SpatialMind: Spatially Aware On-Device Embodied AI via Viewpoint Integration

Haoming Wang, Qiyao Xue, Weichen Liu, Boyuan Yang, Xiangyu Yin and Wei Gao
University of Pittsburgh

Abstract

On-device embodied AI allows local processing of visual data and prompt responses to user requests. Applying embodied AI to more complex domains requires spatial cognition, to properly comprehend the 3D environment and geometric relations between objects. However, existing vision-language models (VLMs) are weak in spatial cognition due to their limitations to 2D image understandings. In this paper, we present SpatialMind, a new on-device embodied AI technique that bridges the gap between 2D visual inputs and 3D physical reality, by injecting 3D spatial knowledge into on-device VLMs in a sparse form of 2D allocentric spatial memory. This memory is constructed by systematically aligning cross-frame visual features to a unified global viewpoint, and we prune the scope of on-device computation to maximize the compute efficiency in memory construction and VLM reasoning. Experiment results in diverse indoor environments show that SpatialMind greatly enhances the performance of spatially aware on-device embodied AI tasks, with high adaptability, robustness and compute efficiency.

CCS Concepts

• **Computer systems organization** → **Embedded and cyber-physical systems**; • **Computing methodologies** → **Artificial intelligence**.

Keywords

On-Device Embodied AI, Vision Language Models, Spatial Cognition, Viewpoint Integration

ACM Reference Format:
Haoming Wang, Qiyao Xue, Weichen Liu, Boyuan Yang, Xiangyu Yin and Wei Gao. 2026. SpatialMind: Spatially Aware On-Device Embodied AI via Viewpoint Integration. In *The 32nd Annual International Conference on Mobile Computing and Networking (MobiCom '26), October 26–30, 2026, Austin, TX, USA.* ACM, New York, NY, USA, 19 pages. https://doi.org/10.1145/3795866.3844467

MobiCom '26, Austin, TX, USA

ACM ISBN 979-8-4007-2505-0/2026/10
https://doi.org/10.1145/3795866.3844467

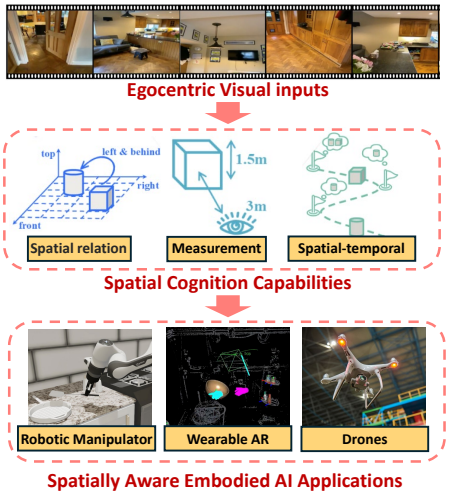


Figure 1: Spatially aware embodied AI relies on on-device VLM's ability of spatial cognition to extract core spatial patterns from egocentric video inputs

1 Introduction

Today's embodied and mobile devices, such as smartphones, AR/VR devices [38, 53, 55, 83] and robot-mounted cameras [34, 85, 91], widely use Vision-Language Models (VLMs) [11, 66, 72] to interpret visual observations of the physical world. Running these VLMs locally allows visual data to remain on the device and enables spatial reasoning without continuous dependence on cloud connectivity [31, 42, 62, 66, 93].

Currently, on-device VLMs have excelled at appearance-based perception tasks from 2D visual features (e.g., color, texture, shape), such as object detection [9, 18, 39] and semantic segmentation [28, 39, 75]. However, as embodied AI expands into more complex and interactive domains, simply detecting and recognizing objects is no longer sufficient; instead, the models must possess profound capabilities in **spatial cognition**, to comprehend the 3D physical properties of the environment and geometric relations (e.g., distance, orientation, and relative positioning) between objects in different views. This capability, as shown in Figure 1, is an important perceptual primitive for many embodied applications [21, 40]. For instance, before a service robot can execute a command such as "grab the cup beside the laptop", its perception system must determine the relative locations of the referenced objects across changing viewpoints. Similarly, an AR assistant may need to estimate the dimensions

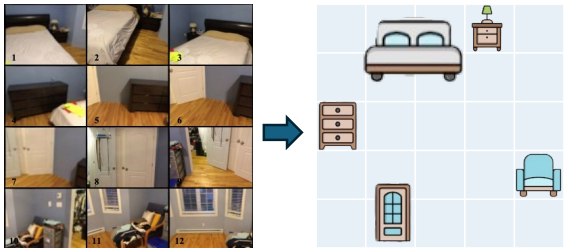


Figure 2: SpatialMind transforms egocentric video frames into a sparse allocentric 2D spatial memory

or relative positions of nearby objects before a downstream application can make an interaction decision.

Unfortunately, the performance of current VLMs is inadequate in these tasks [13, 26, 48], because they are mainly designed and trained to encode flat 2D image frames as token sequences. They might succeed at determining 2D pixel-coordinate relations within images, but fail when perceiving the 3D physical world through continuous egocentric observations across multiple views [22, 74, 84]. Especially when on-device embodied AI can only support VLMs with small parameter sizes[1] due to the limited onboard battery life, thermal threshold and memory capacity, the performance of on-device VLMs is far below the human level, even in simple tasks that involve only few objects and views [57, 63, 64, 72]. This limitation becomes the key barrier for transforming embodied AI from lab prototypes to practical products.

Recent cloud-based VLMs (e.g., GPT-5 and Gemini-3-Pro), on the other hand, started involving 3D scenes into training data, to gain spatial abilities [58] by building 3D representations in their large embedding spaces. However, these large proprietary models cannot be used for on-device deployment, and remote access to cloud-based VLMs involve extra concerns about communication latency and data privacy.

To address such limitation without involving large VLM parameter sizes and extra compute costs, a paradigm shift in how on-device VLM perceives and interacts with the physical world is needed. In this paper, we present *SpatialMind*, a zero-shot [2] VLM inference technique for on-device embodied AI, which bridging the gap between 2D egocentric visual inputs and 3D physical reality by transforming how spatial knowledge is injected into and processed by lightweight on-device VLMs. Rather than forcing a small on-device VLM to implicitly deduce the 3D geometry from unstructured video pixels, as shown in Figure 2, SpatialMind explicitly simplifies the 3D scene into an **allocentric spatial memory**, by integrating the 3D knowledge contained in different viewpoints into a unified 2D representation. For computationally efficient on-device deployment, this allocentric spatial memory is designed as a sparse top-down grid, and is constructed by extracting the shifting and fragmented 3D coordinates from multiple egocentric frames into a global coordinate system. By feeding this 2D memory to the VLM via a carefully crafted visual prompt, SpatialMind allows the on-device VLM to effortlessly deduce complex spatial relations among objects in the scene, without involving difficult 3D cognition.

This spatial memory is locally constructed, but construction via on-device VLM's native reasoning is infeasible due to its limited representation power. Instead, we develop a specialized pipeline to incrementally populate the spatial memory by systematically aligning cross-frame visual features. First, we utilize lightweight on-device perception models (e.g., efficient segmentation and depth estimators) to extract 3D spatial features from individual video frames. Next, we establish cross-frame visual correlations using lightweight image matching within the bounding boxes of task-relevant objects. Crucially, rather than relying on naive sequential frame integration which inevitably accumulates severe spatial drift over time, SpatialMind introduces a topology-aware multi-frame alignment strategy. By organizing the video frames into a Maximum Spanning Tree (MST) based on visual similarity and anchoring them to a central root frame, SpatialMind computes global transformations exclusively along high-confidence paths. This topology naturally isolates alignment errors to leaf nodes, prevents drift propagation, and enables recovery of occluded or partially visible objects by projecting geometric priors from aligned viewpoints.

To further improve the compute efficiency, we avoid processing every single video frame, and instead develop an approach of iterative key frame selection by leveraging the inherent temporal locality of objects' appearances in video. More specifically, we iteratively update the sampling distribution being used to select key frames according to the visual similarity among frames, and hence allow SpatialMind to progressively focus on the most informative video segments, with respect to the current on-device compute constraints.

To our best knowledge, SpatialMind is the first that empowers small on-device VLMs with complex and cross-frame spatial cognition capabilities. SpatialMind does not introduce new visual perception models; instead, its main contribution lies in the systematic integration of the existing visual perception modules into a query-conditioned, resource-aware, and VLM-compatible spatial cognition pipeline.

We deployed and evaluated SpatialMind across a variety of embodied AI platforms, including NVIDIA Jetson Orin and Oneplus Ace 3 Pro smartphone. Since we focus specifically on evaluating SpatialMind's capability of spatial cognition

[1]Edge computing devices, such as smartphones, AR/VR devices or NVIDIA Jetson Orin, are generally restricted to supporting VLM inference at most 3B parameters, within their standard 8GB memory constraint [11].

[2]We refer "zero-shot" to the absence of task-specific or scene-specific VLM fine-tuning. With that being said, we use pre-trained perception modules for object grounding, segmentation, depth estimation, and image matching.

rather than its downstream generation and control of embodied actions, our evaluations directly measures the performance of SpatialMind on spatial reasoning tasks, allowing us to isolate the VLM's ability of integrating multi-view geometric information without confounding it with motor-control or hardware-specific failures. From experiment results in diverse indoor environments (residential homes, offices, libraries, etc), we have the following conclusions:

- SpatialMind significantly *improves embodied AI task accuracy*. On difficult embodied AI tasks that involve spatial information in multiple views, SpatialMind achieves up to 40% higher accuracy compared to competitive baselines of video understanding.
- SpatialMind is *highly adaptive and robust*. It can reliably adapt to different types of complex indoor scenes with severe occlusions and partial visibility, and experiences 30% less performance loss than existing schemes.
- SpatialMind is *computationally efficient*, and can be executed on most embodied AI platforms. On NVidia Jetson Orin, it reduces memory usage by 26% and compute latency by 5% compared to competitive baselines.

2 Background and Motivation

2.1 Deficiency of VLMs in Spatial Cognition

Existing VLMs struggle in spatial cognition tasks [26, 48, 57]. Unlike humans who can naturally form 3D allocentric "mental maps" to establish cross-view correspondence, VLMs process flat 2D token sequences [24, 25]. Consequently, they fail to maintain spatial consistency across shifting viewpoints, as shown in Figure 3. This ability of spatial cognition also scales with the VLM's parameter size [76]. Existing studies show that, on relatively simple tasks such as estimating the object sizes, a state-of-the-art VLM (e.g., GPT-5 or Gemini3-Pro) can achieve human-level performance, but a small VLM with 3B parameters barely outperforms a random guess [13, 61]. As shown in Figure 4, for small on-device VLMs (e.g., Qwen3-VL-8B), the deficiency in spatial cognition[3] is much more severe due to their limited representation power and the subsequent lack of visuospatial commonsense [78]. Clearly, deploying a large VLM on embodied AI devices is impractical, and this constraint motivates us to improve the spatial cognition of small on-device VLMs beyond their representation power.

Recent analysis further showed that VLMs do not develop an internal 3D understanding [13, 20, 90]; instead, their spatial cognition relies on 2D positional encodings (e.g., RoPE) within the visual encoder, leaving them structurally

[3]Results shown in Figure 4 is a diagnostic comparison based on the spatial-cognition task taxonomy in MMSI-Bench [78]. We evaluate GPT-5 and Qwen3-VL-8B on the same benchmark categories and use human performance reported by MMSI-Bench as the human baseline. For visualization, category-level scores are normalized independently on each axis.

