^{ij}), \\ l_{ij} &= (\Delta p^{ij}, \Delta t^{ij}).\end{aligned}\quad (12)$$

To quantify the similarity between the current state and the existing experience map, a score S is used. If the score exceeds a certain threshold, a new node is created in the experience map. The score S^i for each node is calculated as follows:

$$S^i = \mu_p |P^i - P| + \mu_v |V^i - V|, \quad (13)$$

where μ_p and μ_v correspond to the weights of the pose and local view, respectively.

When a loop closure occurs, a new connection is established in the experience map. To reduce the accumulation of pose errors, a graph relaxation algorithm [11] is used to distribute the accumulated odometry error along the trajectory and update each experience node p . The pose transformation Δp for each node is expressed as:

$$\Delta p^i = \alpha \left[\sum_{j=1}^{N_e} (p^j - p^i - \Delta p^{ij}) + \sum_{k=1}^{N_l} (p^k - p^i - \Delta p^{ki}) \right], \quad (14)$$

where α represents a correction rate constant, while N_e and N_l denote the number of experience nodes e and links l , respectively. After the pose transformation Δp is updated, the pose estimate is obtained, and hence the accumulated error is corrected.

V. EXPERIMENTS AND RESULTS

In the experimental section, we first introduce the experimental platform, datasets and evaluation metrics. Subsequently, we describe the baseline comparison methods and experimental setting before presenting experimental analysis and ablation experiments.

A. Experimental Platform

The robotic platform employed in this study was a Agilex SCOUT MINI Off-road Edition (dimensions: 627 mm × 550 mm × 252 mm; curb weight: 20 kg; payload: 10 kg), featuring independent four-wheel drive with a maximum speed of 10.8 km/h, zero-radius steering and independent swing arm suspension. Computational tasks are powered by an NVIDIA AGX Xavier unit (8-core ARM v8.2 CPU, 512-core Volta GPU). For data acquisition, the platform integrates a DVXplorer event camera (640×480 resolution), an 8 kHz IMU, an Intel RealSense D435 depth camera, and an RS-Helios-16P LiDAR sensor. All experiments were conducted on a computer with 32 GB of RAM and an Intel Core i7-8300U processor.

TABLE I

DESCRIPTION OF THE DATASETS FOR DIFFERENT SEQUENCES.

Sequence	Length	Max velocity	Illumination	Texture
01_bright_low	73.7	1.0	bright	rich
02_bright_fast	103.2	2.0	bright	rich
03_dark_low	60.4	1.0	dark	rich
04_dark_fast	68.3	2.0	dark	rich
05_dark_very	67.1	3.0	very dark	rich
06_dark&bright	73.5	2.0	dark&bright	rich
07_long_time	226.1	2.0	bright	rich
08_corridor1	120.9	2.0	bright	sparse
09_corridor2	132.1	2.0	bright	sparse

B. Datasets and Evaluation Metrics

The dataset was collected in indoor rooms and corridors, corresponding to Figures 8 and 9, respectively. We created various conditions including well-lit, dark, texture-less, motion-blurred, and overexposed scenarios by controlling lighting and movement speeds. The robot equipped with different sensors collected data under different conditions, with its ground truth trajectories derived from LiDAR-based localization. The maximum moving speeds were set to 1 m/s, 2 m/s, and 3 m/s, corresponding to "low", "fast" and "very" in sequences 01-07 of Table I. Sequences 01-07 in Table I were collected in indoor rooms, where "bright" and "dark" mark lighting conditions. Among them, Sequence 07 ran for nearly three minutes under good lighting with a maximum speed of 2 m/s. Sequences 08 and 09 were collected in corridors, covering scenarios with less textures and varying illumination. Sequences 01-06 have an average length of 74 meters, while Sequence 07 extends to 220 meters, and sequences 08 and 09 each reach 120 meters. The dataset will be made available online.

Since the experiments were conducted using a robot moving on a plane, we only considered the translation of the plane and the rotation on the yaw axis. To compare the proposed method

with baseline methods, we use the Absolute Pose Error (APE) and Relative Pose Error (RPE) as the performance metrics. APE measures the overall performance of a SLAM system by comparing the absolute difference between the estimated poses and the ground truth poses. RPE is evaluated by comparing relative pose transformations between consecutive frames [40].



Fig. 8. The indoor environment includes good lighting condition, motion blur conditions and dark condition.



Fig. 9. The experimental conditions covered by the corridor: a) A well-lit corridor; b) Texture-less scene; c) Motion blur caused by rapid movement and dark environments; d) An overexposed environment.

