SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

May 2025, Vol. 68, Iss. 5, 150207:1–150207:16
https://doi.org/10.1007/s11432-024-4384-6

Special Topic: Embodied Intelligence

# VF-Nav: visual floor-plan-based point-goal navigation

Wangtian SHEN, Ziyang MENG*, Pengfei GU, Pengkun ZHOU, Fangwen YU & Xu LV

*Department of Precision Instrument, Tsinghua University, Beijing 100084, China*

Received 12 June 2024/Revised 20 November 2024/Accepted 9 March 2025/Published online 24 April 2025

**Abstract** In indoor navigation tasks, the use of floor plans is an efficient and cheap way to provide globally consistent metric and topological information about various environments. However, most studies on floor-plan-based navigation have relied on LiDAR rather than RGB cameras because of the difficulty of performing cross-modality matching. In this paper, we instead focus on the visual indoor navigation problem and propose VF-Nav, a visual floor-plan-based point-goal navigation algorithm combining a brain-inspired localization method with a topological planning technique. In the proposed approach, continuous and accurate localization is achieved by combining the metric information provided by the floor plan with a brain-inspired localization model. Then, the global path to the point goal is generated by building the topological map from the floor plan, and a short-term target is provided at each step. Finally, a reinforcement learning control module guides the robot to reach each short-term target. The experimental results on a simulated point-goal navigation dataset demonstrate the excellent performance of the proposed approach in a complicated indoor environment. Our method achieves a success rate of up to 88% and a success weighted by path length of 71%.

**Keywords** point-goal navigation, brain-inspired localization, floor plan

**Citation** Shen W T, Meng Z Y, Gu P F, et al. VF-Nav: visual floor-plan-based point-goal navigation. Sci China Inf Sci, 2025, 68(5): 150207, https://doi.org/10.1007/s11432-024-4384-6

## 1 Introduction

Autonomous navigation is one of the most crucial tasks for intelligent mobile robots. Navigation tasks can be categorized into point-goal, image-goal, object-goal, and exploration tasks, among others. Traditional navigation methods typically rely on precise occupancy maps generated by LiDAR to implement planning and control strategies, requiring carefully designed control algorithms [1, 2]. Such approaches limit navigation flexibility, whereas vision can provide richer semantic and contextual information. Therefore, this paper focuses on point-goal navigation tasks with a vision-only sensor setting commonly used in many robotics applications (i.e., [3–5]). In particular, the navigation strategy instructs a robot equipped with an RGB camera to reach a given target specified by its coordinates [6].

Recently, learning-based methods have been developed to solve the point-goal navigation task [5, 7, 8], and promising results have been obtained by training networks through reinforcement learning (RL) in simulated environments [3, 4, 9, 10]. However, all relevant previous studies were based on a critical assumption that the robot can achieve perfect localization using absolutely accurate GPS+Compass sensors. This is obviously not feasible for robots in practical indoor applications where the key gap is to address the accurate localization problem. Traditional methods for visual localization, such as simultaneous localization and mapping (SLAM) [11, 12], may suffer from position drift, especially in environments with weak textures and over long-distance trajectories. The introduction of a prior map is an effective way to eliminate drift, while the commonly used point cloud map typically requires large storage and significant labor costs to generate an accurate point cloud map. For indoor applications, a floor plan can serve as a prior map; it is readily available for many architectures and only requires a small amount of storage. Moreover, the topological information provided by floor plans can be used for path planning, facilitating more efficient navigation.

* Corresponding author (email: ziyangmeng@tsinghua.edu.cn)

 

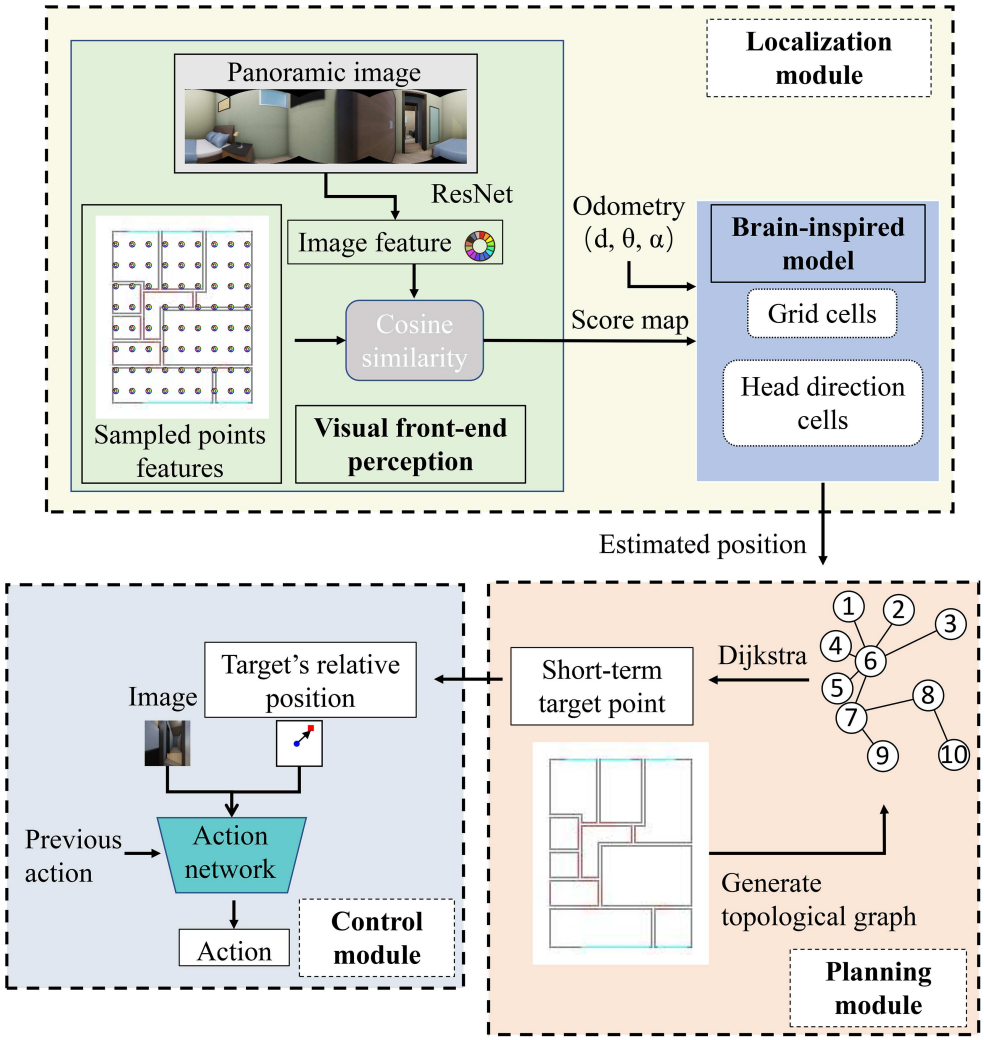


**Figure 1** (Color online) System flow of our navigation framework. Our framework consists of three components: localization module, planning module, and control module.

Most of the current studies on floor-plan-based localization or navigation rely on the use of LiDAR [13–15]. However, LiDAR is costly and lacks robustness against object occlusion or furniture movement. To reduce deployment costs, visual floor plan localization approaches have been proposed recently [16,17], but only a single image retrieval problem has been considered. The direct use of such methods suffers from navigation performance degradation when indoor environments have rooms with similar structures. Motivated by the observation that previous studies have performed less robustly in environments with weak textures or similar structures, we integrate continuous attractor networks with feature extraction and matching methods proposed previously [16] to obtain an accurate and robust localization result. A vision-only learning-based method with a complete localization and navigation pipeline is proposed to realize the point-goal navigation task using floor plan maps as prior information.