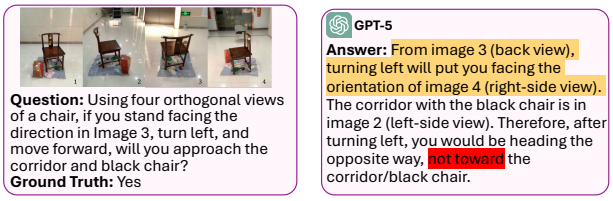


Figure 3: Deficiency of VLMs in spatial cognition: a multi-view transition task where strong VLMs like GPT-5 fail to build correct cross-view correspondence

unequipped for 3D spatial understanding. Therefore, merely scaling up the training data is ineffective to resolve the VLMs' deficiency in spatial cognition, and instead motivates us to *externalize* the 3D spatial cognition process, i.e., turning the 3D representations to 2D spatial memory, which is easier for VLMs to perceive due to its alignment with VLM's inherent reliance on explicit 2D positional features.

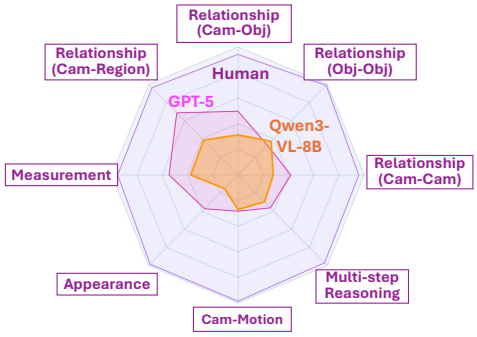


Figure 4: Existing VLMs exhibit severe deficiencies in spatial cognition compared to humans, and the small on-device VLM (Qwen3-VL-8B) performs the worst

2.2 Other Sensory Modalities as VLM Inputs

Depth map [92] and point clouds [36] produced by LiDAR and mm-Wave can also be used as VLM inputs to provide more spatial information and hence enhance VLM's capability in spatial cognition. However, hardware devices with sufficient sensing accuracy are often too expensive for low-cost embodied AI devices such as small robots and AR/VR headsets [52, 53]. While software-based monocular depth estimators [4, 77] offer a lightweight alternative, their estimation errors escalate substantially in the far field [23, 86]. For instance, experiments in [86] show that errors of depth estimation largely rises once the normalized depth range exceeds 60%. Due to this discrepancy, rather than directly using the raw depth estimations to calculate object locations, in SpatialMind we iteratively align depth estimates (from RGB video captures) across frames, and only use the aligned depth point to construct the spatial memory.

Furthermore, even when accurate depth maps or point clouds are available, they cannot be directly used as visual

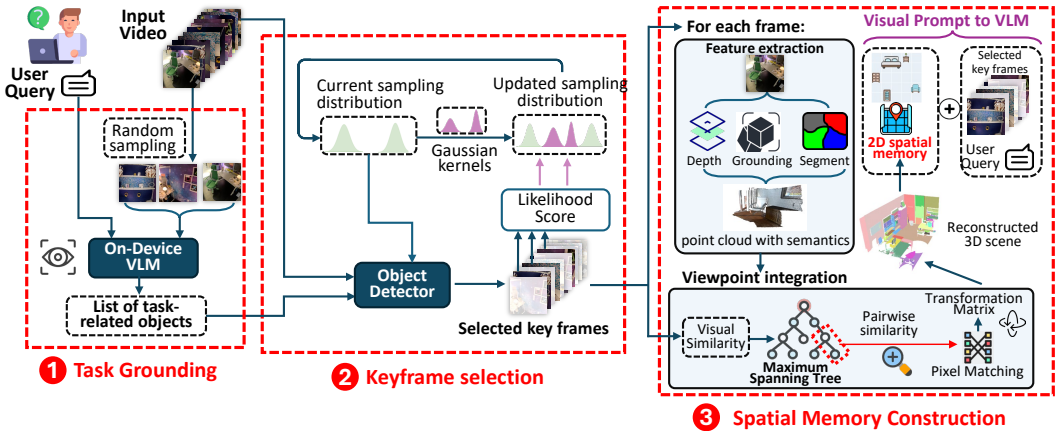


Figure 5: Overview of SpatialMind design

input to small on-device VLMs (3B-7B parameters), which are not trained to parse continuous 3D matrices. When fed dense depth maps as pseudo-RGB inputs, these small VLMs will collapse precise geometric metrics into rudimentary foreground/background zones [46, 48]. This deficiency drives the design of SpatialMind. Instead of forcing the VLM to process error-prone depth arrays, we decouple 3D geometry extraction from language reasoning, and translate complex 3D scenes into the VLM's native 2D memory.

3 SpatialMind Design

SpatialMind is designed for indoor spatial cognition at the room scale from recent on-device egocentric RGB observations, without any preloaded 3D scan, semantic map, scene graph, or environment-specific object database. At runtime, it conducts a spatial cognition task with a received natural-language query q and a recent video buffer $\mathcal{V} = I_1, \ldots, I_N$ with N video frames, acquired through normal device or camera motion. Unlike visual tasks requiring high metric precision such as robotic grasping [43] or dense 3D reconstruction [45], SpatialMind's tasks include object localization, metric measurement, relative object relations (e.g., left/right, near/far, adjacent) and cross-view perspective reasoning, and the resulting spatial memory may provide geometric context to the downstream embodied applications such as action generation, active navigation, and closed-loop manipulation.

Note that, the video buffer may not fully cover the indoor environment and could only contain the region observed during the recent acquisition interval, but we assume that the objects of interest are visible in the video buffer, so that the available visual cues make the given query to be answerable. We assume moderate camera motion and static or slowly changing scene geometry during this interval, although partial visibility and occasional occlusions may occur.

Since processing the entire raw video is too expensive for embodied AI devices, SpatialMind instead enforces a computationally efficient pipeline that procedurally prunes the scope of on-device computation, as shown in Figure 5. First, it extracts task-relevant objects and filters redundant visual data via *Task Grounding*. Next, it uses *Key Frame Selection* to sample video segments within the compute budget. Lastly, within the pruned scope of computation, we use *Viewpoint Integration* to build the allocentric 2D spatial memory from the 3D scene layout, and inject the spatial memory via a structured visual prompt to VLM, effectively driving its spatial cognition in the 2D space.

3.1 Task Grounding

Instead of feeding the entire raw video to on-device VLM, we first establish what the embodied AI system looks for, and then filter the input video to only involve the needed spatial information. Given a user query, SpatialMind first identifies task-relevant objects. We prompt a lightweight on-device VLM with the text query alongside a small, randomly sampled subset of video frames, to disambiguate object attributes and scene context. As shown in Figure 6, the VLM will output a list of *target objects* that are directly mentioned in the query, ranked by their relevance to the query and likelihoods of appearance in the video, based on joint understanding of question semantics and visual priors inferred from the sampled frames. In addition, it will also provide a list of *cue objects* that are related to the current task and serve as spatial landmarks, and each cue object is associated with a significance weight indicating its relevance to the task.

We notice that adjacent frames in a video stream are often similar, and processing each of them yields diminishing returns. To further reduce the on-device compute workload, we will detect adjacent frames with high similarity and filter out redundant frames. Rather than running expensive deep-learning similarity metrics (e.g., SSIM [68]) on every frame, we use a hierarchical filtering strategy: we first apply ultra-lightweight motion-based keypoint tracking method (e.g., ORB/SIFT [49]) to detect candidate scene changes, and only invoke the costlier SSIM checks on this filtered subset.

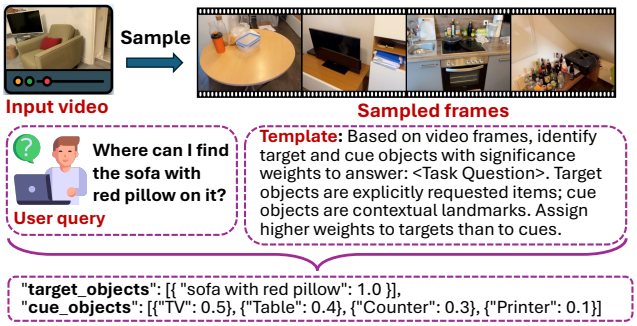


Figure 6: Example of task grounding

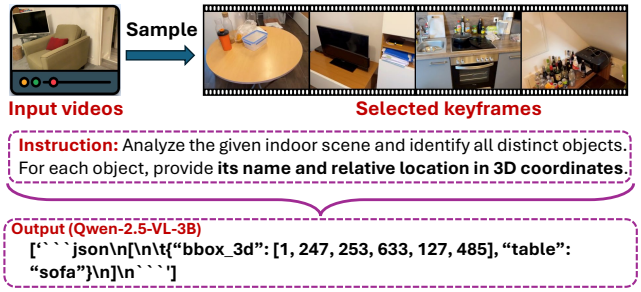


Figure 7: Ineffective generation of the 3D map by directly prompting the small on-device VLM

3.2 Iterative Key Frame Selection

Once the target and cue objects are identified, SpatialMind looks for video frames that contain these objects from the video stream. A straightforward solution to finding these key video frames is to score every video frame with an object detector such as GroundingDINO [39] or Yolo-World [9]), but doing so incurs massive compute overhead and large extra latency. Instead, to improve the compute efficiency of on-device embodied AI, SpatialMind enforces a runtime compute budget, defined as the number of key frames to be selected, based on the available compute resources and task-dependent requirements on the embodied AI device. Then, it employs an iterative temporal search to select a minimal yet sufficient set of key frames.

Our approach is to exploit the temporal locality in the video stream, i.e., if an object is detected in one frame, the chance of it appearing in adjacent frames is higher, due to the continuity of camera motion [44, 71, 88] (details are in Appendix A). As shown in Figure 5, key frame selection is conducted in multiple iterations. In each iteration, we score only a small subset of frames drawn from a sampling distribution. When a sampled frame yields a high score of object detection, we adaptively update the sampling distribution, by applying a Gaussian kernel to that frame's temporal index in the sampling distribution and dynamically scaling the kernel's amplitude and standard deviation based on cross-frame visual similarity. In this way, we allow SpatialMind to progressively zoom into the most informative video segments, while aggressively skipping other irrelevant segments. Details of such sampling distribution are in Section 4.1, and how to adapt key frame selection to the current compute budget is tackled in Section 4.2.

3.3 Viewpoint Integration

With the selected key frames, SpatialMind integrates fragmented visual information in these frames to reconstruct the 3D scene layout, which is then converted to a 2D spatial memory (§ 3.4). Directly prompting the small on-device VLM to output the structured 3D map for the entire scene, however, catastrophically fails. As shown in Figure 7, small VLMs lack sufficient representation power to accurately deduce 3D measurements and relative positions from image pixels, yielding incomplete object lists and hallucinated layouts.

Instead, SpatialMind first extracts spatial features using specialized vision models, and then procedurally aligns these fragmented features across multiple views into a unified global coordinate system. For each frame, a lightweight segmentation model (e.g., MobileSAM with 1B parameters [87]) isolates individual objects, and a compact depth estimator (e.g., ZoeDepth-N with 3.4B parameters [3]) projects them into a local 3D point cloud[4].

Since an object's observed location and orientation could shift across different egocentric views, *cross-frame alignment* is required to fuse all objects in different viewpoints into a global coordinate system. In particular, to optimize for the user's query of spatial cognition, we anchor this global coordinate system to the local coordinates of the frame that is most likely to contain the target objects, hence ensuring the allocentric spatial memory remains target-centric.