C. Baseline and Experimental Setting

The proposed EvBrainSLAM undergoes benchmarking against not only the most widely adopted bio-inspired RatSLAM [11] but also conventional optimization-based UltimateSLAM approach [8]. RatSLAM [11] utilizes a rodent hippocampal model based on traditional visual sensors, employing a continuous attractor network composed of pose cells

to represent the robot's orientation. UltimateSLAM [8] is a multi-source fusion method, integrating standard frames, event data, and inertial data. It employs a graph optimization method for pose estimation. In our experiments, we implemented UltimateSLAM using only inertial and event data for fair comparison. We have also evaluated EVO [7] on our dataset, but it failed across all sequences. The same problem was encountered in [9], and therefore we did not adopt the EVO for comparison.

TABLE II

THE PARAMETERS ASSOCIATED WITH POSE CELL MODEL AND ULTIMATESLAM.

Pose Cell Model		UltimateSLAM	
Parameters	Values	Parameters	Values
Network size $N_x * N_y * N_\theta$	11*11*36	Event packet size	50000
Cell size	1.0	Norm factor	3.0
Excitatory constant N_{excit}	7×7	Interval size	30
Inhibition constant N_{inhib}	5×5	Use time interval	true
Excitatory variance E_{var}	1.0	Event frame size	30000
Inhibition variance I_{var}	2.0	Noise rate	20000
Global inhibition constant φ	0.0002	Motion correction	true

Table II illustrates the parameters associated with pose cell model and UltimateSLAM. For pose cell model, the parameters include size of the pose cell network $N_x * N_y * N_\theta$, the size of individual cells, excitatory constant N_{excit} , inhibition constant N_{inhib} , excitatory variance E_{var} , inhibition variance I_{var} , and global inhibition constant φ . Note that the RatSLAM shares some parameters as the proposed method, and these parameters were set to the same values. For UltimateSLAM, we need to set the event packet size, frame norm factor, interval size between event packets, whether to use time interval, event frame size, noise rate and whether to use motion correction.

D. Results and Analysis

1) *Comparison with Representative SLAM Baselines:* We first conduct comprehensive evaluations of APE and RPE in all test sequences (01-09). As quantitatively demonstrated in Table III, our method demonstrates superior performance over other approaches in various challenging scenarios, benefiting from the combined advantages of event-driven sensing and bio-inspired neural processing mechanism. RatSLAM, equipped with a traditional camera, seems to be vulnerable to lighting variations and motion blur, failing to generate reasonable trajectory estimations particularly on sequences 03, 04, and 05. While UltimateSLAM successfully predicts trajectories for all sequences with relatively low errors based on event streams, it suffers from significant cumulative errors due to the absence of loop closure mechanism.

Next, we describe the specific performance of the three methods in detail. Sequence 01 is under good lighting conditions with low-speed movement, on which all methods ran successfully. Our method outperforms other methods, the achieved APE and RPE are lower than RatSLAM by 18.9% and 4.4% and UltimateSLAM by 66.7% and 31.2%, respectively. On Sequence 02, as the robot's motion speed

TABLE III

TEST RPE (M) AND APE (M) COMPARISON OF DIFFERENT METHODS ON NINE SEQUENCES.

Sequences	RatSLAM		UltimateSLAM		Ours	
	APE	RPE	APE	RPE	APE	RPE
01_bright_low	0.781	0.113	1.926	0.157	0.641	0.108
02_bright_fast	1.193	0.117	3.204	0.113	0.626	0.087
03_dark_low	-	-	2.746	0.137	0.670	0.093
04_dark_fast	-	-	1.713	0.144	0.425	0.084
05_dark_very	-	-	2.532	0.123	1.561	0.103
06_dark&bright	2.934	0.162	1.681	0.143	1.124	0.091
07_long_time	1.231	0.163	3.961	0.135	0.647	0.123
08_corridor1	4.931	0.172	2.473	0.176	2.641	0.127
09_corridor2	3.521	0.147	2.362	0.172	1.635	0.133

increases, the performance of other methods degrades, while our method becomes even more superior. This can be attributed to the higher speed triggering more events, and our method can provide more robust pose estimation. For sequences 03, 04, and 05 under low-illumination conditions, RatSLAM's reliance on a simplistic visual odometry renders it vulnerable to motion-induced blur artifacts and photometric variations, ultimately failing to construct reasonable topological trajectories. By contrast, our method is less affected by external environmental factors, and with the effective loop closure, it demonstrates significant advantages on these sequences, followed by UltimateSLAM. However, we also found that in extremely dark environments, the event camera is slightly affected, and the number of triggered events is less than in the scene with brighter lighting. This can be demonstrated on Sequences 05 and 06 including some regions with good lighting and dim lighting, where the APE of the proposed method is also relatively high though it still outperforms other methods. On Sequence 07, which has a longer runtime of approximately 3 minutes, our method remains the best, while RatSLAM performs slightly better than UltimateSLAM due to the use of loop closure detection. The sequences 08 and 09 were captured in a corridor, imposing certain challenges of less textures and poor lighting in some regions, and the measured APE is generally higher. In summary, our method performs the best in all sequences except for sequence 08, where the APE is slightly worse than UltimateSLAM. This may be due to larger corrections during loop closures, but overall, it maintains a good topological trajectory.