More specifically, we adopt the classical architecture of "localization-planning-control" as our framework, as shown in Figure 1. In the localization module, visual descriptors and a brain-inspired neural model are utilized to achieve cross-modal localization with the visual input, odometry input, and floor plan. Simulation results verify that the accuracy of the proposed localization method is significantly superior to those of pure odometry and similarity matching methods. In the planning module, a topological map is created from the floor plan, which is used for global path planning. In the control module, using the generated global path and current image input, we employ a deep RL network to output discrete actions for the robot. A safe route is then generated to reach the local target while avoiding potential obstacle collisions. The main contributions of this study are as follows.

- Based on a floor plan, a localization method that integrates learning-based visual features with

continuous attractor neural networks (CANNs) is proposed for indoor navigation.

• We present a comprehensive floor plan-based indoor navigation framework VF-Nav that integrates localization, planning, and execution.

• Through experiments and ablation studies, we demonstrate that VF-Nav surpasses existing indoor navigation strategies and exhibits robustness against errors introduced by odometry and floor plans.

## 2 Related work

### 2.1 Point-goal navigation

In a point-goal navigation task, a robot is given the coordinates of a target point with respect to the starting position, and the objective is to navigate to the target point using its sensory input [6]. Traditional approaches divide this problem into the geometric mapping of environments [11, 12] and path planning [18,19] to the target point. However, recent studies have indicated that learning-based solutions might be more robust. For example, neural networks have been introduced to construct a cognitive map from RGB-D images and plan paths to a target point [5, 7, 8]. Along with the rapid development of simulation platforms [3, 9, 20] and RL algorithms such as proximal policy optimization (PPO) [21] and decentralized distributed PPO (DD-PPO) [4], end-to-end networks extensively trained by RL in simulated environments have shown strong performance. In this respect, a previous study [3] presented Habitat, a simulation platform for training and evaluating virtual robots, which employed PPO to train robots for learning point-goal navigation capabilities. Furthermore, a larger dataset [9] and a training approach utilizing additional computational resources [4] enabled robots to achieve nearly perfect performance in point-goal navigation tasks. Nevertheless, all these studies operated under the crucial assumption that robots can attain excellent localization using impeccably accurate GPS+Compass sensors. In practical applications, especially for indoor robots, this is obviously unfeasible.

Current studies on realistic point-goal navigation problems, such as the case without global positioning equipment, mainly rely on a convolutional neural network visual odometry at every step to obtain the current localization result [10, 22, 23]. However, the main issue with such methods is that the localization accuracy inevitably drifts and results in the failure of navigation tasks in long-distance applications. Moreover, the simulations conducted in the current literature primarily considered cases where the starting and target points were distributed in nearby rooms, reducing their applicability in real environments. We also demonstrate that, when navigating through multiple rooms, end-to-end RL networks achieve suboptimal performance. Instead of end-to-end approaches, we propose a modular navigation method where each module handles a specific subtask. This reduces the performance requirements and training complexity for each network component. In fact, modular methods have been shown to outperform end-to-end approaches in other navigation tasks [24, 25], and we follow such a framework in this paper.

### 2.2 Floor-plan-based localization

As mentioned above, it is unfeasible for indoor robots to be equipped with absolutely precise GPS+Compass in reality. Traditional solutions for estimating robot position are SLAM algorithms, such as ORB-SLAM2 [11] and LSD-SLAM [12], which rely on feature point extraction and may suffer from sparse texture scenes and long-term pose drift. Therefore, a natural idea is to introduce a map to enhance localization accuracy, and classical solutions in this respect include iterative closest point (ICP) [26] and normal distributions transform [27]. However, these methods are computationally costly and heavily rely on hard-to-generate high-quality point clouds. Meanwhile, leveraging readily available architectural floor plans, which primarily represent the fixed elements of buildings such as walls, windows, and doors, can serve as a cheap and effective way to improve localization accuracy and avoid accumulated drift. Previous studies [13–15] primarily utilized LiDAR to perceive the surrounding environment and achieve localization within a floor plan using particle filters [28] or ICP [26]. Considering that LiDAR is expensive and such geometric-measurement-based methods have limited robustness when facing object occlusions, the use of RGB images can lower the deployment cost and provide additional semantic information to aid navigation tasks. The floor-plan-based localization problem has previously been considered using RGB images as input [16, 17, 29]. These studies used a learning-based approach to separately calculate feature descriptors for the image and the points in the map and obtain the localization result by comparing their similarity. For instance, previous studies [16, 17] sampled points in a map and rendered their feature

descriptors using a learned codebook or a network, whereas another study [29] directly generated a latent floor plan from an original floor plan using a U-Net [30]. These methods demonstrated their capability to learn a common feature space for two distinct modalities (i.e., vision and floor plan). We extend the above studies from a single image retrieval problem to the case of continuous trajectory estimation by combining the descriptors obtained from a brain-inspired localization model.

### 2.3 Brain-inspired SLAM

In contrast to geometric technologies, numerous animals exhibit the ability to robustly map and navigate within unfamiliar environments by relying on their brain activity. The navigation capability of mammals' brains is thought to depend on the hippocampus, which has been shown to achieve spatial localization based on the integration of self-motion signals and calibration from visual signals [31]. The hippocampus contains several kinds of spatially encoding cells that have strong spatial characteristics when rodents are moving. Some specific place cells are activated when rodents reach corresponding locations in their environment and fire to a lesser degree as they move away [32]. Head direction cells are activated when a rodent's head is at specific global orientations, and their activity is not influenced by the rodent's position [33, 34]. With the discovery of the spatial neural mechanism in the brain, some brain-inspired methods based on place and head direction cells have been developed to achieve localization in two-dimensional (2D) or three-dimensional (3D) environments. For instance, previous studies [35–37] developed RatSLAM, a rodent brain-inspired SLAM algorithm, which successfully performed localization, mapping, and navigation in suburb and office environments. Using 3D spatial representation in the brains of bats, rats, and humans, other studies [38, 39] expanded the RatSLAM to 3D using a 3D place cell model. Among these brain-inspired studies, CANNs play a crucial role in modeling the behavior of place and head direction cells. CANN is a type of neural network model with fixed weights between neural units and can converge to specific states in the absence of external input. Although existing brain-inspired localization models have demonstrated good performance, they primarily rely on loop closure detection, which is only available when robots encounter previously visited locations. In this work, we introduce a methodology that integrates floor plans to enable the continuous calibration of localization at each step rather than limiting it to occurrences of loop closure.

## 3 Problem formulation and system overview

In the point-goal navigation task, the target position relative to the initial point is supposed to be known and the action space of the robot is considered to be discrete—comprising stop, move forward, turn left, and turn right. In our setting, noise is added to the robot's actuation. The 2D architectural floor plan denotes walls, doors, and windows of a floor of a building, whereas unstructured elements like tables and chairs are not considered. The robot is equipped with a panoramic camera and noisy odometry. In real-world scenarios, the odometry sensor can be provided by a wheel encoder, IMU, or other equipment. In our simulation environment, we simulate odometry by adding noise to the true relative pose between two steps.

The system overview is shown in Figure 1. The system comprises a localization module, a planning module, and a control module. In particular, the localization module comprises a visual front-end perception module and a brain-inspired model. In the visual front-end perception, feature descriptors for sampled points in the floor plan are prepared before the task. Simultaneously, the feature descriptor of the observed panoramic image is computed at each step and used to calculate a score map to indicate the possibility of each location. Then, the score map, along with the odometry information, is processed by the brain-inspired model to update the robot's position. The planning module utilizes the topological graph abstracted from the floor plan to plan a path to the target point and determine the short-term target on the basis of the estimated position. Finally, the control module outputs the appropriate action according to the current observation and relative position to the short-term target.

# 4 Methods

## 4.1 Visual front-end perception

In this part, we uniformly sample points from the floor plan and employ two networks to learn the feature descriptors of the observed panoramic image and sampled points in the floor plan. The similarities between these learned feature descriptors are used to compute a score map representing the current perception of the environment for subsequent localization.