The foundational step of this alignment is estimating the camera transformation matrix $\mathbf{T}_{i\to j} \in SE(3)$ between a pair of frames i and j, where $SE(3)$ denotes the space of 3D rigid transformations [56] containing both translation and rotation. Specifically, the transformation is decomposed as:

$$\mathbf{T}_{i\to j} = \begin{bmatrix} \mathbf{R}_{i\to j} & \mathbf{t}_{i\to j} \\ \mathbf{0}^\top & 1 \end{bmatrix},$$

where $\mathbf{R}_{i\to j} \in SO(3)$ is the rotation matrix and $\mathbf{t}_{i\to j} \in \mathbb{R}^3$ is the camera translation between frames. By applying these transformations, points in a frame's local coordinate system of can be projected into a global coordinate system, fusing fragmented views into a coherent cognitive map.

Conventionally, the transformation matrix $\mathbf{T}_{i\to j} \in SE(3)$ between two frames is computed using Iterative Closest Point (ICP) algorithms for 3D point cloud matching [12]. However, ICP is highly unreliable under strict on-device compute constraints which only allows a sparsely budgeted

[4]These lightweight models can efficiently locate object coordinates with high accuracy, e.g., 75% mIoU on COCO dataset [35] for MobileSAM, and 0.05 relative error on NYU Depth V2 dataset [51] for ZoeDepth-N.

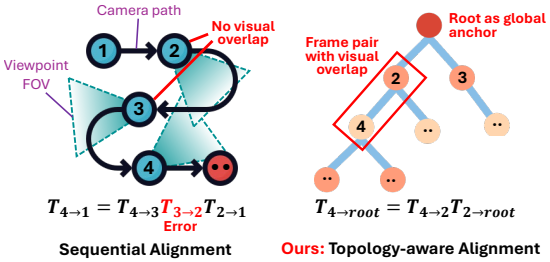


Figure 8: Different methods of cross-frame alignment

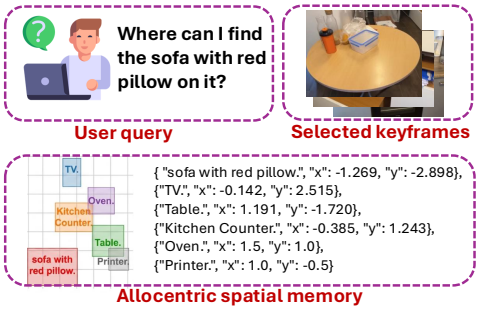


Figure 9: Full input being injected to the VLM

set of video frames for segmentation and depth estimation, and the 3D overlapping regions between these frames are often too minimal for ICP to converge accurately. To address this limitation, SpatialMind use a pretrained matching model, such as MatchAnything [15], to identify the overlapping regions first and then only compute the transformation matrix based the matching pixels from task related objects. More details about such matching-based computation of the transformation matrix are presented in Section 5.1.

To integrate these pairwise estimates into the global coordinate system, a naive approach is sequential integration. As shown in Figure 8, temporal adjacency in a video does not guarantee visual similarity, and chaining transformations in temporal order is susceptible to cumulative spatial drift. Instead, SpatialMind employs a topology-aware approach to multi-frame alignment. We organize key frames into a Maximum Spanning Tree (MST) rooted at the central "global anchor" frame, where edges denote the maximum visual overlap between nodes. By computing global poses via unique paths to the root rather than following the chronological video order, we ensure transformations are derived exclusively from high-confidence pairs. This prevents error propagation and safely isolates mismatched outliers at the branch ends. Details about such alignment are in Section 5.2.

Another challenge that may affect the matching accuracy is the possible occlusion or partial visibility of objects, which often prevents the segmentation model from detecting them. Once all frames are aligned, SpatialMind estimates the global camera pose (location and orientation) for every frame based on the calculated transformation matrices. Using these camera parameters, the system mathematically infers which objects logical fall within the current field of view (FoV) but were missed by the lightweight segmentation model due to partial visibility. The current matching results are then used to guide the segmentation model to reliably recover these occluded or partially visible object components. More details are provided in Section 5.3.

3.4 Injection of Spatial Memory to the VLM

The final step of SpatialMind's pipeline is to allow the on-device VLM to effectively comprehend the spatial information from integrated viewpoints. Directly feeding raw and dense 3D point clouds is impractical, as VLMs would require additional training to perceive this unseen input modality. Furthermore, rendering the point clouds as RGB images is also unsuitable, since unavoidable errors during the integration phase would result in blurred visuals.

To bridge this gap, SpatialMind projects the 3D scene layout generated in § 3.3 into *allocentric spatial memory*, in the form of a simplified 2D top-down (BEV) spatial grid. To eliminate any visual ambiguity inherent to grid-based rendering, as shown in Figure 9, we further craft a visual prompt as the input to VLM, by pairing the grid with strict text-based bounding box coordinates[5] corresponding to the physical objects. Each object is assigned a unique color identifier. By feeding this unified and lightweight prompt to the VLM, SpatialMind can trigger complex spatial cognition and enable the device to deduce global spatial relationships effortlessly with the minimum compute overhead. Alternatives to this prompting method and their respective system trade-offs are discussed and evaluated in Appendix B.

4 Optimizing the Key Frame Selection

4.1 Applying the Gaussian Kernel

When updating the sampling distribution of key frame selection, we enforce the temporal locality by applying a Gaussian kernel to each scored frame, so that other frames that are temporally close to this scored frames will have higher chances to be scored. More specifically, letting the input video frames be indexed as $1, 2, \ldots N$ in time, the Gaussian kernel applied to a scored frame i can be written as

$$\textsc{GaussianKernel}(x;\, I_i, \sigma_i^2) = \exp\left(-\frac{(x - I_i)^2}{2\sigma_i^2}\right),$$

where I_i is the likelihood score of frame i, and the kernel width σ_i (i.e., the standard deviation) is adaptively determined by the average normalized similarity score (s_i), measured by histogram-based metrics [59, 89], between frame i and other frames within a time window centered at i. In practice, the proper size of time window depends on the specific

[5]We intentionally use bounding boxes rather than complex contours, to compress the context length and minimize the VLM's compute cost.

characteristics of temporal locality between frames, which is scene dependent. To further verify the temporal locality and quantize such time window, we conducted a small-scale benchmark study using the VSI-Bench benchmark that depicts indoor scenes [76]. Based on results in Appendix A, we set the size of time window (r) as 3, to ensure that the Gaussian kernel can capture short-range temporal correlations while avoiding other frames with weak semantic relevance.

To ensure alignment between the kernel spread and the window, we constrain the standard deviations as $\sigma_{\min} = 1/k$ and $\sigma_{\max} = r/k$, where $k = \sqrt{2}\,\mathrm{erf}^{-1}(I_i;\alpha)$ and $\alpha \in (0,1)$ specifies the desired fraction of probability mass within $[I_i - k\sigma_i, I_i + k\sigma_i]$. For example, setting $\alpha = 0.6827$ recovers $k \approx 1$ (the $1\sigma_i$ band), while $\alpha = 0.9545$ yields $k \approx 2$ (the $2\sigma_i$ band). Given these bounds, we interpolate the variance as

$$\sigma_i(s_i) \;=\; \sigma_{\min} \;+\; \big(\sigma_{\max} - \sigma_{\min}\big)\, s_i, \quad s_i \in [0,1] \qquad (1)$$

By setting $k = 2$, we obtain $\sigma_i(s_i) \in \left[\frac{1}{2}, \frac{r}{2}\right]$, which ensures that when frame similarity is low in the window, the kernel collapses towards a narrow peak around i within the adjacent frame index, restricting influence to adjacent frames; conversely, when the frame similarity is high, the kernel allocates most of its mass across the entire window, facilitating evidence propagation among semantically similar frames.

4.2 Adapting to the Runtime Frame Budget

In SpatialMind, key frame selection is not configured with a fixed operating point. Instead, the system treats the number of key frames (B_t) as a parameterized frame budget to be dynamically adjusted at runtime (t) according to the instantaneous condition of the embodied AI device, which may vary over time due to changes in battery level, thermal state, processor contention, and end-to-end latency requirements. In particular, in latency-sensitive applications such as assistive robotics, the embodied AI system can opt to retain bounded response time when the task deadline approaches, by adopting a smaller number of key frames [38, 50].

A key challenge is that this number of key frames may change while the 3D scene layout (§ 3.3) or the 2D spatial memory (§ 3.4) is already being updated from previous selections. Naively reselecting a completely new set of key frames would lead to unstable construction of spatial memory and abrupt shifts in VLM's reasoning. SpatialMind instead enforces a warm-transition policy that preserves continuity across successive updates of the frame budget. Specifically, it maintains a ranked candidate buffer that stores previously sampled frames, together with their accumulated utility scores. When the budget decreases, SpatialMind retains the frames with highest scores and prunes those with lowest scores. In contrast, when the budget increases, SpatialMind expands the set of key frames by including the most informative frames. Hence, the set of key frames evolves incrementally rather than being replaced abruptly.

Formally, let τ_{t-1} and τ_t denote the sets of key frame indices before and after a budget adjustment. SpatialMind aims to preserve temporal continuity by maximizing the overlap between two selections with respect to the new budget:

$$\max_{\tau_t} \; |\tau_t \cap \tau_{t-1}| \qquad \text{s.t.} \qquad |\tau_t| \le B_t. \qquad (2)$$

In practice, this objective of continuity is prioritized over frames' utility scores, and we implement it with a hysteresis rule: an existing key frame is removed only when its utility score falls sufficiently lower than a competing candidate, hence avoiding oscillations when the device condition fluctuates near a boundary. The spatial memory therefore degrades gracefully, rather than collapsing due to abrupt resampling.

With the Gaussian kernels applied and runtime frame budget adopted, our approach to iterative key frame selection is in Algorithm 1. In Appendix C, we also provide detailed analyses and discussions of system parameters being involved in Algorithm 1, including the threshold of frame's utility score (γ) and the number of temporal search iterations (B).

Algorithm 1 Key frame selection

Input: Video V of length L; targets $\mathcal{T}$; cues C; number of iterations B; select batch m; key frame selection threshold γ; window step r; Gaussian kernel std bounds $\sigma_{\min}, \sigma_{\max}$; scoring weights $w_{\mathcal{T}} > w_C > 0$

Output: Selected keyframes F with frame indices τ

Initialize: $S, N \leftarrow 0^L,\ 1^L;\ P \leftarrow \frac{1}{L}1^L;\ F, \tau \leftarrow \varnothing$
while $B > 0$ **do**
  $I \leftarrow \textsc{Sample}(P \odot N, m);\quad B \leftarrow B - m$
  $(\{d_i^\ell\}, O_i)_{i \in I} \leftarrow \textsc{Detect}(V[I])$
  for $i \in [1..|I|]$ **where** $O_i \cap (\mathcal{T} \cup C) \neq \varnothing$ **do**
    $\bar{s}_i \leftarrow \textsc{Similarity}(V, I_i, r)$
    $\sigma_i \leftarrow \sigma_{\min} + (\sigma_{\max} - \sigma_{\min})\, s_i$
    $K_i \leftarrow \textsc{GaussianKernel}(I_i, \sigma_i^2)$
    $\alpha_i \leftarrow \sum_{\ell \in O_i \cap \mathcal{T}} w_{\mathcal{T}}^\ell d_i^\ell + \sum_{\ell \in O_i \cap C} w_C^\ell d_i^\ell$
    $\widehat{K}_i \leftarrow \alpha_i \cdot K_i$
    $S \leftarrow S + \widehat{K}_i;\quad N[I_i] \leftarrow 0$
  $P \leftarrow \textsc{Normalize}(S \odot N)$
  for $j \in [1..L] \setminus \tau$ **where** $S[j] \ge \gamma$ **do**
    $F, \tau \leftarrow F \cup \{V[j]\},\ \tau \cup \{j\}$
return (F, τ)

5 Integration among Viewpoints

5.1 Pairwise Viewpoint Alignment

As described in Section 3.3, to efficiently compute the transformation matrix for pairwise viewpoint alignment, we employ deep feature-based image matching models to establish pixel correspondences across RGB images in these two

viewpoints, subsequently aligning the 3D coordinates of the matched points. To further minimize the alignment error, rather than utilizing all pixel correspondences identified by the model, we retain only those located within the pre-computed bounding boxes of task-relevant objects. This approach excludes irrelevant background regions (e.g., walls and floors) and hence results in higher alignment accuracy.