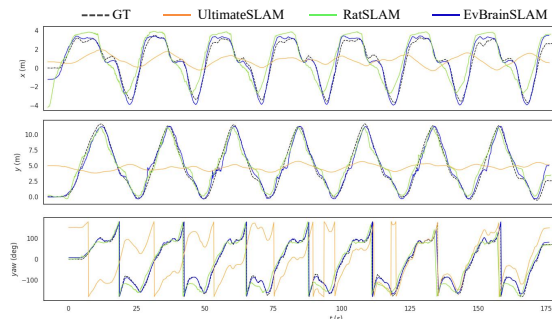


Fig. 10. Real-time comparison was made for different methods along each axis with the ground truth to assess their alignment.

To better quantify the trajectory error, we analyze the trajectory error variation of sequence 07 with a longer running time, as shown in Figure 10. This analysis measures the deviation between the estimated trajectory and the ground truth along each axis in real time. It can be observed that the proposed method (black curve) closely follows the ground truth trajectory, particularly along the y-axis and yaw axis. Compared with RatSLAM, our method achieves lower trajectory deviations, while UltimateSLAM suffers from significant cumulative errors along multiple axes due to the absence of loop closure. For fair comparison, we further evaluate the relative pose consistency using RPE, since UltimateSLAM does not incorporate a loop-closure mechanism. Figure 11 presents the RPE comparison between EvBrainSLAM and UltimateSLAM on sequences 01, 08, and 09. The proposed method achieves consistently lower relative pose errors, indicating smaller pose variations between consecutive frames. This improvement benefits from the stable CANN, which provides more reliable motion estimation and maintains local pose consistency during navigation.

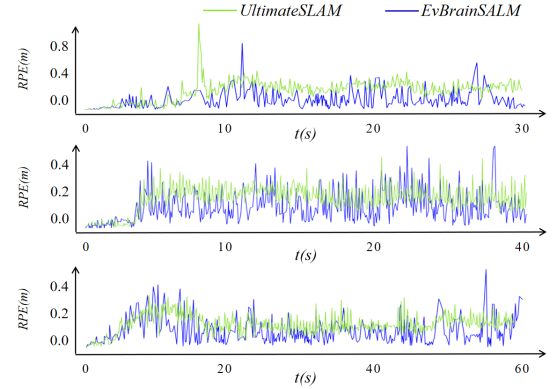


Fig. 11. Comparison of relative pose errors between our method and UltimateSLAM in three different scenarios. The sequence from top to bottom is 01, 08, 09.

TABLE IV

TEST RPE (M) AND APE (M) COMPARISON OF DIFFERENT METHODS ON NINE SEQUENCES.

Sequences	EventEMin		NHEVO		Ours	
	APE	RPE	APE	RPE	APE	RPE
01_bright_low	0.716	0.143	0.843	0.136	0.641	0.108
02_bright_fast	1.063	0.139	2.136	0.141	0.626	0.087
03_dark_low	1.192	0.162	1.864	0.195	0.670	0.093
04_dark_fast	1.381	0.173	2.248	0.204	0.425	0.084
05_dark_very	1.726	0.154	1.873	0.186	1.561	0.103
06_dark&bright	1.139	0.112	1.492	0.164	1.124	0.091
07_long_time	1.286	0.128	1.396	0.158	0.647	0.123
08_corridor1	4.842	0.185	3.437	0.231	2.641	0.127
09_corridor2	3.731	0.167	2.649	0.213	1.635	0.133
Average	1.897	0.151	1.993	0.181	1.108	0.105

2) Comparison with Recent Event-Based Visual Odometry Methods: We further compare EvBrainSLAM with two recent event-based visual odometry methods, NHEVO [41] and EventEMin [42], [43]. NHEVO is a purely event-based

visual odometry method designed for planar ground vehicles. It exploits the non-holonomic motion constraints of Ackermann-steering platforms and combines quasi-continuous event tracks, polynomial motion constraints, and histogram voting to robustly estimate instantaneous motion. EventEMin estimates camera motion by incrementally minimizing the dispersion of motion-compensated events. By processing events individually rather than in batches, it supports low-latency and real-time motion estimation under general camera-motion models.