**Circular feature.** We represent the features of a panoramic image or a point sampled in a map using circular feature descriptors suitable for floor-plan-based localization, as proposed by LASER [16]. Each feature descriptor is in the form of a circular vector:

$$F = \{f^k | k = 0, \ldots, V-1\}, \tag{1}$$

where $V$ is the number of feature segments and $f^k \in \mathbb{R}^D$ is a $D$-dimensional vector that represents the field of view within the range of $[\frac{2\pi k}{V}, \frac{2\pi(k+1)}{V})$. The reason for designing such circular feature vectors is that the leftmost and rightmost regions of the panoramic image correspond to the same orientation angle.

**Similarity measurement.** The similarity measurement between two circular features $F_i = \{f_i^k | k = 0, \ldots, V-1\}$ and $F_j = \{f_j^k | k = 0, \ldots, V-1\}$ is defined as

$$S_{\text{similar}}(F_i, F_j) = \frac{\sum_{k=0}^{V-1} \cos(f_i^k, f_j^k)}{2V} + 0.5, \tag{2}$$

where $\cos(\cdot, \cdot)$ is the computation of the vector cosine similarity. The value 0.5 is used to make the similarity measurement value belong to $[0, 1]$. Because the comparison between a panoramic image and a sampled point is related to not only the position but also the direction, a rotation function $R(F, \theta)$ is defined as below to rotate a circular descriptor with a given angle $\theta$, where

$$R(F, \theta) = \{f^{(k+V\frac{\theta}{2\pi})\%V} | k = 0, \ldots, V-1\}. \tag{3}$$

**Map branch.** To achieve accurate robot localization within the floor plan, we sample a grid of points across the floor plan, where each sampled point is associated with a circular feature descriptor. The circular descriptors are generated on the basis of the features along the boundary of the surrounding walls. The readers can refer to a previous work [16] for more details.

**Image branch.** We utilize a ResNet50 encoder [40] to generate a feature map from the panoramic image. Subsequently, the feature map is vertically squeezed to obtain circular feature descriptors and then undergoes another round of average pooling in the horizontal direction to generate a consistent number of feature segments.

**Training.** We use triplet loss [41] to train the networks in the map branch and image branch. A triplet consists of three data points: an anchor, a positive example, and a negative example:

$$L_{\text{triplet}} = 2 \cdot \max(S_{\text{similar}}(F_I, F^-) - S_{\text{similar}}(F_I, F^+) + 0.5, 0), \tag{4}$$

where the anchor is a circular feature descriptor $F_I$ of a panoramic image, the positive example is the map circular feature descriptor at ground-truth pose $F^+ = R(F_{t_{\text{gt}}}, \theta_{\text{gt}})$, and the negative example is the map circular feature descriptor at a random pose $F^- = R(F_{t_{\text{rand}}}, \theta_{\text{rand}})$.

**Score map.** By calculating the descriptors of the current observed image and every sampled map point, we can obtain a score map covering the entire map area and all directions using

$$S = \{s_{i,j,k} | i = 0, \ldots, X-1, j = 0, \ldots, Y-1, k = 0, \ldots, V-1\}, \tag{5}$$

$$s_{i,j,k} = S_{\text{similar}}\left(F_I, R\left(F_{m_{i,j}}, 2\pi\frac{k}{V}\right)\right), \tag{6}$$

where $(X, Y)$ is the size of the grid of sampled points in the floor plan map and $F_{m_{i,j}}$ is the descriptor of the sampled point at coordinate $(i, j)$.

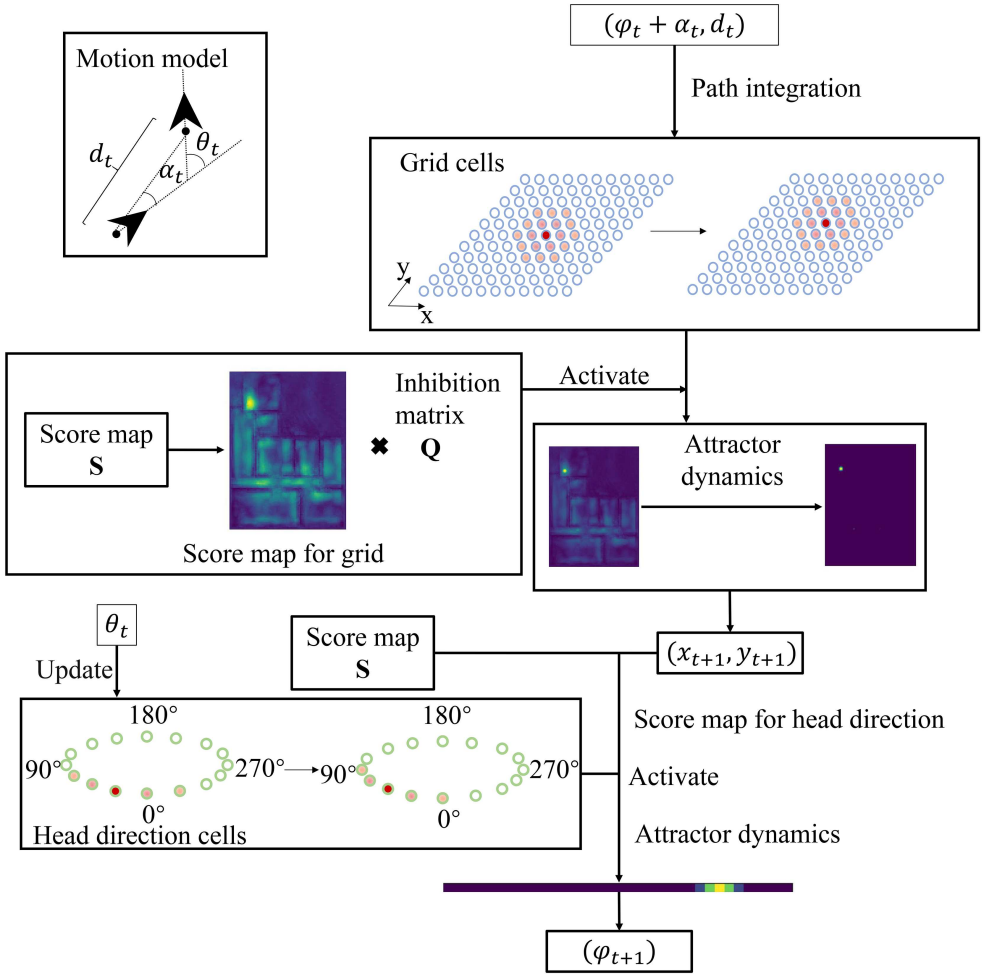


**Figure 2** (Color online) Architecture of the proposed brain-inspired model. The system consists of grid cells and head direction cells. They input odometry $(d_t, \theta_t, \alpha_t)$ and the visual front-end score map $S$ to obtain the current estimated pose $(x_{t+1}, y_{t+1}, \varphi_{t+1})$.

## 4.2 Brain-inspired model

The proposed brain-inspired model consists of a grid cell network and a head direction network, whose architecture is shown in Figure 2. We follow Yu et al. [39] in their use of a 2D-CANN and a one-dimensional (1D)-CANN to model the grid cells and head direction cells, respectively. The robot's pose $(x_t, y_t, \varphi_t)$ in the 2D environment can be represented by the activity of the cells in the two networks. At each step, the noisy odometry information is given by adding Gaussian noise to the ground truth, and their detailed definitions are given in Subsection 4.2.1. The brain-inspired model inputs the score map $S$ obtained from the visual front-end feature descriptors and odometry information, updating their activity state and obtaining the current position of the robot.

### 4.2.1 *Motion model*

To estimate the movement of the robot at every step with the cell networks, the motion update model is defined as follows, and the illustration can be found at the top-left corner of Figure 2:

$$(x_{t+1}, y_{t+1}, \varphi_{t+1}) = (x_t + d_t\cos(\varphi_t + \alpha_t), y_t + d_t\sin(\varphi_t + \alpha_t), \varphi_t + \theta_t), \tag{7}$$

where $d_t$ is the distance moved by the robot, $\alpha_t$ denotes the direction of movement, and $\theta_t$ represents the change in orientation.