However, this method cannot ensure the accuracy of alignment for arbitrary frames. As demonstrated in Figure 10, the results are only reliable when the pair of frames exhibits significant visual overlap.

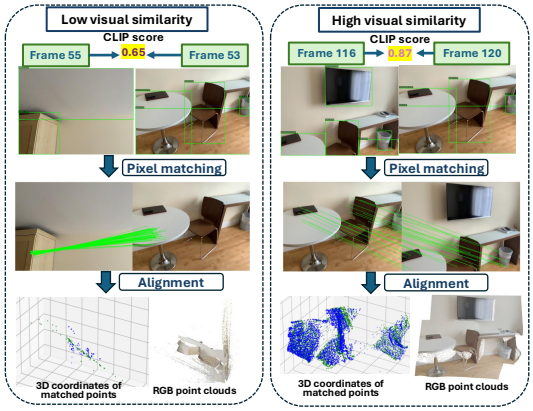


Figure 10: Results of pairwise alignment with different levels of visual similarities between frames, measured by cosine similarity between their CLIP embeddings

5.2 Multi-Viewpoint Alignment

Due to the above limitations, we cannot rely on naive sequential integration to merge these pairwise alignments into a globally consistent scene layout, as temporal adjacency of frames in a sparsely sampled video does not guarantee sufficient visual similarity. If the camera moves significantly between two selected frames, the transformation estimation may fail, and a single inaccurate estimation can propagate the estimation error through the sequence of other frames.

Therefore, we instead propose a topology-aware alignment strategy which organizes the selected key frames into a hierarchical tree structure with the maximum visual overlap between connected nodes in the tree that represent frames. This formulation ensures that global alignment relies exclusively on the estimates between frames with sufficient visual feature overlap. Furthermore, this tree topology provides inherent isolation in estimation errors, because frames that lack similarity with all other frames naturally correspond to different leaf nodes in the tree. As a result, any estimation errors associated with these outliers remain localized and do not propagate to corrupt the alignment of other frames.

Specifically, let $\mathcal{V} = \{1, \ldots, N\}$ be the set of indices for key frames, we define a fully connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where an edge $(i, j) \in \mathcal{E}$ indicates the visual similarity between

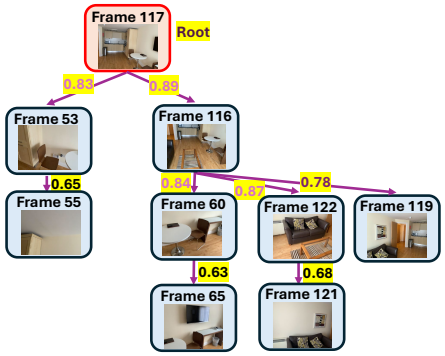


Figure 11: An example of the frame similarity tree

the two connected nodes (frames), via its associated weight w_{ij} as the cosine similarity between the CLIP embeddings of frame i and j. This metric provides a computationally efficient proxy for measuring semantic overlap, prior to detailed geometric matching. We then construct a *Tree* $\mathcal{T}$ from $\mathcal{G}$ to determine the optimal alignment as a path in $\mathcal{T}$. This process consists of three steps as described below:

1. Root Selection. We identify a global anchor frame (root), denoted as r, which serves as the origin of the global coordinate system for cross-frame alignment. The root is selected as the frame with the highest aggregate similarity to all other frames, ensuring maximal centrality defined as

$$r = \text{argmax}_{i \in \mathcal{V}} \sum_{j \in \mathcal{V}, j \neq i} w_{ij}. \tag{3}$$

2. Tree Construction. To ensure that transformations across viewpoints are exclusively estimated between visually similar pairs, we construct $\mathcal{T}$ as the MST of $\mathcal{G}$. The tree topology is optimized to maximize the sum of edge weights, such that

$$\mathcal{T} = \text{argmax}_{\mathcal{T}' \subseteq \mathcal{G}} \sum_{(i,j) \in \mathcal{E}(\mathcal{T}')} w_{ij}, \tag{4}$$

This optimization can be implemented by the Kruskal's algorithm ($O(N^2 \log N)$), which iteratively adds edges that have the highest similarity scores but do not form cycles. The constructed tree structure then ensures that every frame connects to the graph via its most reliable visual connections, and frames with low overall similarity are naturally pushed to the leaves. Figure 11 shows an example of the tree, where Frame 55 is positioned as a leaf node due to its low similarity with other frames. If we utilize the naive sequential alignment scheme, an inaccurate transformation estimation between Frames 53 and 55 would propagate errors to all subsequent frames; our tree topology prevents this cascade.

3. Computing the Global Transformation. For any frame i, there exists a unique path in $\mathcal{T}$ connecting it to the root r. Let this path be represented as a sequence of nodes $P_{i \to r} = (v_0, v_1, \ldots, v_m)$ where $v_0 = i$ and $v_m = r$, the global transformation matrix $\mathbf{M}_i \in SE(3)$, which maps points from frame i to the global coordinate system (frame r), is computed by compounding the pairwise transformations along this path:

$$\mathbf{M}_i = \mathbf{T}_{v_{m-1} \to v_m} \cdot \mathbf{T}_{v_{m-2} \to v_{m-1}} \cdots \mathbf{T}_{v_1 \to v_0}, \quad (5)$$

where $\mathbf{T}_{u \to v}$ is the relative transformation matrix derived from image matching between adjacent nodes u and v in the tree. By strictly following this path with the maximum similarity, we can isolate estimation errors from outlier frames at the branches' ends and prevents drift propagation.

5.3 Geometry-Guided Refinement

To handle object occlusion and partial visibility, we introduce a refinement stage following alignment. Using the calculated camera intrinsics and extrinsics, we identify objects that are geometrically expected to appear within a frame's FoV but are missing from the initial segmentation. For instance, if an object is successfully segmented in Frame A but remains undetected in Frame B, despite camera poses indicating it should be visible, we leverage cross-frame pixel correspondences. By matching pixels from the segmented region in Frame A to unsegmented areas in Frame B, we generate candidate locations for the missing object. These locations then serve as priors to guide the segmentation model, effectively recovering partially occluded or fragmented objects.

6 Experiment Settings

We deployed SpatialMind on multiple embodied AI platforms, including NVidia Jetson Orin and Oneplus Ace 3 Pro smartphone. We then apply VLMs with various parameter sizes to match the compute power constraints of each device: Qwen3-VL-32B-4bit [2] and InternVL3-8B [8] on NVidia Jetson Orin and Qwen-2.5-VL-3B [2] on the smartphone.

System setup. On NVidia Jetson Orin, VLM inference was operated using PyTorch with TensorRT acceleration, enabling optimized GPU execution. To deploy VLMs and run VLM inference on the smartphone, we used the ONNX Runtime [1] for Android, leveraging its NNAPI backend to enable heterogeneous hardware acceleration. This setup automatically switches between GPU and CPU execution based on workload characteristics, ensuring optimal performance while maintaining energy efficiency.

Execution scheduling. SpatialMind invokes multiple pretrained perception modules, but these modules are not loaded on a per-frame basis. Instead, selected key frames are processed in stage-level batches. For example, depth estimation is executed over the key-frame batch before subsequent feature extraction and cross-frame matching are done. This batching avoids repeatedly loading or switching perception modules for individual frames and reduces execution overhead. The end-to-end latency reported in Section 7 includes this model switching and scheduling overhead.

Benchmarks. Our experiments evaluate the spatial cognition capabilities of SpatialMind, rather than downstream robot control. We group the benchmark tasks into three categories: 1) *Metric Measurement* that encompasses distance, object size, and object location to verify the VLM's spatial understanding of scale and depth; 2) *Perspective-dependent Reasoning* that covers relative direction and object counting to test the spatial ability to perform mental rotation; and 3) *Spatiotemporal Tracking* that identifies the appearance order of objects in video to assess whether the VLM can maintain a persistent memory of object locations. These tasks are considered as representing the most fundamental spatial cognition capabilities that are transferrable to real-world applications [6]. We use two benchmark datasets that cover these tasks, as described below and shown in Figure 12.

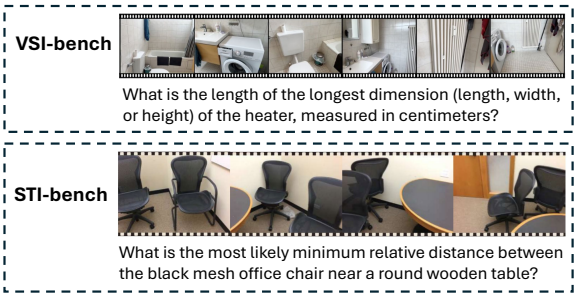


Figure 12: Data samples in benchmark datasets

- **VSI-Bench [76]** contains >5,000 question–answer (QA) pairs from 288 videos. It includes 8 spatial tasks and we use 6: absolute distance (A.D.), object size (O.S.), relative distance (R.Dt.), relative direction (R.Dr.), object count (O.C.) and appearance order (Ap.Or.).
- **STI-Bench [33]** contains 300 videos and >2,000 QA pairs for spatio-temporal video understanding. It includes 8 types of spatial tasks and we use two of them to complement those tasks in VSI-Bench: dimensional measurements (D.Meas.) and 3D video grounding (Grd.).

Figure 13: Real-world indoor scenes

Real-World Experiments: We also deployed our embodied AI systems into 5 different types of real-world scenes for experiments, as shown in Figure 13, where video streams are recorded by a hand-held camera and fed to on-device VLM for spatial reasoning. Two spatial tasks are evaluated in these scenes: Measurements (Meas.) and Perspective Taking (Pers.). Since these scenes capture the complexity of daily activities, they introduce challenging variables such as dynamic object distributions, varying lighting conditions, and frequent occlusions. More details of these scenes are in Appendix D.

We evaluate QA task performance as percentage of correct answer choices. For tasks deciding the absolute distance and object size, performance is evaluated using the Mean Relative Accuracy. Since SpatialMind is an inference-time technique,

Method	VSI-Bench							STI-Bench			Real-world		
	A.D.	O.S.	R.Dt.	R.Dr	O.C.	Ap.Or.	**Avg**	D.Meas.	Grd.	**Avg.**	Meas.	Pers.	**Avg.**
Direct Input	17.3	25.1	25.9	23.2	13.5	20.1	20.2	15.8	17.4	16.2	27.3	27.5	27.4
Video-CoT [17]	21.9	27.7	26.8	25.5	15.8	23.6	19.7	21.5	19.9	20.7	31.8	32.3	32.0
Scene Graph Reconstruction [10]	22.4	26.3	27.1	27.5	21.3	24.7	22.1	23.6	21.4	22.5	28.4	31.1	30.6
APC [29]	24.8	28.4	30.6	28.4	23.0	24.5	25.1	24.1	23.2	23.6	32.7	33.9	33.3
SpatialMind	29.4	36.6	38.9	33.5	28.3	32.2	33.4	30.8	31.8	31.6	45.5	40.1	42.8
Ground Truth spatial memory	30.3	37.7	39.5	34.0	31.7	32.4	34.6	31.3	32.9	32.1	-	-	-

Table 1: Performance on Oneplus 12R Smartphone with Qwen-2.5-VL-3B. Task acronyms are defined in Section 6.