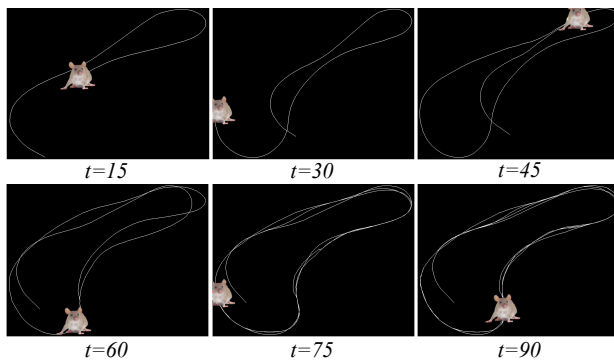


Fig. 12. Progressive construction of a cognitive map by EvBrainSLAM in the indoor environment. The white curves represent the progressively constructed cognitive map, while the robot marker indicates its current location. As exploration proceeds, newly perceived spatial information is accumulated and refined, resulting in an increasingly complete and coherent cognitive map.

As shown in Table III, EvBrainSLAM achieves the lowest APE and RPE on all nine sequences. Its average APE and RPE are 1.108 m and 0.105 m, respectively, corresponding to reductions of 41.6% and 30.4% compared with EventEMin, and 44.4% and 41.7% compared with NHEVO. The improvement is particularly evident on Sequence 04, where EvBrainSLAM reduces the APE to 0.425 m and the RPE to 0.084 m, demonstrating improved robustness when rapid motion and low illumination occur simultaneously. Although all methods exhibit larger errors in the texture-poor corridor sequences, EvBrainSLAM still consistently outperforms both baselines. These results indicate that the proposed event-inertial self-motion cue extraction and CANN-based cue integration improve both local motion consistency and global trajectory accuracy under challenging illumination, rapid-motion, long-duration, and texture-poor conditions.

3) Long-Term Localization Analysis: To further qualitatively evaluate the localization performance, we visualize the estimated robot trajectories in different indoor environments. Figures 12, 13, and 14 present representative trajectories under different navigation scenarios. The results demonstrate that the proposed event-based SLAM framework is capable of producing continuous and smooth trajectories in environments with different spatial structures. Although noticeable deviations can be observed at the beginning of navigation, the estimated trajectories gradually approach the ground truth as the robot explores the environment and obtains more reliable local-view observations.

In the indoor environment shown in Fig. 12, the navigation area is relatively compact and contains limited structural varia-

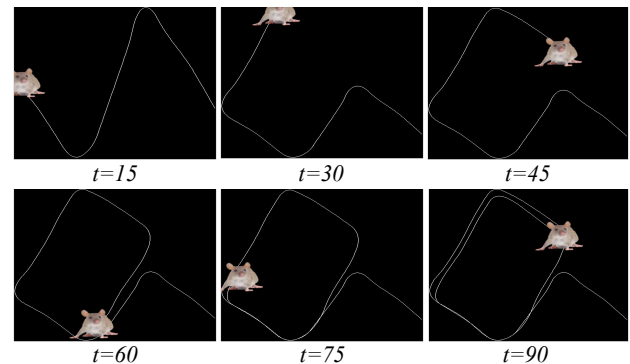


Fig. 13. Progressive construction of a cognitive map by EvBrainSLAM in the Corridor 1 environment. The sequence shows the cognitive map being incrementally updated as the robot explores the corridor.

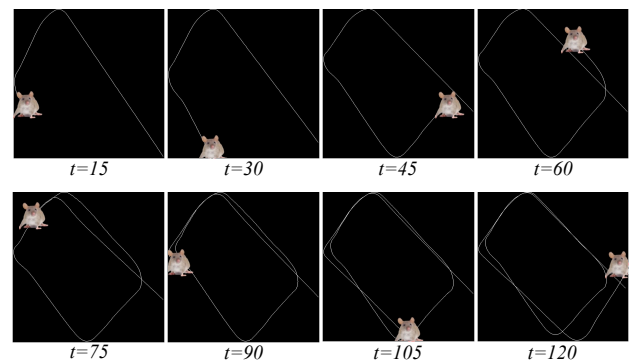


Fig. 14. Progressive construction of a cognitive map by EvBrainSLAM in the Corridor 2 environment. The sequence further demonstrates the progressive cognitive-mapping capability of EvBrainSLAM in a different corridor environment.

tions. At the initial stage of exploration, the robot suffers from relatively large localization errors because only a small number of local-view templates are available for establishing reliable observation associations. Meanwhile, the accumulated motion estimation uncertainty causes the estimated trajectory to deviate from the actual path. However, as the robot continues moving and revisits previously observed regions, the system gradually builds sufficient local-view correspondences. The detected loop closures provide additional global constraints to correct the accumulated drift. After loop-closure optimization, the trajectory estimation becomes significantly more consistent, and the final trajectory maintains good agreement with the reference path. For the Corridor1 and Corridor2 environments shown in Figs. 13 and 14, the navigation spaces are larger and more extended compared with the indoor environment, resulting in greater challenges for long-term localization. Due to the elongated structures and limited repeated observations during the initial exploration stage, the accumulated pose uncertainty gradually increases, leading to noticeable deviations between the estimated and reference trajectories. As the robot continues to traverse the corridors, more local-view templates are collected and reliable associations between current observations and previously visited regions are established. The detected loop closures introduce global constraints into the optimization process, effectively compensating for accumulated translational and rotational drift.