#### 4.2.2 *2D grid cell network*

The grid cell model is a 2D CANN, which mimics mammalian brain spatial representation, as shown in Figure 2. Cells are distributed in a grid pattern and interconnected with each other. Each cell represents a particular position $(x, y)$ in the 2D environment and has an activity value $P^{\mathrm{gc}}_{x,y}$. At each step, the network performs path integration, activation, and attractor dynamics sequentially.

**Path integration.** Path integration shifts the activity of grid cells in the 2D environment on the basis of the previous estimated orientation $\varphi_t$ from head direction cells, the direction of movement $\alpha$, and the movement distance $d$. The activity of a cell located at $(x, y)$ after path integration is calculated as follows:

$$P^{\mathrm{gc}'}_{x,y} = \sum_{i=\delta x_0}^{\delta x_0+1} \sum_{j=\delta y_0}^{\delta y_0+1} \gamma_{i,j} P^{\mathrm{gc}}_{x+i,y+j}, \tag{8}$$

where $\delta x_0, \delta y_0$ represent the integer part of the translation coordinates and $\gamma_{i,j}$ is the residual coefficient. In addition, the decimal parts are defined as $\delta x_f, \delta y_f$:

$$\begin{bmatrix} \delta x_0 \\ \delta y_0 \end{bmatrix} = \begin{bmatrix} \lfloor d\cos(\varphi_t + \alpha) \rfloor \\ \lfloor d\sin(\varphi_t + \alpha) \rfloor \end{bmatrix}, \tag{9}$$

$$\begin{bmatrix} \delta x_f \\ \delta y_f \end{bmatrix} = \begin{bmatrix} d\cos(\varphi_t + \alpha) - \delta x_0 \\ d\sin(\varphi_t + \alpha) - \delta y_0 \end{bmatrix}. \tag{10}$$

Then the residual coefficient $\gamma_{i,j}$ in (8) is defined as

$$\begin{aligned} &\gamma_{i,j} = f(\delta x_f, i - \delta x_0) f(\delta y_f, j - \delta y_0), \\ &f(a, b) = \begin{cases} a, & \text{if } b = 1, \\ 1 - a, & \text{if } b = 0. \end{cases} \end{aligned} \tag{11}$$

**Activation.** Previous studies [35,36,39] maintained local view libraries and energizing cells upon recognizing familiar views. However, despite their effectiveness for loop closure, they exhibited accumulated drift in the absence of familiar images. In the proposed algorithm, we employ a front-end score map to activate cells at each step, enabling continuous calibration and limited localization error. As the size of the score map is $X \times Y \times V$ (as mentioned in Subsection 4.1), we calculate the score map for the grid cells by taking the maximum value of the scores in different directions at each step:

$$\left\{ s_{i,j} = \max_k s_{i,j,k} | i = 0, \ldots, X-1, j = 0, \ldots, Y-1 \right\}. \tag{12}$$

The score map for the grid cells represents the position estimation based on the current visual observation. We inject the activation into the entire 2D-CANN based on this estimation, correcting errors introduced by noisy odometry in path integration. To reduce the influence of locations with similar structures, we design an inhibition matrix $\boldsymbol{Q}$ as

$$\begin{aligned} &\boldsymbol{Q}_{i,j} = 1/\sigma \left( \sigma - \mathrm{e}^{(\eta \sqrt{(i-i_0)^2 + (j-j_0)^2})} \right), \\ &i_0, j_0 = \arg\max_{i,j} P^{\mathrm{gc}}_{i,j}, \end{aligned} \tag{13}$$

where $\sigma$ and $\eta$ are constants controlling the activation strength from the score map. Then, the activity of a grid cell is updated by

$$P^{\mathrm{gc}'}_{x,y} = P^{\mathrm{gc}}_{x,y} + Q_{i,j} s_{i,j}. \tag{14}$$

**Attractor dynamics.** The attractor dynamics of a 2D CANN consists of three stages: local excitation, local and global inhibition, and normalization. During the process of attractor dynamics, each cell excites the cells around it and injects activity into them according to the weights $\epsilon^{\mathrm{gc}}_{i,j}$, which follow a 2D Gaussian distribution. Meanwhile, each cell is inhibited by nearby cells with the 2D Gaussian inhibition weights $\Phi^{\mathrm{gc}}_{i,j}$ and a global inhibition $\phi^{\mathrm{gc}}$. Then, each cell's activity is updated by

$$P^{\mathrm{gc}'}_{x,y} = \left( \sum_{m=0}^{X-1} \sum_{n=0}^{Y-1} \epsilon^{\mathrm{gc}}_{m-x,n-y} P^{\mathrm{gc}}_{m,n} - \Phi^{\mathrm{gc}}_{m-x,n-y} P^{\mathrm{gc}}_{m,n} \right) - \phi^{\mathrm{gc}}. \tag{15}$$

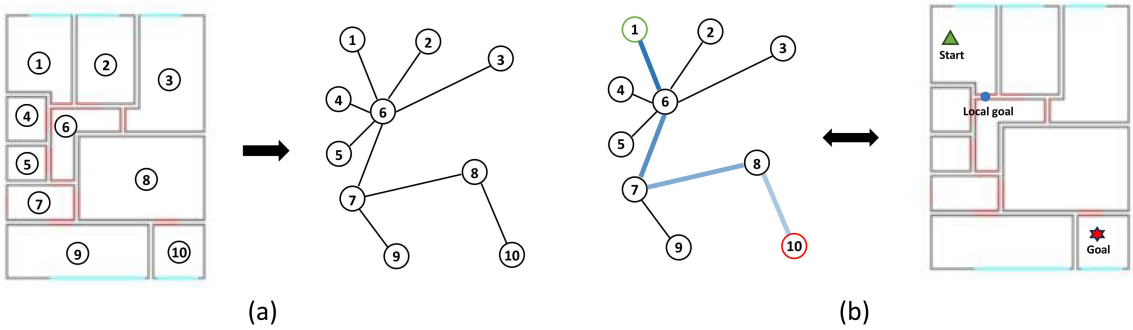


**Figure 3** (Color online) Spatial topology information abstracted from the floor plan. (a) Topological graph built from the floor plan by treating each room and each door as a topological node and edge, respectively; (b) planning based on the built topological graph.

Finally, normalization is performed according to

$$\begin{aligned} P_{x,y}^{\mathrm{gc}'} &= \max(P_{x,y}^{\mathrm{gc}'}, 0), \\ P_{x,y}^{\mathrm{gc}} &= \frac{P_{x,y}^{\mathrm{gc}'}}{\sum_x \sum_y P_{x,y}^{\mathrm{gc}'}}. \end{aligned} \tag{16}$$

#### 4.2.3 *1D head direction cell network*

The head direction cell network is a circular 1D-CANN that mimics the direction cognition of mammals, as shown in Figure 2. A cell represents a particular direction in the 2D space and has an activity value $P_{\varphi}^{\mathrm{hdc}}$. Because of its circular structure, we have $P_{\varphi+kV}^{\mathrm{hdc}} = P_{\varphi}^{\mathrm{hdc}}$. The steps of the head direction cell model are similar to those of grid cells.

**Head direction update.** In this stage, the activity of head direction cells is shifted by the turning angle $\theta$. The detailed computational process can be considered a 1D version of the path integration mentioned in the 2D grid cell part.

**Activation.** As with the activation of grid cells, head direction cells are activated by the score map for head direction, which is achieved according to the score map $S$ and the position estimated by grid cells $(x_{\mathrm{est}}, y_{\mathrm{est}})$. We obtain the score map for the head direction by taking the maximum similarity in each direction within the vicinity of the position obtained from the grid cells:

$$\left\{ s_k = \max_{|i-x_{\mathrm{est}}|<\mathrm{th}, |j-y_{\mathrm{est}}|<\mathrm{th}} s_{i,j,k} | k = 0, \ldots, V-1 \right\}, \tag{17}$$

where th represents the size of the sampling range in the vicinity. The score map is extracted only in the vicinity of the current estimated position to avoid the influence of the scores in other locations. The score map contains the current perception of the robot's direction and is used to calibrate the error from the odometry. Then, the score map is added to $P^{\mathrm{hdc}}$ to realize activation.