Method	VSI-Bench							STI-Bench			Real-world		
	A.D.	O.S.	R.Dt.	R.Dr	O.C.	Ap.Or.	**Avg**	D.Meas.	Grd.	**Avg.**	Meas.	Pers.	**Avg.**
Direct Input	21.2	32.5	32.8	27.7	47.8	32.2	31.0	18.5	24.9	22.8	30.1	31.2	30.7
Video-CoT [17]	23.9	35.7	34.7	29.6	50.5	35.1	33.2	21.1	26.0	24.5	32.5	34.6	33.5
Scene Graph Reconstruction [10]	25.4	37.5	36.4	31.5	56.7	38.5	36.7	25.2	29.1	27.4	35.2	37.3	36.2
APC [29]	25.0	40.8	36.0	30.4	59.1	40.1	40.8	27.1	31.7	29.4	38.8	40.1	39.4
SpatialMind	36.6	51.2	48.2	39.7	81.5	49.3	50.8	35.3	38.9	38.1	51.9	50.7	51.2
Ground Truth spatial memory	37.3	51.6	49.9	40.2	85.6	49.2	52.9	36.8	39.0	38.3	-	-	-

Table 2: Performance on NVidia Jetson Orin with InternVL3-8B. Task acronyms are defined in Section 6.

we exclude training-based methods [7, 16, 60] but compare against 5 training-free baselines, as described below:

- **Direct input** downsamples the video and directly feeds video frames to VLM for spatial reasoning.
- **Video-COT prompting [17]** focuses on video understanding and selects video frames via optimization.
- **Scene graph reconstruction [10]** models long videos with an entity–relation graph, where an LLM tracks entities and their interactions over time.
- **APC (Mental imagery simulation) [29]** is a framework where VLMs reason from arbitrary viewpoints, by converting a 2D image into a coarse 3D abstraction.

In addition, with VSI-Bench and STI-Bench benchmarks, we also evaluated the VLM performance using ground truth spatial memory, which is derived from the 3D point clouds and segmentation masks provided in datasets. Although still containing large errors due to the use of small on-device VLMs, this performance is considered as the best among using all types of spatial information input on embodied AI devices, and serves as an explicit validation of the spatial memory generated by SpatialMind; a smaller performance gap from the ground truth indicates higher accuracy and fewer errors in our constructed memory.

7 Performance Evaluation

7.1 Spatial Task Performance

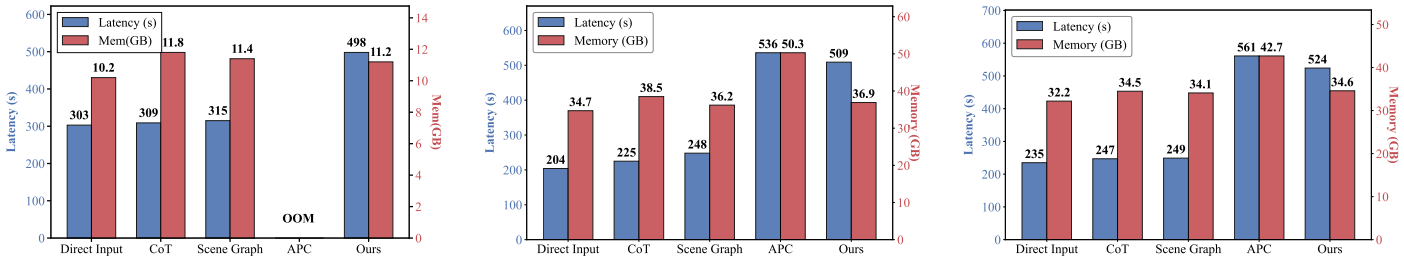
We conducted comprehensive evaluations of the performance of different spatial cognition tasks, over various embodied AI devices with the corresponding VLMs that can be deployed on these devices. Results in Tables 1-3 validated that SpatialMind largely enhances the spatial task performance with different VLMs, embodied AI devices and benchmark settings. In general, SpatialMind outperforms the baselines by 25% across different tasks. When larger VLMs are used in on-device embodied AI, the spatial task performance would be generally higher. With larger VLMs (e.g., Qwen3VL-32B-4Bit on NVidia Jetson Orin), the performance enhancement of SpatialMind can be up to 40%, especially in certain types of spatial tasks such as object counting (O.C.), absolute distance (A.D.) and dimensional measurements (D.Meas.).

Baselines like Video-CoT and Scene Graph Reconstruction show the minimal improvement. Due to the lack of explicit 3D information, their extended context windows often overwhelm the model's reasoning capabilities, hence degrading performance. On the other hand, although APC incorporates 3D data, it presents it frame-by-frame, but current VLMs struggle to integrate such fragmented information, hence resulting in only marginal performance gains. In contrast, SpatialMind excels by stitching multiview data into a coherent global spatial memory, hence driving significant improvements in tasks that involve multiple views.

Further, although a small performance gap remains compared to the ground-truth spatial memory, SpatialMind's superior results validate our core hypothesis: providing VLMs with an explicit and simplified spatial representation of 3D space is essential for achieving better performance in spatially aware embodied AI tasks. The remaining error is not caused by imperfect spatial-memory construction but the VLM's limited ability to interpret and reason over an explicit spatial representation. SpatialMind largely improves spatial reasoning over existing baselines, but it does not fully solve spatial cognition for lightweight VLMs.

Method	VSI-Bench							STI-Bench			Real-world		
	A.D.	O.S.	R.Dt.	R.Dr	O.C.	Ap.Or.	**Avg**	D.Meas.	Grd.	**Avg.**	Meas.	Pers.	**Avg.**
Direct Input	31.7	56.3	39.7	33.2	54.4	31.1	41.4	22.5	24.2	22.1	32.6	35.3	33.9
Video-CoT [17]	33.1	59.4	42.7	34.9	57.4	34.2	43.5	24.6	27.8	26.1	35.2	37.0	36.1
Scene Graph Reconstruction [10]	35.5	62.5	44.1	36.0	60.2	36.7	45.6	26.7	28.9	27.3	37.2	39.5	38.6
APC [29]	38.7	66.9	48.2	39.4	58.3	35.3	49.1	28.0	27.8	27.9	39.4	40.6	39.9
SpatialMind	53.2	79.3	64.4	51.8	82.3	45.0	67.1	36.4	33.9	34.8	49.2	54.0	52.6
Ground Truth spatial memory	55.2	80.5	65.1	53.6	87.4	45.2	68.3	37.5	34.4	35.9	-	-	-

Table 3: Performance on NVidia Jetson Orin with Qwen3VL-32B-4bit. Task acronyms are defined in Section 6.



(a) Qwen-2.5-VL-3B on Smartphone (b) Intern-VL3-8B on Jetson Orin (c) Qwen3VL-32B-4bit on Jetson Orin

Figure 14: Compute efficiency on different embodied AI devices

7.2 On-Device Compute Efficiency

Figure 14 presents the compute latency and memory cost of SpatialMind and baselines, with different VLMs being deployed on the corresponding embodied AI devices. SpatialMind demonstrates robust feasibility on these resource-constrained devices, where the most competitive baseline (i.e., APC) fail on the smartphone (Fig. 14(a)) due to insufficient memory. On the NVidia Jetson Orin (Figs. 14(b) and 14(c)), SpatialMind also outperforms APC in resource efficiency, reducing memory consumption by 26% and compute latency by 5%. SpatialMind does incur higher absolute cost than simpler direct-input and prompting baselines, reflecting the additional perception and multi-view alignment required to construct explicit spatial memory. Therefore, our current implementation targets spatial cognition on the local embodied AI device where privacy, connectivity, or local execution is important, rather than sub-second or high-frequency control. Further reducing the absolute compute latency remains important future system work.

7.3 Performance over Difference Levels of Scene Complexities

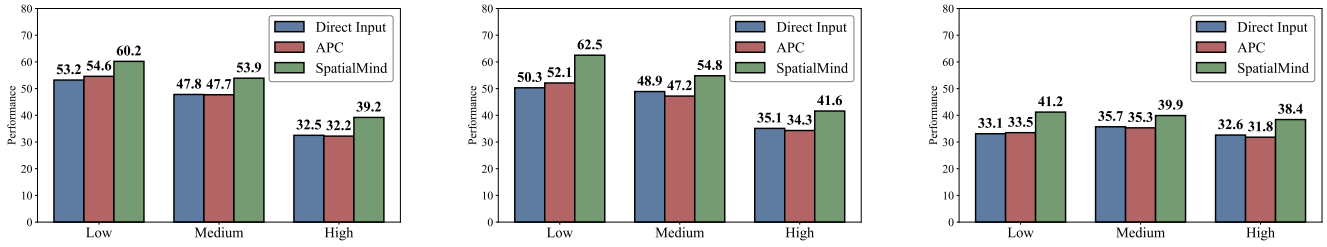
In this section, we analyze how the scene complexity affects the spatial task performance, using the InternVL3-8B model. We define the scene complexity along three aspects: (1) *object number* measured by the number of objects in the scene, (2) *spatial scale* measured by the room size, and (3) *temporal duration* measured by the length of video clip. We use the VSI-Bench benchmark which provides segmentation results of scenes, and compare SpatialMind with the most competitive baseline (APC) identified in Section 7.1.

Results in Figure 15 reveal that Object Number is the primary bottleneck for on-device embodied AI tasks, as the task accuracy drops sharply for all methods in cluttered scenes. However, SpatialMind still performs better than the baselines with 30% less performance drop. Crucially, SpatialMind shows generalizability to Spatial Scale, outperforming baselines in large environments where all the baselines struggle. This confirms that our spatial memory effectively integrates fragmented views, preserving spatial relationships in large space. Temporal Duration had the minimal impact, indicating that our key frame selection method is effective to retrieve necessary frames with different video lengths. Moreover, as shown in Figure 16, the computational cost of SpatialMind remains stable across different complexity levels.

7.4 Ablation Studies

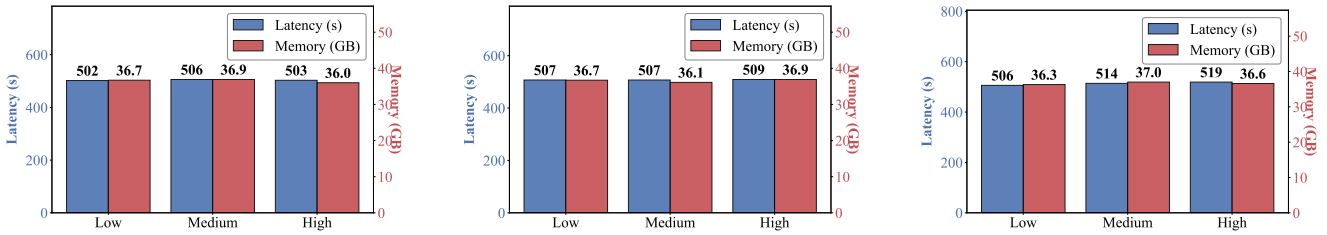
Different key frame selection strategies. To validate the effectiveness of our iterative temporal search for key frame seletion in Section 4, we compare it against two baselines: (1) processing all frames (at 1 FPS) and (2) uniform sampling, on NVidia Jetson Orin with Inverl-VL3-8B and Qwen3VL-32B-4bit using VSI-bench. As shown in Table 4, processing all frames results in Out-of-Memory; even in simulation, accuracy degrades due to the excessively long context window. Conversely, while uniform sampling is computationally feasible, it suffers from lower accuracy by missing critical, task-relevant frames. Our iterative approach achieve the optimal balance between accuracy and computational cost by concentrating resources on semantically significant frames.