TABLE V
LOCALIZATION ERROR OVER TIME.

	15s	30s	45s	60s	75s	90s
APE	1.91	1.27	0.92	0.68	0.65	0.66
RPE	0.38	0.26	0.19	0.14	0.13	0.13

Table V quantitatively presents the evolution of localization errors during the long-term operation in the indoor environment. At the early stage of navigation (15s), the system suffers from relatively large errors, with the APE and RPE reaching 1.91 m and 0.38 m, respectively. This is mainly attributed to the insufficient local-view experience and the lack of effective loop-closure constraints, which make the accumulated motion estimation uncertainty dominate the localization results. As the operation continues, more environmental observations are collected and the system gradually establishes reliable associations between current observations and previously visited locations. Consequently, both APE and RPE decrease continuously, reaching 0.68 m and 0.14 m at 60s, respectively. This indicates that the learned local-view representations and topological connections can effectively constrain the accumulated drift during long-term navigation. After sufficient exploration and loop-closure optimization at the final stage (90s), the localization errors are further reduced to 0.66 m for APE and 0.13 m for RPE. Although the improvement compared with the previous stage is relatively limited, the final error reduction demonstrates that the introduced loop-closure mechanism maintains global trajectory consistency and prevents further drift accumulation.

E. Ablation Experiments

1) Effect of CANN Parameters on Localization Performance:

As shown in Table VI, the CANN configuration is generally robust to moderate parameter variations, whereas excessively small networks or parameters that deviate substantially from the default setting lead to noticeable performance degradation.

The network size has the most evident influence on localization accuracy. Reducing the network size from $11 \times 11 \times 36$ to $3 \times 3 \times 12$ increases the average APE from 1.108 m to 1.147 m and the average RPE from 0.105 m to 0.131 m. This result indicates that an excessively small pose-cell network cannot provide sufficient spatial and angular resolution. Increasing the network size beyond the $11 \times 11 \times 36$ configuration, however, does not yield a consistent improvement. The $15 \times 15 \times 48$ configuration achieves the lowest RPE of 0.104 m, but its APE slightly increases to 1.110 m.

The cell size has a limited influence within the evaluated range. As the cell size varies from 0.6 to 1.4, the average APE changes only slightly, from 1.117 m to 1.108 m, while the average RPE decreases from 0.116 m to its minimum of 0.105 m at a cell size of 1.0 and then slightly increases. Both 1.0 and 1.4 achieve the lowest APE of 1.108 m, but 1.0 yields a lower RPE.

The excitation and inhibition ranges mainly affect the propagation and stability of the dominant activity packet. Reducing the ranges to 5×5 and 3×3 increases the APE and RPE

to 1.110 m and 0.109 m, respectively, indicating that overly narrow connections limit activity propagation and weaken the suppression of residual competing activities. The default ranges of 7×7 and 5×5 achieve the best overall performance, with an APE of 1.106 m and an RPE of 0.105 m. Further increasing the ranges provides no additional benefit, as the errors gradually increase to 1.109 m and 0.107 m for the 11×11 and 9×9 configuration.

The excitatory and inhibitory variances show similar sensitivity trends. The default settings, $E_{\text{var}} = 1.0$ and $I_{\text{var}} = 2.0$, achieve the best overall performance, with an APE of 1.108 m and an RPE of 0.105 m. Deviating from these values gradually increases both errors, particularly when the variances become large. This suggests that excessively broad excitation spreads the activity packet and reduces pose precision, whereas overly broad inhibition may suppress the dominant activity packet. Conversely, insufficient excitation or inhibition weakens activity propagation or the suppression of competing activities.

The global inhibition coefficient φ also performs best at the default value of 2×10^{-4} . Reducing it to 1×10^{-4} causes only a slight performance decrease. In contrast, stronger global inhibition leads to a more evident degradation. When φ increases to 4×10^{-4} , the APE and RPE rise to 1.121 m and 0.110 m, respectively. This result indicates that overly strong global inhibition weakens the dominant activity packet and reduces the stability of pose estimation.

Overall, the results demonstrate that the CANN is insensitive to moderate changes in the cell size, connection ranges, kernel variances, and global inhibition coefficient. Nevertheless, excessively coarse network resolution, overly broad excitation or inhibition, and strong global inhibition can degrade localization performance. The selected configuration is therefore adopted based on the sensitivity analysis rather than empirical tuning alone, providing a reliable trade-off between pose representation accuracy, activity-packet stability, and computational efficiency.