**Attractor dynamics.** The attractor dynamics of the head direction cells is the 1D version of that in grid cells (see (15) and (16)), which also consist of three stages: local excitation, local and global inhibition, and normalization.

### 4.3 Planning module

Despite the success of end-to-end networks in point-goal navigation, they mainly consider cases where the starting and target points are in nearby rooms, limiting their applicability in the real world. When facing navigation tasks that involve traversing multiple rooms, the performance of such end-to-end methods tends to degrade as they fail to leverage spatial information for more efficient navigation. In this respect, we propose a planning method that extracts topological information from floor plans. We represent each room as a topological node, whereas the doors (assumed to be passable) serve as edges connecting the rooms (as illustrated in Figure 3(a)). We extract topological nodes and edges from the map through connected component detection. The topological graph can be defined as $G = (V, E)$. In particular, $V = \{v_i\}$ is the set of vertices/nodes, where $v_i$ represents the entire spatial coverage of a room and

**Algorithm 1** Short-term goal planning.

$v_x = l2t(x), v_f = l2t(g_f)$;
**if** $v_x = v_f$ **then**
  $g_s = g_f$;
**else**
  $\{p_i\}_{i=1:n} = \text{Dijkstra}(v_x, v_f)$;
  $g_s = p_1$;
**end if**
**while** $|x - g_f| > 0.3$ m **do**
  $v_x = l2t(x), v_f = l2t(g_f)$;
  **if** $v_x = v_f$ **then**
    $g_s = g_f$;
    continue;
  **end if**
  **if** $|x - g_s| < 0.3$ m **then**
    $\{p_i\}_{i=1:n} = \text{Dijkstra}(v_x, v_f)$;
    $g_{s'} = p_1$;
    **if** $|g_{s'} - x| < 0.3$ m **then**
      **if** $n < 2$ **then**
        $g_s = g_f$;
      **else**
        $g_s = p_2$;
      **end if**
    **else**
      $g_s = g_{s'}$;
    **end if**
  **end if**
**end while**

$E = \{(e_j, r_j, s_j)\}$ is the set of edge tuples, with each edge $e_j$ representing the door connecting the nodes $r_j$ and $s_j$. We utilize component detection to extract the set of rooms $V = \{v_i\}$ and generate the connecting edges.

Using the estimated position obtained from the localization module, we first determine the topological nodes to which the current estimated location $x$ and the final goal location $g_f$ belong using the topological graph, where $x$ and $g_f$ represent the 2D coordinates of the corresponding points. Then, the current and the goal topological nodes can be obtained by $v_x = l2t(x)$ and $v_f = l2t(g_f)$, respectively. Subsequently, we employ the Dijkstra algorithm to plan the shortest path $\{p_i\}_{i=1:n}$ from the current node to the target node. The position of the door associated with the first edge along the chosen path is selected as the short-term goal $g_s$ and passed to the control module, as shown in Figure 3(b). When the estimated position is within a distance of less than 0.3 m from the short-term target point, the robot is considered to have reached the short-term target point, and the next target point is planned. The detailed planning procedure in navigation is given in Algorithm 1.

Instead of regarding the floor plan as an occupancy map for path planning, we use doors as edges to connect the room nodes and construct a topological graph. This choice addresses the challenge of incomplete spatial knowledge from the floor plan alone. Leveraging known door passability improves navigation efficiency, enabling smoother movements between rooms. This approach ensures safer paths and avoids potential collisions.

### 4.4 Control module

We utilize the model proposed by Wijmans et al. [4] as the control module. This model comprises a ResNet50 [40] encoder and a two-layer long short-term memory (LSTM) [42]. Training is conducted using DD-PPO [4] within the Habitat platform [3]. We choose this method as our control strategy because it is trained on the simulation platform with 2.5 billion steps of RL, equipping it with generalized navigation skills. It has been shown to perform well across diverse environments [4,10]. The control module receives three inputs: the previous action (stop, move forward, turn left, and turn right), the goal coordinates relative to the current position (a vector of length 2), the RGB image captured by the forward-facing camera (a vector with a shape of $3 \times 256 \times 256$), and the state vector (of length 512) output by the LSTM at the previous time step. Then, the control module outputs the action decision (stop, move forward, turn left, and turn right) and the state vector of the LSTM at the current time step. The reward for the control module is calculated using the reduction in the distance to the goal: $r_t = d_{t-1} - d_t - 0.01$. When the robot reaches the goal, an additional reward $r_{\text{reach}} = 2.5$ is obtained. The computation flow and the reward function of the control module are shown in Figure 4. Our control module is trained specifically

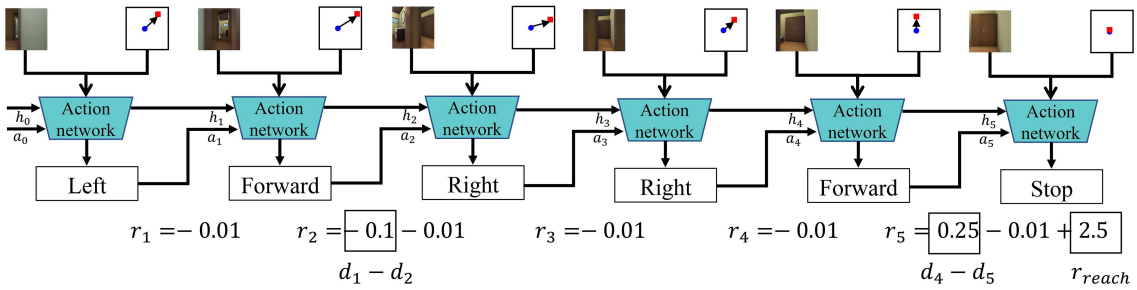


**Figure 4** (Color online) Computation flow and reward function of the control module.

for idealized point-goal navigation tasks, assuming that the robot is equipped with GPS+Compass. Notably, according to Partsey et al. [10], action models trained for idealized point-goal navigation tasks demonstrate good transferability across different environments. Therefore, our control module is only trained within the Habitat platform, and no fine-tuning is implemented.

## 5 Experimental evaluation

### 5.1 Simulation environment and metrics

In our experiment, we incorporate a total of 248 point-goal navigation episodes within four simulated indoor environments constructed using the Unity engine. These simulated environments primarily consist of meticulously designed indoor house scenes [43], as shown in Figure 5. Each navigation task features distinct starting and target points located in different rooms, requiring traversal across multiple rooms to reach the objective.

To evaluate the navigation method, we select four primary metrics.

(1) Success rate, where a navigation attempt is considered successful if the robot reaches the target point within a 0.3-m range and a failure if it exceeds 500 steps without reaching the target.

(2) Success weighted by the path length (SPL) [6], which is defined as

$$\mathrm{SPL} = \frac{1}{N}\sum_{i=1}^{N} S_i \cdot \frac{l_i}{\max(z_i, l_i)}, \tag{18}$$

where $S_i = 1$ if the navigation task is successful; otherwise, $S_i = 0$. $l_i$ represents the total length of the robot's path, and $z_i$ is the length of the theoretically shortest path from the starting point to the target point (the shortest path is provided by the NavMeshAgent component in the Unity engine).

(3) SoftSPL [22] is SPL with binary success being replaced by progress toward the goal:

$$\mathrm{SoftSPL} = \frac{1}{N}\sum_{i=1}^{N}\left(1 - \frac{d_{T_i}}{d_{0_i}}\right) \cdot \frac{l_i}{\max(z_i, l_i)}, \tag{19}$$

where $d_{T_i}$ is the distance to the target point at the end of the episode (on both successes and failures) and $d_{0_i}$ is the initial distance to the target point.

(4) $d_G$, which represents the distance between the robot and the target point at the end of the episode.