Efficacy of different visual prompting methods. We further investigate the impact of different visual prompting



(a) Object Number (b) Spatial Scale (c) Temporal Duration

Figure 15: Spatial task performance with different levels of scene complexities



(a) Object Number (b) Spatial Scale (c) Temporal Duration

Figure 16: Compute efficiency of SpatialMind with different levels of scene complexities

Intern-VL3-8B	Task accuracy	Compute latency
All frames (1 FPS)	43.2%	OOM
Uniform sampling	46.0%	389 s
Ours (Iterative)	**58.7%**	501 s
Qwen3VL-32B-4bit	**Task accuracy**	**Compute latency**
All frames (1 FPS)	59.5%	OOM
Uniform sampling	62.0%	409 s
Ours (Iterative)	**73.3%**	524 s

Table 4: Ablation study on key frame selection methods

Intern-VL3-8B	Accuracy	Latency
Dense Points (BEV)	46.1%	172 s
Text Description	45.2%	169 s
Allocentric spatial memory + text	**58.7%**	172 s
Qwen3VL-32B-4bit	**Accuracy**	**Latency**
Dense Points (BEV)	61.3%	202 s
Text Description	61.0%	187 s
allocentric spatial memory + text	**73.3%**	201 s

Table 5: Ablation study on format of visual prompts

methods. Table 5 compares the prompting methods used in SpatialMind (allocentric spatial memory + Text) against two alternatives: projecting dense point clouds directly as visual inputs and using pure text-based coordinate descriptions. The experiments were conducted using InternVL3-8B and Qwen3VL-32B-4bit on the Jetson Orin with the VSI-Bench dataset. The results show that allocentric spatial memory is the most effective input format. It significantly outperforms text descriptions which VLMs struggle to interpret spatially, and dense point clouds which can introduce visual noise. Our method enhances reasoning accuracy without imposing a substantial latency overhead compared to simpler baselines.

7.5 Training-based Comparisons

In addition to inference-time baselines, we evaluate whether task-specific supervised fine-tuning can directly improve the spatial reasoning of the VLM backbone. We construct three training-based methods with the same training data and optimization budget: (1) **SFT-RGB**, which fine-tunes the VLM using the original RGB video observations; (2) **SFT-RGB+Depth**, which supplements RGB observations with estimated depth information; and (3) **SFT-SpatialMemory**, which fine-tunes the VLM using allocentric spatial memory reconstructed by SpatialMind. We fine-tuned InternVL3-8B with the VSI-Bench benchmark and test the fine-tuned model with our self-collected videos of real-world scenes.

As shown in Table 6, SFT-SpatialMemory achieves the best performance and further improves over the training-free methods. This supports our hypothesis that the main limitation of SpatialMind is not the correctness of constructed spatial memory, but VLM's capability to interpret and reason over it. SFT-RGB provides only a marginal improvement, as it is difficult for the model to learn spatial perception directly from raw RGB inputs. SFT-RGB+Depth achieves a larger improvement, but still underperforms SFT-SpatialMemory, likely because dense depth maps expose the model to excessive low-level geometric details rather than the compact high-level spatial layout provided by SpatialMind.

8 Related Work

Spatial memory and scene representations. Prior embodied systems represent environments using semantic maps, 3D

Method	Training	Accuracy (%)
Base VLM	No	30.7
SFT-RGB	Yes	33.9
SFT-RGB+Depth	Yes	41.7
SFT-SpatialMemory	Yes	56.3

Table 6: Comparison with training-based baselines

scene graphs, or hierarchical spatial memories [14, 19, 69, 79]. Many methods organize semantic information, such as what objects are present and how they are hierarchically related, but spatial relations still need to be inferred by VLM from RGB observations or constructed scene representation. In contrast, SpatialMind explicitly encodes object-level spatial relationships into a compact representation that is easier for lightweight on-device VLMs to interpret. Existing methods also often assume stronger sensing or reconstruction conditions. For example, [47] relies on RGB, depth, and odometry, while [70] assumes RGB-D input and reconstructs the 3D environment from all frames before selecting a subset for subsequent processing. SpatialMind instead targets resource-constrained devices with RGB-only input.

Efficient video analytics. Prior systems reduce video analysis cost by avoiding redundant frame processing. NoScope [27] accelerates video queries using specialized cascades, and Reducto [32] performs content-aware filtering to reduce the volume of video processed by downstream models. They demonstrate the effectiveness of exploiting temporal redundancy, but their objective differs from SpatialMind. SpatialMind handles open-vocabulary spatial queries that are not fixed in advance, and frame selection must satisfy two requirements: selected observations should contain task-relevant objects and retain sufficient visual overlap for cross-view geometric alignment. Hence, our iterative sampling strategy is jointly designed with spatial memory construction rather than serving as a generic frame-dropping method.

9 Discussions

Adaptability to downstream embodied control. While our evaluations formulate spatial cognition as natural language QA tasks, practical on-device embodied AI systems often require direct integration with physical actuation pipelines. In such practical deployments, task specifications may take the form of continuous control directives rather than conversational queries, and the VLM's outputs could be multimodal (bounding boxes or low-level robotic actuations) instead of text [80, 93]. SpatialMind easily adapts to these requirements. By modifying the lightweight VLM's output decoder, our explicitly constructed spatial memory can be mapped directly to structured coordinates or action primitives, allowing the framework to function as a highly efficient, end-to-end spatial controller without requiring a separate translation layer.

Scalability in large-scale environments. Our current experiments focus on small-scale indoor scenes, but SpatialMind can be adapted to large-scale environments with more objects and scene variability (e.g., warehouses or hospitals) . By offloading the spatial memory to local storage and leveraging advanced techniques of Retrieval-Augmented Generation (RAG) [5, 65, 73] and sparse activation [41, 54], we can fetch only the strictly task-relevant sub-regions of the map. This keeps the prompt context size small, ensuring that on-device VLM can execute complex and long-horizon queries.

Dynamic scenes and active exploration. SpatialMind assumes that scene geometry is static or changes slowly in the video buffer. This allows observations of the same object from different viewpoints to be aligned into a consistent allocentric spatial memory. Moving objects or substantial object relocation may violate these cross-frame correspondences and cause outdated or inconsistent locations in the memory. Handling such environments would require explicit dynamic-object tracking and continual spatial-memory updates, which are outside the scope of this work. Besides, SpatialMind also performs spatial reasoning only from observations already contained in the video buffer. If a task-relevant object has never been observed, the current system does not actively control the camera or embodied agent to search. Supporting this setting would require an active exploration policy, which jointly determines where to move and which new observations to acquire. We leave such dynamic-object tracking and active exploration to future work.

10 Conclusion

In this paper, we present SpatialMind, a new zero-shot inference framework that empowers small on-device VLMs with spatial cognition capabilities for spatially aware embodied AI applications. Rather than forcing lightweight VLMs to implicitly deduce 3D geometry from raw video pixels, SpatialMind explicitly convert visual inputs into a unified and sparse allocentric spatial memory. With resource-aware key frame selection and a topology-aware multi-frame alignment, our system incrementally populates this memory while isolating spatial drift and adhering to on-device compute budgets. Evaluations across diverse indoor environments demonstrate that SpatialMind improves the spatial task accuracy by up to 40% compared to competitive baselines.

Acknowledgment

We thank the anonymous reviewers and shepherd for their insightful comments and feedback. This work was supported in part by National Science Foundation (NSF) under grant number IIS-2205360 and CCF-2215042, and National Institutes of Health (NIH) under grant number R01HL170368 and R01HL184168.

References

[1] [n. d.]. Open Neural Network Exchange. https://github.com/onnx/onnx.

[2] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. 2025. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923* (2025).

[3] Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller. 2023. Zoedepth: Zero-shot transfer by combining relative and metric depth. *arXiv preprint arXiv:2302.12288* (2023).

[4] Aleksei Bochkovskii, AmaÃĢl Delaunoy, Hugo Germain, Marcel Santos, Yichao Zhou, Stephan R Richter, and Vladlen Koltun. 2024. Depth pro: Sharp monocular metric depth in less than a second. *arXiv preprint arXiv:2410.02073* (2024).

[5] Meghan Booker, Grayson Byrd, Bethany Kemp, Aurora Schmidt, and Corban Rivera. 2024. Embodiedrag: Dynamic 3d scene graph retrieval for efficient and scalable robot task planning. *arXiv preprint arXiv:2410.23968* (2024).

[6] Ellis Brown, Arijit Ray, Ranjay Krishna, Ross Girshick, Rob Fergus, and Saining Xie. 2025. Sims-v: Simulated instruction-tuning for spatial video understanding. *arXiv preprint arXiv:2511.04668* (2025).

[7] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. 2024. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14455–14465.

[8] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. 2024. Expanding Performance Boundaries of Open-Source Multimodal Models with Model, Data, and Test-Time Scaling. *arXiv preprint arXiv:2412.05271* (2024).

[9] Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. 2024. Yolo-world: Real-time open-vocabulary object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 16901–16911.

[10] Meng Chu, Yicong Li, and Tat-Seng Chua. 2025. Understanding long videos via llm-powered entity relation graphs. *arXiv preprint arXiv:2501.15953* (2025).

[11] Xiangxiang Chu, Limeng Qiao, Xinyu Zhang, Shuang Xu, Fei Wei, Yang Yang, Xiaofei Sun, Yiming Hu, Xinyang Lin, Bo Zhang, et al. 2024. Mobilevlm v2: Faster and stronger baseline for vision language model. *arXiv preprint arXiv:2402.03766* (2024).

[12] A Dai, M Nießner, M Zollhöfer, S Izadi, and C Theobalt. 2016. Bundlefusion: Real-time globally consistent 3d reconstruction using online surface re-integration. *arXiv preprint arXiv:1604.01093* (2016).

[13] Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A Smith, Wei-Chiu Ma, and Ranjay Krishna. 2024. Blink: Multimodal large language models can see but not perceive. In *European Conference on Computer Vision*. Springer, 148–166.

[14] Qiao Gu, Ali Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, et al. 2024. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 5021–5028.

[15] Xingyi He, Hao Yu, Sida Peng, Dongli Tan, Zehong Shen, Hujun Bao, and Xiaowei Zhou. 2025. Matchanything: Universal cross-modality image matching with large-scale pre-training. *arXiv preprint arXiv:2501.07556* (2025).

[16] Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 2023. 3d-llm: Injecting the 3d world into large language models. *Advances in Neural Information Processing Systems* 36 (2023), 20482–20494.

[17] Jian Hu, Zixu Cheng, Chenyang Si, Wei Li, and Shaogang Gong. 2025. Cos: Chain-of-shot prompting for long video understanding. *arXiv preprint arXiv:2502.06428* (2025).

[18] Kai Huang and Wei Gao. 2022. Real-time neural network inference on extremely weak devices: agile offloading with explainable AI. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*. 445–458.

[19] Kai Huang, Xiangyu Yin, Heng Huang, and Wei Gao. 2025. Modality Plug-and-Play: Runtime Modality Adaptation in LLM-Driven Autonomous Mobile Systems. In *Proceedings of the 31st Annual International Conference on Mobile Computing and Networking*.

[20] Xiaohu Huang, Jingjing Wu, Qunyi Xie, and Kai Han. 2025. MLLMs Need 3D-Aware Representation Supervision for Scene Understanding. *arXiv preprint arXiv:2506.01946* (2025).

[21] Mengdi Jia, Zekun Qi, Shaochen Zhang, Wenyao Zhang, Xinqiang Yu, Jiawei He, He Wang, and Li Yi. 2025. OmniSpatial: Towards Comprehensive Spatial Reasoning Benchmark for Vision Language Models. *arXiv preprint arXiv:2506.03135* (2025).