2) Effect of the Matching Threshold on Localization Performance:

The loop closure matching threshold ε determines whether the current visual template is associated with a previously observed local view. To evaluate its influence, five threshold settings are tested on Sequences 01-09 while keeping all other parameters unchanged. The average APE and RPE are reported to assess global trajectory accuracy and local pose consistency, respectively. As shown in Table VII, a small threshold imposes an overly strict matching criterion. When ε is set to 0.10, valid loop closures affected by viewpoint, illumination, or event-density variations may be rejected, resulting in an average APE of 1.196 m and an average RPE of 0.116 m. Increasing ε to 0.20 allows more valid revisited scenes to be recognized, reducing the average APE and RPE to 1.108 m and 0.105 m, respectively.

However, when the threshold is further increased to 0.25, visually similar but spatially different scenes become more likely to satisfy the matching condition. Such incorrect associations may inject local-view activity into an inconsistent region of the pose-cell network and introduce erroneous corrections into the cognitive map. Consequently, the average APE and RPE increase slightly to 1.113 m and 0.107 m, respectively. When

TABLE VI

SENSITIVITY ANALYSIS OF THE CANN NETWORK CONFIGURATION AND CONNECTION RANGES. THE BEST RESULT IN EACH COLUMN IS SHOWN IN BOLD.

Parameter	Setting	Avg. APE ↓	Avg. RPE ↓
Network size	$3 \times 3 \times 12$	1.147	0.131
	$7 \times 7 \times 24$	1.124	0.114
	$11 \times 11 \times 36$	1.108	0.105
	$15 \times 15 \times 48$	1.110	0.104
	$19 \times 19 \times 60$	<u>1.109</u>	0.106
Cell size	0.6	1.117	0.116
	0.8	1.113	0.111
	1.0	1.108	0.105
	1.2	<u>1.109</u>	0.107
	1.4	1.108	<u>0.106</u>
Excitation and inhibition ranges	$5 \times 5, 3 \times 3$	1.110	0.109
	$7 \times 7, 5 \times 5$	1.106	0.105
	$9 \times 9, 7 \times 7$	<u>1.107</u>	<u>0.106</u>
	$11 \times 11, 9 \times 9$	1.109	0.107
Excitatory variance E_{var}	0.5	<u>1.111</u>	<u>0.106</u>
	1.0	1.108	0.105
	1.5	1.113	0.107
	2.0	1.118	0.109
Inhibitory variance I_{var}	1.0	<u>1.112</u>	<u>0.106</u>
	2.0	1.108	0.105
	3.0	1.116	0.108
	4.0	1.122	0.110
Global inhibition coefficient φ	1×10^{-4}	<u>1.111</u>	<u>0.106</u>
	2×10^{-4}	1.108	0.105
	3×10^{-4}	1.115	0.108
	4×10^{-4}	1.121	0.110

TABLE VII

SENSITIVITY ANALYSIS OF THE SAD THRESHOLD. THE REPORTED APE AND RPE ARE AVERAGED OVER SEQUENCES 01-09.

SAD threshold ε	Avg. APE ↓	Avg. RPE ↓
0.10	1.196	0.116
0.15	1.141	0.112
0.20	1.108	0.105
0.25	1.113	0.107
0.30	×	×

ε is set to 0.30, the matching criterion becomes excessively permissive, substantially increasing the risk of false-positive loop closures. The resulting incorrect associations disrupt the pose-cell dynamics and cognitive-map optimization, causing localization failure and preventing reliable APE and RPE values from being obtained.

Overall, these results validate the choice of $\varepsilon = 0.20$ and demonstrate that the proposed system is robust to moderate threshold variations, whereas an overly permissive threshold may introduce false loop closures and ultimately cause localization failure.

3) Effect of the CANN Module on Localization Robustness:

To evaluate the contribution of the CANN module, we com-

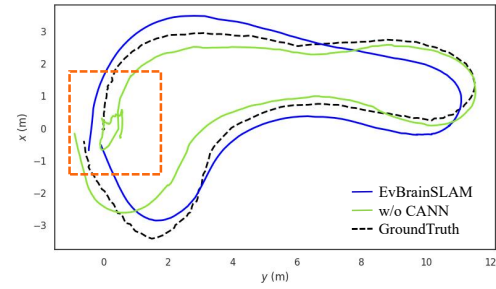


Fig. 15. Trajectory comparison between EvBrainSLAM and its variant without CANN. The orange box highlights the unstable estimates produced by the w/o CANN variant near the starting point.

pare the full EvBrainSLAM with its variant without CANN. Figure 15 compares the trajectories estimated by the two variants. EvBrainSLAM produces a smoother and more consistent trajectory, whereas the w/o CANN variant exhibits unstable estimates near the starting point, as highlighted by the orange box. This result indicates that the CANN module reduces sensitivity to transient disturbances and prevents unstable estimates from propagating through the trajectory.