### 5.2 Experimental details

In the localization module, we set the hyperparameters of networks as $G = H = 32$, $V = 16$, and $D = 128$. We sample the map with a 0.1-m interval and establish the grid cells network with a 0.1 m $\times$ 0.1 m uniform grid. We choose a sampling interval of 0.1 m to balance the computation speed and localization accuracy. To validate the performance of our method in real-world scenarios, Gaussian noise is added with a standard deviation of (0.02 m, 2°, 2°) to the odometry $(d, \theta, \alpha)$. Regarding the robot's actuation, Gaussian noise is added to the forward distance and rotation angle for each step, with standard deviations of 0.02 m and 2°, respectively. For the parameters of the brain-inspired model, we set the parameters of $\boldsymbol{Q}$ in (13) as $\sigma = 10$ and $\eta = 80$. In (15), $\epsilon^{\mathrm{gc}}$ and $\Phi^{\mathrm{gc}}$ meet the Gaussian distribution with a standard deviation of 1.5 and 2, respectively. We set $\phi^{\mathrm{gc}} = 0.015$. Unlike conventional neural networks, the

brain-inspired model we proposed is based on CANN, which does not require training, and we do not need to change the parameters for different scenarios. The number of visual front-end perception model parameters is 31.4 million, whereas the number of control model parameters is 26.8 million.

### 5.3 Navigation performance

We evaluate VF-Nav against several existing methods for indoor point-goal navigation tasks.

**DD-PPO(GT) [4].** This is a policy trained with large-scale RL in the simulation platform. Notably, this method requires robots to obtain the current ground-truth position.

**ANS [24].** This is a hierarchical, modular policy proposed for exploration tasks and can be easily adapted for point-goal tasks. It estimates egocentric occupancy and aggregates maps into a global top-down map, which is then used for planning. We use the point-goal version of this method for evaluation and collect data from our environment for training.

**IMN [10].** This method proposes a learning-based visual odometry for point-goal tasks and develops data augmentation techniques to train neural models. Visual odometry takes RGB-D images and action information as input and outputs the relative translation and rotation between two frames.

**NeuroSLAM [39].** This method utilizes the CANN models of 3D grid cells and multilayered head direction cells to realize localization in 3D environments. NeuroSLAM maintains a sequence of visual templates and achieves loop closure by activating cells corresponding to familiar visual templates. We deploy the 2D variant of NeuroSLAM in our test environment.

**A*.** Many navigation methods based on occupancy maps use A* [44] or fast-marching [45] methods for path planning within a map. However, the floor plan we use only includes basic architectural information (doors, windows, and walls) and is not a precise occupancy map. It lacks information on obstacles within rooms, like tables, chairs, and sofas. Nevertheless, we experiment with such methods by treating walls as occupied areas and doors as passable regions using the A* algorithm to plan a path from the current position to the goal point and picking an intermediate goal (within 1.5 m) to navigate to.

**D* Lite.** Similar to the A* algorithm, by treating walls as occupied areas and doors as passable regions, we use D* Lite [46] for path planning and use the following strategy: when the robot finds its forward path blocked, it marks the area ahead as occupied grids on the map and replans the route.

Notably, the methods mentioned above did not utilize topological planning, and some of them had access to the ground-truth position and depth camera. Therefore, we also report the performance of VF-Nav without the topological planning module. The navigation performance of the different methods is presented in Table 1. ANS heavily relied on the accuracy of occupancy map estimation, and odometry noise significantly degraded its map accuracy. Meanwhile, although IMN employed data augmentation techniques for training, the results indicate that such an odometry method inevitably produces drift. NeuroSLAM's use of visual template matching for loop closure was too naive and struggled to provide adequate support for navigation. The reason for the poor performance of the A* method was the lack of occupancy information for all obstacles in the floor plan. This sometimes resulted in short-term goal points being planned in locations occupied by obstacles, leading to a decline in navigation performance. Although D* Lite used a more advanced path planning algorithm, its performance remained poor because of its inability to obtain accurate occupancy information. Meanwhile, VF-Nav(wo topo) achieved performance nearly identical to that of DD-PPO(GT) and significantly outperformed other approaches. This demonstrates that the proposed localization module exhibits highly robust performance in navigation tasks.

Furthermore, Table 1 shows that VF-Nav, with topological planning, achieves the best navigation performance in terms of all metrics, especially SPL and SoftSPL. This indicates that incorporating spatial topological information from the floor plan significantly benefits navigation efficiency. Figure 6 illustrates examples of the generated trajectories. Without the topological planning module, the robot tends to wander back and forward, resulting in decreased navigation efficiency. Through extensive RL training, the network acquires a certain level of intelligence and the ability to navigate and autonomously explore. Although end-to-end networks utilize the hidden state of LSTMs to store information about a robot's history exploration of the environment, such memory is limited. Our results indicate that the intelligence of the end-to-end network (DD-PPO) is insufficient for navigation tasks that involve traversing multiple rooms and passing through multiple doors, where the robot repeatedly explores the same area and exhibits poor navigation performance. Meanwhile, in the proposed method, the utilization

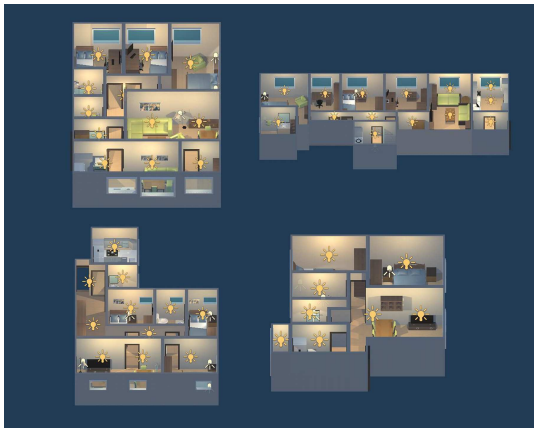

**Figure 5** (Color online) Simulated environments used to test the proposed navigation method.

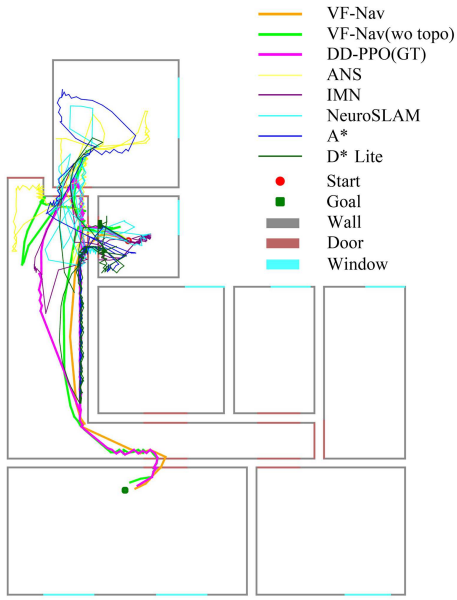


**Figure 6** (Color online) Qualitative results in the indoor environment. Only VF-Nav, VF-Nav(wo topo), and DD-PPO(GT) can reach the goal, with VF-Nav being the most efficient way.

**Table 1** Navigation performance results. The best and second-best results are in bold and underlined, respectively.

| Method | Success rate (%) | SPL (%) | SoftSPL (%) | $d_G$ (m) |
|---|---|---|---|---|
| DD-PPO(GT) | 78.23 | <u>56.39</u> | <u>52.88</u> | 1.42 |
| ANS | 9.27 | 2.80 | 0.70 | 5.34 |
| IMN | 36.29 | 26.09 | 35.51 | 2.30 |
| NeuroSLAM | 38.31 | 24.54 | 25.65 | 2.79 |
| A* | 63.71 | 39.67 | 38.79 | 1.67 |
| D* Lite | 66.13 | 47.68 | 50.93 | 1.43 |
| VF-Nav(wo topo) | <u>79.84</u> | 54.32 | 51.83 | <u>1.26</u> |
| VF-Nav | **88.71** | **71.54** | **68.33** | **0.77** |

of topological information reduces the demands on the control module, enabling it to focus on short-term planning and obstacle avoidance.