[22] Siyang Jiang, Mu Yuan, Xiang Ji, Bufang Yang, Zeyu Liu, Lilin Xu, Yang Li, Yuting He, Liran Dong, Wenrui Lu, et al. 2026. A large-scale multimodal dataset and benchmarks for human activity scene understanding and reasoning. In *Proceedings of the 24th Annual International Conference on Mobile Systems, Applications and Services*. 352–370.

[23] Jianbo Jiao, Ying Cao, Yibing Song, and Rynson Lau. 2018. Look deeper into depth: Monocular depth estimation with semantic booster and attention-driven loss. In *Proceedings of the European conference on computer vision (ECCV)*. 53–69.

[24] Philip N Johnson-Laird. 1980. Mental models in cognitive science. *Cognitive science* 4, 1 (1980), 71–115.

[25] Philip Nicholas Johnson-Laird. 1983. *Mental models: Towards a cognitive science of language, inference, and consciousness*. Number 6. Harvard University Press.

[26] Amita Kamath, Jack Hessel, and Kai-Wei Chang. 2023. What's" up" with vision-language models? investigating their struggle with spatial reasoning. *arXiv preprint arXiv:2310.19785* (2023).

[27] Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, and Matei Zaharia. 2017. Noscope: optimizing neural network queries over video at scale. *arXiv preprint arXiv:1703.02529* (2017).

[28] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*. 4015–4026.

[29] Phillip Y Lee, Jihyeon Je, Chanho Park, Mikaela Angelina Uy, Leonidas Guibas, and Minhyuk Sung. 2025. Perspective-aware reasoning in vision-language models via mental imagery simulation. *arXiv preprint arXiv:2504.17207* (2025).

[30] Xuanyu Lei, Zonghan Yang, Xinrui Chen, Peng Li, and Yang Liu. 2024. Scaffolding coordinates to promote vision-language coordination in large multi-modal models. *arXiv preprint arXiv:2402.12058* (2024).

[31] Xiang Li, Heqian Qiu, Lanxiao Wang, Hanwen Zhang, Chenghao Qi, Linfeng Han, Huiyu Xiong, and Hongliang Li. 2025. Challenges and Trends in Egocentric Vision: A Survey. *arXiv preprint arXiv:2503.15275* (2025).

[32] Yuanqi Li, Arthi Padmanabhan, Pengzhan Zhao, Yufei Wang, Guoqing Harry Xu, and Ravi Netravali. 2020. Reducto: On-camera filtering for resource-efficient real-time video analytics. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*. 359–376.

[33] Yun Li, Yiming Zhang, Tao Lin, XiangRui Liu, Wenxiao Cai, Zheng Liu, and Bo Zhao. 2025. Sti-bench: Are mllms ready for precise spatial-temporal world understanding? *arXiv preprint arXiv:2503.23765* (2025).

[34] Bo Liang, Chen Gong, Wei Gao, and Chenren Xu. 2026. Wave2Body: Rethinking mmWave Human Pose Estimation as Radar-to-Body Token Translation. *arXiv preprint arXiv:2607.18875* (2026).

[35] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer, 740–755.

[36] Xiongkun Linghu, Jiangyong Huang, Xuesong Niu, Xiaojian Shawn Ma, Baoxiong Jia, and Siyuan Huang. 2024. Multi-modal situated reasoning in 3d scenes. *Advances in Neural Information Processing Systems* 37 (2024), 140903–140936.

[37] Benlin Liu, Yuhao Dong, Yiqin Wang, Yongming Rao, Yansong Tang, Wei-Chiu Ma, and Ranjay Krishna. 2024. Coarse correspondence elicit 3d spacetime understanding in multimodal language model. *arXiv e-prints* (2024), arXiv–2408.

[38] Luyang Liu, Hongyu Li, and Marco Gruteser. 2019. Edge assisted real-time object detection for mobile augmented reality. In *The 25th annual international conference on mobile computing and networking*. 1–16.

[39] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. 2024. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European conference on computer vision*. Springer, 38–55.

[40] Weichen Liu, Qiyao Xue, Haoming Wang, Xiangyu Yin, Boyuan Yang, and Wei Gao. 2025. Spatial Reasoning in Multimodal Large Language Models: A Survey of Tasks, Benchmarks and Methods. *arXiv preprint arXiv:2511.15722* (2025).

[41] Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Re, et al. 2023. Deja vu: Contextual sparsity for efficient llms at inference time. In *International Conference on Machine Learning*. PMLR, 22137–22176.

[42] Yueen Ma, Zixing Song, Yuzheng Zhuang, Jianye Hao, and Irwin King. 2024. A survey on vision-language-action models for embodied ai. *arXiv preprint arXiv:2405.14093* (2024).

[43] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. 2017. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312* (2017).

[44] Anna Manasyan, Maximilian Seitzer, Filip Radovic, Georg Martius, and Andrii Zadaianchuk. 2025. Temporally consistent object-centric learning by contrasting slots. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 5401–5411.

[45] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

[46] Srija Mukhopadhyay, Abhishek Rajgaria, Prerana Khatiwada, Vivek Gupta, and Dan Roth. 2024. Mapwise: Evaluating vision-language models for advanced map queries. *arXiv preprint arXiv:2409.00255* (2024).

[47] Albert Gassol Puigjaner, Angelos Zacharia, and Kostas Alexis. 2026. Relationship-aware hierarchical 3d scene graph for task reasoning. *arXiv preprint arXiv:2602.02456* (2026).

[48] Pooyan Rahmanzadehgervi, Logan Bolton, Mohammad Reza Taesiri, and Anh Totti Nguyen. 2024. Vision language models are blind. In *Proceedings of the Asian Conference on Computer Vision*. 18–34.

[49] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. 2011. ORB: An efficient alternative to SIFT or SURF. In *2011 International conference on computer vision*. Ieee, 2564–2571.

[50] Mahadev Satyanarayanan. 2017. The emergence of edge computing. *Computer* 50, 1 (2017), 30–39.

[51] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. 2012. Indoor segmentation and support inference from rgbd images. In *European conference on computer vision*. Springer, 746–760.

[52] Nathaniel Simon and Anirudha Majumdar. 2023. Mononav: Mav navigation via monocular depth estimation and reconstruction. In *International Symposium on Experimental Robotics*. Springer, 415–426.

[53] Przemysław Skurowski, Dariusz Myszor, Marcin Paszkuta, Tomasz Moroń, and Krzysztof A Cyran. 2024. Energy demand in ar applications—a reverse ablation study of the hololens 2 device. *Energies* 17, 3 (2024), 553.

[54] Jifeng Song, Xiangyu Yin, Boyuan Yang, Kai Huang, Weichen Liu, and Wei Gao. 2026. Attribution-based Sparse Activation in Large Language Models. In *Proceedings of Machine Learning and Systems (MLSys)*, Vol. 8. 1685–1700.

[55] Xingzhe Song, Kai Huang, and Wei Gao. 2022. FaceListener: Recognizing human facial expressions via acoustic sensing on commodity headphones. In *Proceedings of the 2022 21st ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE/ACM, 145–157.

[56] Mark W Spong, Seth Hutchinson, Mathukumalli Vidyasagar, et al. 2006. *Robot modeling and control*. Vol. 3. Wiley New York.

[57] Ilias Stogiannidis, Steven McDonagh, and Sotirios A Tsaftaris. 2025. Mind the gap: Benchmarking spatial reasoning in vision-language models. *arXiv preprint arXiv:2503.19707* (2025).

[58] Jingwen Sun, Jing Wu, Ze Ji, and Yu-Kun Lai. 2024. A survey of object goal navigation. *IEEE Transactions on Automation Science and Engineering* 22 (2024), 2292–2308.

[59] Michael J Swain and Dana H Ballard. 1991. Color indexing. *International journal of computer vision* 7, 1 (1991), 11–32.

[60] Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai C Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. 2024. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *Advances in Neural Information Processing Systems* 37 (2024), 87310–87356.

[61] Fei Wang, Xingyu Fu, James Y Huang, Zekun Li, Qin Liu, Xiaogeng Liu, Mingyu Derek Ma, Nan Xu, Wenxuan Zhou, Kai Zhang, et al. 2024. Muirbench: A comprehensive benchmark for robust multi-image understanding. *arXiv preprint arXiv:2406.09411* (2024).

[62] Haoming Wang and Wei Gao. 2025. When Device Delays Meet Data Heterogeneity in Federated AIoT Applications. In *Proceedings of the 31st Annual International Conference on Mobile Computing and Networking*. 391–406.

[63] Haoming Wang and Wei Gao. 2026. Uncovering and Shaping the Latent Representation of 3D Scene Topology in Vision-Language Models. *arXiv preprint arXiv:2605.07148* (2026).

[64] Haoming Wang, Qiyao Xue, and Wei Gao. 2026. InfiniBench: Infinite Benchmarking for Visual Spatial Reasoning with Customizable Scene Complexity. *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2026).

[65] Haoming Wang, Boyuan Yang, Xiangyu Yin, and Wei Gao. 2025. Never Start from Scratch: Expediting On-Device LLM Personalization via Explainable Model Selection. In *Proceedings of the 23rd Annual International Conference on Mobile Systems, Applications and Services*.

[66] Xubin Wang, Zhiqing Tang, Jianxiong Guo, Tianhui Meng, Chenhao Wang, Tian Wang, and Weijia Jia. 2025. Empowering edge intelligence: A comprehensive survey on on-device ai models. *Comput. Surveys* 57, 9 (2025), 1–39.

[67] Xiaohan Wang, Yuhui Zhang, Orr Zohar, and Serena Yeung-Levy. 2024. Videoagent: Long-form video understanding with large language model as agent. In *European Conference on Computer Vision*. Springer, 58–76.

[68] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* 13, 4 (2004), 600–612.

[69] Abdelrhman Werby, Chenguang Huang, Martin Büchner, Abhinav Valada, and Wolfram Burgard. 2024. Hierarchical open-vocabulary 3d scene graphs for language-grounded robot navigation. In *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*.

[70] Abdelrhman Werby, Dennis Rotondi, Fabio Scaparro, and Kai O Arras. 2025. Keysg: Hierarchical keyframe-based 3d scene graphs. *arXiv preprint arXiv:2510.01049* (2025).

[71] Yue Wu, Zhaobo Qi, Junshu Sun, Yaowei Wang, Qingming Huang, and Shuhui Wang. 2025. Video Language Model Pretraining with Spatio-temporal Masking. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 8557–8567.

[72] Jie Xiao, Qianyi Huang, Xu Chen, and Chen Tian. 2024. Understanding Large Language Models in Your Pockets: Performance Study on COTS Mobile Devices. *arXiv preprint arXiv:2410.03613* (2024).

[73] Quanting Xie, So Yeon Min, Pengliang Ji, Yue Yang, Tianyi Zhang, Kedi Xu, Aarav Bajaj, Ruslan Salakhutdinov, Matthew Johnson-Roberson, and Yonatan Bisk. 2024. Embodied-rag: General non-parametric embodied memory for retrieval and generation. *arXiv preprint arXiv:2409.18313* (2024).

[74] Qiyao Xue, Weichen Liu, Shiqi Wang, Haoming Wang, Yuyang Wu, and Wei Gao. 2026. Reasoning path and latent state analysis for multi-view visual spatial reasoning: A cognitive science perspective. *European Conference on Computer Vision (ECCV)* (2026).

[75] Qiyao Xue, Xiangyu Yin, Boyuan Yang, and Wei Gao. 2025. PhyT2V: LLM-Guided Iterative Self-Refinement for Physics-Grounded Text-to-Video Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 18826–18836.

[76] Jihan Yang, Shusheng Yang, Anjali W Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. 2025. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 10632–10643.