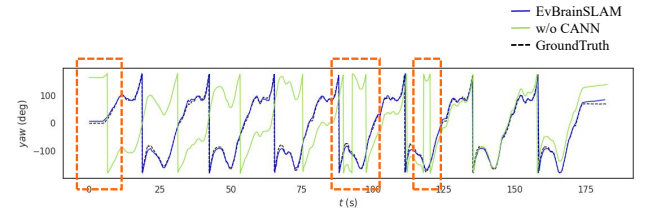


Fig. 16. Yaw estimates obtained by EvBrainSLAM and its variant without CANN, compared with the ground truth. The orange rectangles indicate the large deviations produced by the w/o CANN variant.

Figure 16 further compares the temporal evolution of the yaw estimates with the ground truth. The yaw estimated by EvBrainSLAM remains consistently aligned with the ground truth throughout the sequence. In contrast, the w/o CANN variant exhibits several large deviations. These deviations indicate that removing the CANN module increases sensitivity to transient estimation errors.

VI. CONCLUSION

By leveraging brain-inspired navigation mechanisms and integrating a novel event camera, this work proposes an event-driven brain-inspired SLAM approach. The effectiveness of our method has been demonstrated through a series of experiments, highlighting its adaptability to dynamic environments and robustness under uncertain conditions. The SLAM system retains the complexity in the visual cue acquisition process while simplifying the local view module in a complementary manner. EvBrainSLAM not only pioneers a new paradigm in the SLAM field but also lays the groundwork for multi-sensor sensing studies. In the future, we will commit to investigate the feasibility of combining spiking neural networks [44] with CANN for better biologically-plausible and more efficient positioning and navigation.