### 5.4 Ablation study

In Table 2, we show the ablation study over different navigation framework components. VF-Nav represents the complete algorithm. VF-Nav(wo topo) removes the topological planning module and directly inputs the final target point into the control module. VF-Nav(wo topo wo CANN) removes the topological planning module and uses a maximum likelihood matching method for localization. VF-Nav(wo topo wo loc-module) removes the topological planning module and uses odometry accumulation to determine the current location. The topological planning module demonstrates its advantages in improving the navigation success rate and efficiency. The poor result of the method that relies solely on odometry without our localization module indicates that our method does not require high-accuracy odometry. Compared with the method without the CANN models and solely relying on maximum similarity matching, the proposed approach still demonstrates superior performance in navigation tasks. This is primarily because the mapping search method tends to fail in localization when structurally similar sampled points exist in the map, causing worse navigation performance, whereas our proposed method is robust in such cases. The failures of the two comparison methods in localization are discussed in greater detail in Subsection 5.5.

**Table 2** Ablation study over framework components. Without CANN, the map sample point with the highest similarity to the current image feature descriptor is selected as the localization point. Without the localization module, the current position is obtained by accumulating the results from the noisy odometry. The best results are in bold.

| Method | Success rate (%) | SPL (%) | SoftSPL (%) | $d_G$ (m) |
|---|---|---|---|---|
| VF-Nav | **88.71** | **71.54** | **68.33** | **0.77** |
| VF-Nav(wo topo) | 79.84 | 54.32 | 51.83 | 1.26 |
| VF-Nav(wo topo wo CANN) | 68.95 | 44.85 | 42.89 | 1.75 |
| VF-Nav(wo topo wo loc-module) | 43.55 | 26.70 | 31.39 | 1.92 |

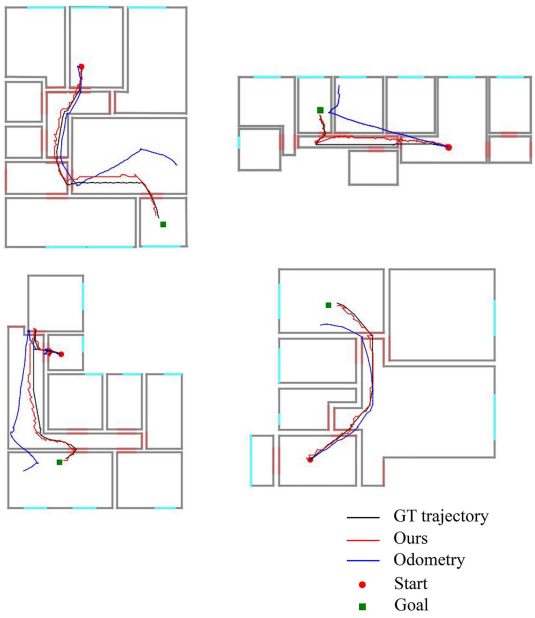


**Figure 7** (Color online) Examples of localization results of the proposed brain-inspired localization method (red) and accumulation of noisy odometry (blue), where the ground-truth trajectories are in black.

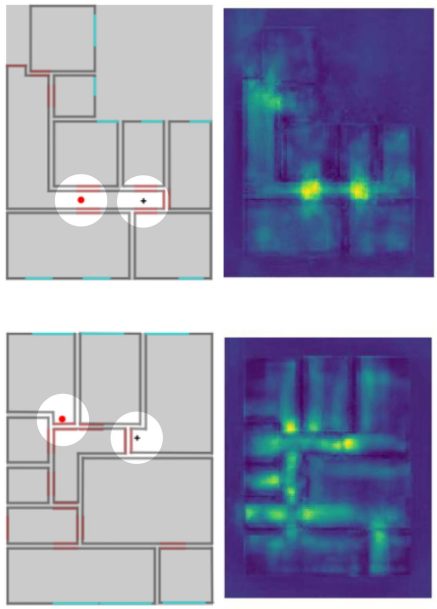

**Figure 8** (Color online) Some examples of localization failure in the maximum similarity matching method. The red dots represent estimated positions, whereas the black crosses represent true positions.

### 5.5 Localization performance

We compare the proposed localization module with the methods that separately use noisy odometry and maximum similarity matching for localization. Their deployment details are as follows.

**Odometry.** The current position is obtained by accumulating results from noisy odometry and derivation at each step following (7). For evaluation convenience, the noise of odometry is set to be consistent with that in the motion model of the brain-inspired model.

**Matching.** This method selects the map sample point with the highest similarity to the current image feature descriptor as the localization point.

In Figure 7, we present some examples of the results of the proposed localization method and the accumulation of noisy odometry. Although the level of noise is the same, the proposed method effectively corrects the accumulated position drift caused by the noise. In longer trajectories, the drift of the accumulation of the noisy odometry becomes more pronounced to the extent that the estimated position may even be in a different room from the real position, thereby adversely affecting navigation tasks. This indicates that the proposed method does not have high requirements for odometry and that robust localization can be achieved using just a low-accuracy odometry.

To further evaluate the proposed localization module, we collected trajectories whose total length was 6103.08 m and evaluated the localization accuracy. We conducted tests on the proposed method, the maximum similarity matching method, and the noisy odometry method. Table 3 shows that the proposed brain-inspired localization method had an outstanding performance, with almost no significant drift. Because of the presence of structurally similar sampled points in the map, the maximum similarity matching method may fail, as shown in Figure 8. This is why the mapping search method performs poorly in navigation. However, the proposed localization method integrates structural information from the floor

**Table 3** Localization performance results. The proposed brain-inspired localization method has outstanding performance on the trajectories whose total length is 6103.08 m, achieving near-perfect recall. The best results are in bold.

| Metric | Ours | Matching | Odometry |
|---|---|---|---|
| t_error (m) | **0.18** | 0.65 | 0.88 |
| r_error (°) | **12.80** | 21.96 | 20.66 |
| 1-m recall (%) | **98.75** | 87.22 | 73.67 |
| 0.5-m recall (%) | **97.06** | 84.11 | 55.13 |
| 0.25-m recall (%) | **87.52** | 76.79 | 34.79 |


**Figure 9** (Color online) (a) Equipment used in the real-world experiments and (b) some examples of the test scenarios.

**Table 4** Real-world localization performance results. The best result is in bold.

| Metric | Ours | VINS-Mono | Wheel odometry |
|---|---|---|---|
| t_error (m) | **0.21** | 2.53 | 5.13 |

plan and positional information between consecutive frames, achieving stable long-term localization.

### 5.6 Real-world experiments

We validated the localization performance of the proposed algorithm in real-world scenarios by collecting four trajectories totaling 434.73 m. A panoramic camera was mounted on an Autolabor robot, using the Autolabor's wheel encoder as the odometry sensor (see Figure 9). We ran a LiDAR-inertial odometry [47] to provide ground-truth localization. Our method was compared with an existing classical geometric visual-inertial odometry method, VINS-Mono [48], and a method that only used wheel odometry. The experimental results are shown in Table 4. For more information about the localization trajectories, please refer to Figure B1. The results show that our method is robust in handling real-world challenges such as lack of texture information, odometry drift, and camera shake.

## 6 Conclusion

In this paper, we propose a vision-only navigation framework for point-goal navigation using a 2D floor map as prior information. We adopt the classical localization-planning-control framework, and the proposed localization module incorporates learned features into a brain-inspired model. The proposed brain-inspired localization method surpasses the pure odometry and maximum similarity matching methods and achieves low localization error below 0.2 m with respect to the ground truth. The planning module leverages topological information extracted from the floor plan for the control module, guaranteeing efficient and reliable navigation and demonstrating superior robustness in complex tasks compared with end-to-end methods.

The proposed method still has limitations, such as the requirement of panoramic images to perform localization. Future work aims to include the study of a localization method using a single-viewpoint regular camera. As a single-viewpoint camera provides limited information, a possible solution is the fusion of images from multiple time steps to achieve robust monocular-camera-based localization on a floor plan. Some future challenges include determining the states of doors (open or closed) and incorporating dynamic planning accordingly. Furthermore, incorporating real robots' dynamic models requires finer control over their movements, including velocity and angular velocity, rather than relying on a discrete action space.

**Acknowledgements** This work was supported in part by National Natural Science Foundation of China (Grant No. 62273195).