REFERENCES

- [1] H.-W. Chae, J.-H. Choi, and J.-B. Song, "Robust and autonomous stereo visual-inertial navigation for non-holonomic mobile robots," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 9613–9623, 2020.
- [2] Y. Wang, D. Gu, X. Ma, J. Wang, and H. Wang, "Robust real-time auv self-localization based on stereo vision-inertia," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 6, pp. 7160–7170, 2023.
- [3] H. Zhao, J. Shang, K. Liu, C. Chen, and F. Gu, "Edgevo: An efficient and accurate edge-based visual odometry," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 10 630–10 636.
- [4] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.
- [5] A. Rosinol, A. Violette, M. Abate, N. Hughes, Y. Chang, J. Shi, A. Gupta, and L. Carlone, "Kimera: From slam to spatial perception with 3d dynamic scene graphs," *The International Journal of Robotics Research*, vol. 40, no. 12-14, pp. 1510–1546, 2021.
- [6] A. Censi and D. Scaramuzza, "Low-latency event-based visual odometry," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 703–710.
- [7] H. Rebecq, T. Horstschaefer, G. Gallego, and D. Scaramuzza, "Evo: A geometric approach to event-based 6-dof parallel tracking and mapping in real time," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 593–600, 2017.
- [8] A. R. Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Ultimate slam? combining events, images, and imu for robust visual slam in hdr and high-speed scenarios," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 994–1001, 2018.
- [9] P. Chen, W. Guan, and P. Lu, "Esvio: Event-based stereo visual inertial odometry," 2023.
- [10] Z. Gong, R. Ying, W. Fei, J. Qian, and P. Liu, "Tightly coupled integration of gnss and vision slam using 10-dof optimization on manifold," *IEEE Sensors Journal*, vol. PP, no. 99, pp. 1–1, 2019.
- [11] D. Ball, S. Heath, J. Wiles, G. Wyeth, P. Corke, and M. Milford, "Openratslam: an open source brain-based slam system," *Autonomous Robots*, vol. 34, no. 3, pp. 149–176, 2013.
- [12] F. Yu, J. Shang, Y. Hu, and M. Milford, "Neuroslam: a brain-inspired slam system for 3d environments," *Biological cybernetics*, vol. 113, no. 5, pp. 515–545, 2019.
- [13] J. O'Keefe and J. Dostrovsky, "The hippocampus as a spatial map: Preliminary evidence from unit activity in the freely-moving rat." *Brain research*, 1971.
- [14] T. Hafting, M. Fyhn, S. Molden, M.-B. Moser, and E. I. Moser, "Microstructure of a spatial map in the entorhinal cortex," *Nature*, vol. 436, no. 7052, pp. 801–806, 2005.
- [15] J. S. Taube, R. U. Muller, and J. B. Ranck, "Head-direction cells recorded from the postsubiculum in freely moving rats. i. description and quantitative analysis," *Journal of Neuroscience*, vol. 10, no. 2, pp. 420–435, 1990.
- [16] E. Kropff, J. E. Carmichael, M.-B. Moser, and E. I. Moser, "Speed cells in the medial entorhinal cortex," *Nature*, vol. 523, no. 7561, pp. 419–424, 2015.
- [17] E. I. Moser, M.-B. Moser, and B. L. McNaughton, "Spatial representation in the hippocampal formation: a history," *Nature neuroscience*, vol. 20, no. 11, pp. 1448–1464, 2017.
- [18] M. J. Milford, G. F. Wyeth, and D. Prasser, "Ratslam: a hippocampal model for simultaneous localization and mapping," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 1. IEEE, 2004, pp. 403–408.
- [19] J. Steckel and H. Peremans, "Batslam: Simultaneous localization and mapping using biomimetic sonar," *PloS one*, vol. 8, no. 1, p. e54076, 2013.
- [20] A. Samsonovich and B. L. McNaughton, "Path integration and cognitive mapping in a continuous attractor neural network model," *Journal of Neuroscience*, vol. 17, no. 15, pp. 5900–5920, 1997.
- [21] G. Zhuang, Z. Bing, Y. Huang, K. Huang, and A. Knoll, "A biologically-inspired simultaneous localization and mapping system based on lidar sensor," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 13 136–13 142.
- [22] Y. Chen, Z. Xiong, and J. Liu, "Brain-inspired multisensor navigation information fusion model based on spatial representation cells," *IEEE Sensors Journal*, vol. 24, no. 11, pp. 18 122–18 132, 2024.
- [23] H. Zhao, F. Yu, X. Hu, Z. Xiong, J. Shang, and F. Gu, "Brain-inspired navigation and sensing for robots," p. 1329324, 2023.
- [24] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.
- [25] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.
- [26] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "Dtam: Dense tracking and mapping in real-time," in *2011 International Conference on Computer Vision*, 2011, pp. 2320–2327.
- [27] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "Slam++: Simultaneous localisation and mapping at the level of objects," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1352–1359.
- [28] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 15–22.
- [29] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, mar 2018.
- [30] B. Kueng, E. Mueggler, G. Gallego, and D. Scaramuzza, "Low-latency visual odometry using event-based feature tracks," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 16–23.
- [31] H. Rebecq, G. Gallego, E. Mueggler, and D. Scaramuzza, "EMVS: Event-based multi-view stereo—3D reconstruction with an event camera in real-time," *Int. J. Comput. Vis.*, vol. 126, pp. 1394–1414, Dec. 2018.
- [32] A. Zihao Zhu, N. Atanasov, and K. Daniilidis, "Event-based visual inertial odometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5391–5399.
- [33] Y. Zuo, J. Yang, J. Chen, X. Wang, Y. Wang, and L. Kneip, "Devo: Depth-event camera visual odometry in challenging conditions," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 2179–2185.
- [34] J. Hidalgo-Carrió, G. Gallego, and D. Scaramuzza, "Event-aided direct sparse odometry," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5781–5790.
- [35] L. Silveira, F. Guth, P. Drews-Jr, P. Ballester, M. Machado, F. Codevilla, N. Duarte-Filho, and S. Botelho, "An open-source bio-inspired solution to underwater slam," *IFAC-PapersOnLine*, vol. 48, no. 2, pp. 212–217, 2015.
- [36] M. J. Milford and G. F. Wyeth, "Mapping a suburb with a single camera using a biologically inspired slam system," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1038–1053, 2008.
- [37] H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization," 2017.
- [38] Bosse, Michael, Furgale, Paul, Leutenegger, Stefan, Siegwart, Roland, Lynen, and Simon, "Keyframe-based visual-inertial odometry using nonlinear optimization," *International Journal of Robotics Research*, 2015.
- [39] M. Mulas, N. Waniek, and J. Conradt, "Hebbian plasticity realigns grid cell activity with external sensory cues in continuous attractor models," *Frontiers in computational neuroscience*, vol. 10, p. 13, 2016.
- [40] M. Grupp, "evo: Python package for the evaluation of odometry and slam." <https://github.com/MichaelGrupp/evo>, 2017.
- [41] W. Xu, S. Zhang, L. Cui, X. Peng, and L. Kneip, "Event-based visual odometry on non-holonomic ground vehicles," in *2024 International Conference on 3D Vision (3DV)*, 2024, pp. 831–841.
- [42] U. Miguel Nunes and Y. Demiris, "Live demonstration: Incremental motion estimation for event-based cameras by dispersion minimisation," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021, pp. 1322–1323.
- [43] U. M. Nunes and Y. Demiris, "Robust event-based vision model estimation by dispersion minimisation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 12, pp. 9561–9573, 2022.
- [44] F. Gu, F. Guo, F. Yu, X. Long, C. Chen, K. Liu, X. Hu, J. Shang, and S. Guo, "Accurate and efficient floor localization with scalable spiking graph neural networks," *Satellite Navigation*, vol. 5, no. 1, p. 6, 2024.