**Supporting information** Appendixes A and B. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

**References**

1 Song X, Peng Z, Song S, et al. Anti-disturbance state estimation for PDT-switched RDNNs utilizing time-sampling and space-splitting measurements. Commun Nonlinear Sci Numer Simul, 2024, 132: 107945
2 Song X, Song Y, Stojanovic V, et al. Improved dynamic event-triggered security control for T-S fuzzy LPV-PDE systems via pointwise measurements and point control. Int J Fuzzy Syst, 2023, 25: 3177–3192
3 Savva M, Kadian A, Maksymets O, et al. Habitat: a platform for embodied AI research. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019. 9339–9347
4 Wijmans E, Kadian A, Morcos A, et al. DD-PPO: learning near-perfect PointGoal navigators from 2.5 billion frames. 2019. ArXiv:1911.00357
5 Gupta S, Davidson J, Levine S, et al. Cognitive mapping and planning for visual navigation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017. 2616–2625
6 Anderson P, Chang A, Chaplot D S, et al. On evaluation of embodied navigation agents. 2018. ArXiv:1807.06757
7 Kojima N, Deng J. To learn or not to learn: analyzing the role of learning for navigation in virtual environments. 2019. ArXiv:1907.11770
8 Mishkin D, Dosovitskiy A, Koltun V. Benchmarking classic and learned navigation in complex 3D environments. 2019. ArXiv:1901.10915
9 Ramakrishnan S K, Gokaslan A, Wijmans E. Habitat-Matterport 3D Dataset (HM3D): 1000 large-scale 3D environments for embodied AI. 2021. ArXiv:2109.08238
10 Partsey R, Wijmans E, Yokoyama N, et al. Is mapping necessary for realistic PointGoal navigation? In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022. 17232–17241
11 Mur-Artal R, Tardos J D. ORB-SLAM2: an open-source SLAM system for monocular, stereo, and RGB-D cameras. IEEE Trans Robot, 2017, 33: 1255–1262
12 Engel J, Schöps T, Cremers D. LSD-SLAM: large-scale direct monocular slam. In: Proceedings of the European Conference on Computer Vision (ECCV), 2014. 834–849
13 Honerkamp D, Guttikonda S, Valada A. Active particle filter networks: efficient active localization in continuous action spaces and Large maps. 2022. ArXiv:2209.09646
14 Boniardi F, Caselitz T, Kümmerle R, et al. A pose graph-based localization system for long-term navigation in CAD floor plans. Robot Auton Syst, 2019, 112: 84–97
15 Zimmerman N, Wiesmann L, Guadagnino T, et al. Robust onboard localization in changing environments exploiting text spotting. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022. 917–924
16 Min Z, Khosravan N, Bessinger Z, et al. Laser: latent space rendering for 2D visual localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022. 11122–11131
17 Howard-Jenkins H, Ruiz-Sarmiento J, Prisacariu V A. LaLaLoc: latent layout localisation in dynamic, unvisited environments. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021. 10107–10116
18 Kavraki L E, Svestka P, Latombe J C, et al. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans Robot Automat, 1996, 12: 566–580
19 LaValle S M, Kuffner J J, Donald B R. Rapidly-exploring random trees: progress and prospects. Algorithmic Comput Robot: New Dir, 2001, 5: 293–308
20 Xia F, Zamir A R, He Z, et al. Gibson Env: real-world perception for embodied agents. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018. 9068–9079
21 Schulman J, Wolski F, Dhariwal P, et al. Proximal policy optimization algorithms. 2017. ArXiv:1707.06347
22 Datta S, Maksymets O, Hoffman J, et al. Integrating egocentric localization for more realistic point-goal navigation agents. In: Proceedings of the Conference on Robot Learning, 2021. 313–328
23 Zhao X, Agrawal H, Batra D, et al. The surprising effectiveness of visual odometry techniques for embodied PointGoal navigation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021. 16127–16136
24 Chaplot D S, Gandhi D, Gupta S, et al. Learning to explore using active neural SLAM. 2020. ArXiv:2004.05155
25 Chaplot D S, Salakhutdinov R, Gupta A, et al. Neural topological SLAM for visual navigation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020. 12875–12884
26 Segal A, Haehnel D, Thrun S. Generalized-ICP. In: Proceedings of Robotics: Science and Systems, Seattle, 2009. 435
27 Biber P, Straßer W. The normal distributions transform: a new approach to laser scan matching. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2003. 2743–2748
28 Dellaert F, Fox D, Burgard W, et al. Monte Carlo localization for mobile robots. In: Proceedings of the IEEE International Conference on Robotics and Automation, 1999. 1322–1328
29 Howard-Jenkins H, Prisacariu V A. LaLaLoc++: global floor plan comprehension for layout localisation in unvisited environments. In: Proceedings of the European Conference on Computer Vision (ECCV), 2022. 693–709
30 Ronneberger O, Fischer P, Brox T. U-Net: convolutional networks for biomedical image segmentation. In: Proceedings of the 18th International Conference on Medical Image Computing and Computer-Assisted Intervention, 2015. 234–241
31 Hafting T, Fyhn M, Molden S, et al. Microstructure of a spatial map in the entorhinal cortex. Nature, 2005, 436: 801–806
32 O'Keefe J, Conway D H. Hippocampal place units in the freely moving rat: why they fire where they fire. Exp Brain Res, 1978, 31: 573–590
33 Taube J S, Muller R U, Ranck J J B. Head-direction cells recorded from the postsubiculum in freely moving rats. I. Description and quantitative analysis. J Neurosci, 1990, 10: 420–435

34 Taube J S, Muller R U, Ranck J J B. Head-direction cells recorded from the postsubiculum in freely moving rats. II. Effects of environmental manipulations. J Neurosci, 1990, 10: 436–447
35 Milford M J, Wyeth G F, Prasser D. RatSLAM: a hippocampal model for simultaneous localization and mapping. In: Proceedings of the IEEE International Conference on Robotics and Automation, 2004. 403–408
36 Milford M J, Wyeth G F. Mapping a suburb with a single camera using a biologically inspired SLAM system. IEEE Trans Robot, 2008, 24: 1038–1053
37 Milford M, Wyeth G. Persistent navigation and mapping using a biologically inspired SLAM system. Int J Robotics Res, 2010, 29: 1131–1153
38 Silveira L, Guth F, Drews-Jr P, et al. An open-source bio-inspired solution to underwater SLAM. IFAC-PapersOnLine, 2015, 48: 212–217
39 Yu F, Shang J, Hu Y, et al. NeuroSLAM: a brain-inspired SLAM system for 3D environments. Biol Cybern, 2019, 113: 515–545
40 He K, Zhang X, Ren S, et al. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016. 770–778
41 Schroff F, Kalenichenko D, Philbin J. FaceNet: a unified embedding for face recognition and clustering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022. 815–823
42 Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput, 1997, 9: 1735–1780
43 Fernandez-Chaves D, Ruiz-Sarmiento J R, Jaenal A, et al. Robot@VirtualHome, an ecosystem of virtual environments and tools for realistic indoor robotic simulation. Expert Syst Appl, 2022, 208: 117970
44 Hart P, Nilsson N, Raphael B. A formal basis for the heuristic determination of minimum cost paths. IEEE Trans Syst Sci Cyber, 1968, 4: 100–107
45 Sethian J. A fast marching level set method for monotonically advancing fronts. Proc Natl Acad Sci USA, 1996, 93: 1591–1595
46 AKoenig S, Likhachev M. D* Lite. In: Proceedings of the 18th National Conference on Artificial Intelligence, 2002. 476–483
47 Chen K, Nemiroff R, Lopez B T. Direct LiDAR-inertial odometry: lightweight LIO with continuous-time motion correction. In: Proceedings of the IEEE International Conference on Robotics and Automation, 2023. 3983–3989
48 Qin T, Li P, Shen S. VINS-Mono: a robust and versatile monocular visual-inertial state estimator. IEEE Trans Robot, 2018, 34: 1004–1020