[77] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. 2024. Depth anything v2. *Advances in Neural Information Processing Systems* 37 (2024), 21875–21911.

[78] Sihan Yang, Runsen Xu, Yiman Xie, Sizhe Yang, Mo Li, Jingli Lin, Chenming Zhu, Xiaochen Chen, Haodong Duan, Xiangyu Yue, et al. 2025. MMSI-Bench: A Benchmark for Multi-Image Spatial Intelligence. *arXiv preprint arXiv:2505.23764* (2025).

[79] Yuncong Yang, Han Yang, Jiachen Zhou, Peihao Chen, Hongxin Zhang, Yilun Du, and Chuang Gan. 2025. 3d-mem: 3d scene memory for embodied exploration and reasoning. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 17294–17303.

[80] Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Faisal Ahmed, Zicheng Liu, Yumao Lu, and Lijuan Wang. 2022. Unitab: Unifying text and box outputs for grounded vision-language modeling. In *European Conference on Computer Vision*. Springer, 521–539.

[81] Jinhui Ye, Zihan Wang, Haosen Sun, Keshigeyan Chandrasegaran, Zane Durante, Cristobal Eyzaguirre, Yonatan Bisk, Juan Carlos Niebles, Ehsan Adeli, Li Fei-Fei, et al. 2025. Re-thinking temporal search for long-form video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 8579–8591.

[82] Baiqiao Yin, Qineng Wang, Pingyue Zhang, Jianshu Zhang, Kangrui Wang, Zihan Wang, Jieyu Zhang, Keshigeyan Chandrasegaran, Han Liu, Ranjay Krishna, et al. 2025. Spatial Mental Modeling from Limited Views. *arXiv preprint arXiv:2506.21458* (2025).

[83] Xiangyu Yin, Boyuan Yang, Weichen Liu, Qiyao Xue, Abrar Alamri, Goeran Fiedler, and Wei Gao. 2025. ProGait: A Multi-Purpose Video Dataset and Benchmark for Transfemoral Prosthesis Users. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

[84] Yang You, Yixin Li, Congyue Deng, Yue Wang, and Leonidas Guibas. 2024. Multiview Equivariance Improves 3D Correspondence Understanding with Minimal Feature Finetuning. *arXiv preprint arXiv:2411.19458* (2024).

[85] Haoqi Yuan, Bohan Zhou, Yuhui Fu, and Zongqing Lu. 2024. Cross-embodiment dexterous grasping with reinforcement learning. *arXiv preprint arXiv:2410.02479* (2024).

[86] Mingxia Zhan, Li Zhang, Xiaomeng Chu, and Beibei Wang. 2025. VistaDepth: Frequency Modulation With Bias Reweighting For Enhanced Long-Range Depth Estimation. *arXiv preprint arXiv:2504.15095* (2025).

[87] Chaoning Zhang, Dongshen Han, Sheng Zheng, Jinwoo Choi, Tae-Ho Kim, and Choong Seon Hong. 2023. Mobilesamv2: Faster segment anything to everything. *arXiv preprint arXiv:2312.09579* (2023).

[88] Yuantong Zhang and Zhenzhong Chen. 2025. Continuous Space-Time Video Resampling with Invertible Motion Steganography. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2116–2126.

[89] Hong Zhao, Wei-Jie Wang, Tao Wang, Zhao-Bin Chang, and Xiang-Yan Zeng. 2019. Key-Frame Extraction Based on HSV Histogram and Adaptive Clustering. *Mathematical Problems in Engineering* 2019, 1 (2019), 5217961.

[90] Duo Zheng, Shijia Huang, Yanyang Li, and Liwei Wang. 2025. Learning from Videos for 3D World: Enhancing MLLMs with 3D Vision Geometry Priors. *arXiv preprint arXiv:2505.24625* (2025).

[91] Ying Zheng, Lei Yao, Yuejiao Su, Yi Zhang, Yi Wang, Sicheng Zhao, Yiyi Zhang, and Lap-Pui Chau. 2025. A survey of embodied learning for object-centric robotic manipulation. *Machine Intelligence Research* (2025), 1–39.

[92] Chenming Zhu, Tai Wang, Wenwei Zhang, Jiangmiao Pang, and Xihui Liu. 2024. Llava-3d: A simple yet effective pathway to empowering lmms with 3d-awareness. *arXiv preprint arXiv:2409.18125* (2024).

[93] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. 2023. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*. PMLR, 2165–2183.

[94] Andreas Zweng, Thomas Rittler, and Martin Kampel. 2011. Evaluation of histogram-based similarity functions for different color spaces. In *International Conference on Computer Analysis of Images and Patterns*. Springer, 455–462.

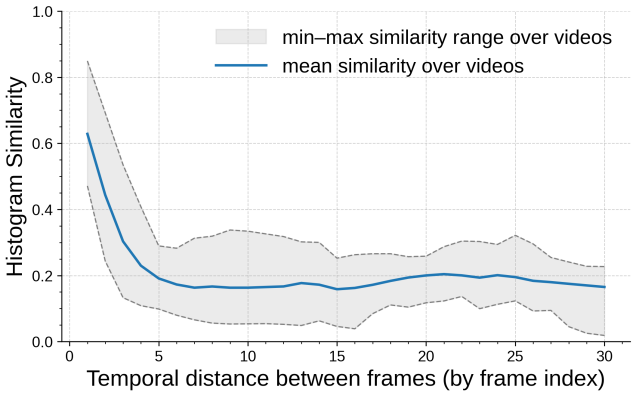


Figure 17: Frame similarity with different temporal distances between frames

A Temporal Locality among Video Frames

Our proposed approach of iterative temporal search for key frame selection in Section 3.2 builds on the temporal locality of object appearance among video frames. That is, if a frame is highly likely to contain an object, other frames that are temporally close could also have high likelihoods to contain the same object, due to the continuity of camera movement in the video.

To experimentally validate such locality, we randomly sampled 100 videos from the VSI-Bench benchmark that depict indoor scenes [76], randomly selected 10% frames from each video as reference frames, and computed the average histogram similarity [94] between each reference frame and its subsequent frames at different temporal distances. As shown in Figure 17, the frame similarity remains high in the first 2-3 frames, and then rapidly decays to a low level. This result suggests strong temporal locality within the range of 2-3 frames, which supports our approach of iterative temporal search, and is also used in Section 4.1 to decide the width of the Gaussian kernel to be applied when updating the sampling distribution over iterations.

B Representing the Spatial Memory to VLM

With the spatial memory, we still need to determine the most effective representation of the spatial memory as the input to VLM. Representing the spatial memory directly in structured text (e.g., JSON) has been proved to perform poorly on small VLMs [82], since the VLM has to internally translate numerical values into a visual representation, which is a shortcoming of most language models nowadays. For example, a statement such as "Object A is 2 meters to the north of Object B, facing east" is immediately apparent in a map, but would be verbose and less intuitive in JSON. This difficulty motivates us to instead adopt visual prompting techniques [30, 37] by plotting the semantic map as an image, as illustrated in Figure 18.

C System Parameters in Key Frame Selection

SpatialMind's key frame selection involves multiple system parameters, whose values and design choices are analyzed and discussed as below.

Utility score threshold (γ). A frame is selected as a candidate key frame when its utility score exceeds a predefined threshold. This threshold controls the trade-off between recall and precision in key frame selection. A lower threshold increases recall by including more candidate frames, but may introduce redundant observations. Conversely, a higher threshold improves precision but increases the risk of missing frames where task-relevant objects are partially occluded or detected with low confidence.

Method	Threshold	Precision	Recall	F_1
Uniform sampling	-	0.29	0.33	0.31
Retrieval-based	-	0.61	**0.79**	0.69
VideoAgent	-	0.58	0.66	0.63
Iterative search (ours)	1	0.55	0.78	0.67
Iterative search (ours)	**1.5**	0.65	0.76	**0.70**
Iterative search (ours)	2	**0.66**	0.67	0.66

Table 7: The impact of different values of the utility score threshold in key frame selection

Table 7 quantifies this trade-off using the LV-HAYSTACK dataset [81]. Across different threshold settings, our iterative temporal search approach achieves performance comparable to retrieval-based methods that score every frame exhaustively, while requiring substantially fewer frame evaluations. Compared with naive strategies such as uniform sampling or LLM-driven frame search [67], our method consistently achieves higher F_1 scores. In practice, a threshold value of 1.5 provides a balanced trade-off between recall and precision.

Number of temporal search iterations (B). Another key parameter is the number of temporal search iterations, given a fixed select batch m. Increasing this number allows the sampling distribution to be updated more frequently over more iterations, improving adaptivity to scene changes. However, a larger number also introduces higher computational overhead. Reducing this number lowers the cost of the selection process but may decrease the overall spatial coverage of the selected frames.

# of iterations	Precision	Recall	F_1	Sampled Frame (%)
100	0.57	0.66	0.63	10.2
33	0.62	0.71	0.66	25.6
20	**0.65**	**0.76**	**0.70**	26.1
14	0.65	0.72	0.68	30.7
11	0.63	0.70	0.67	37.2

Table 8: Performance of key frame selection with different numbers of iterations in temporal search

As shown in Table 8, a moderate number of 20 iterations achieves the best balance between frame selection quality and computational efficiency. Under this configuration, only approximately one quarter of the video frames need to be evaluated while maintaining strong precision and recall performance.

D More Samples of Real-World Indoor Scenes

More sample images of the real-world indoor scenes being involved in our evaluations are shown in Figure 19. These scenes exhibit substantial variation in spatial layout, object

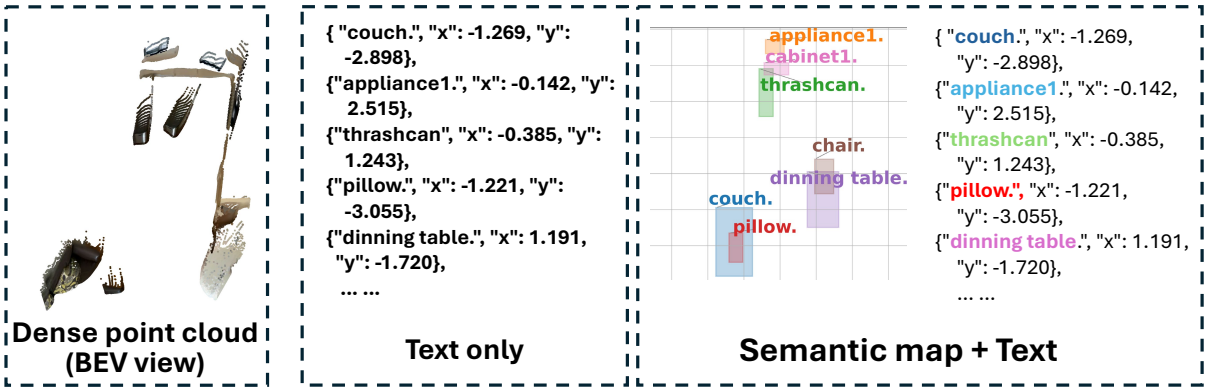


Figure 18: Different formats of space representation as inputs to VLMs

distribution, lighting conditions, and levels of human activity, providing rich visual and contextual cues for spatial understanding from egocentric video streams. They are also diverse in both structure and complexity, ranging from highly organized spaces such as restaurants to cluttered and dynamic environments such as supermarkets and museums. In addition, classroom and laboratory settings introduce semi-structured layouts with functional objects and equipment arranged for specific tasks. This diversity leads to challenging spatial configurations involving occlusions, long-range relationships, and viewpoint changes across frames. For each scene, we manually annotated 40 different spatial questions, yielding 200 spatial cognition tasks in total.

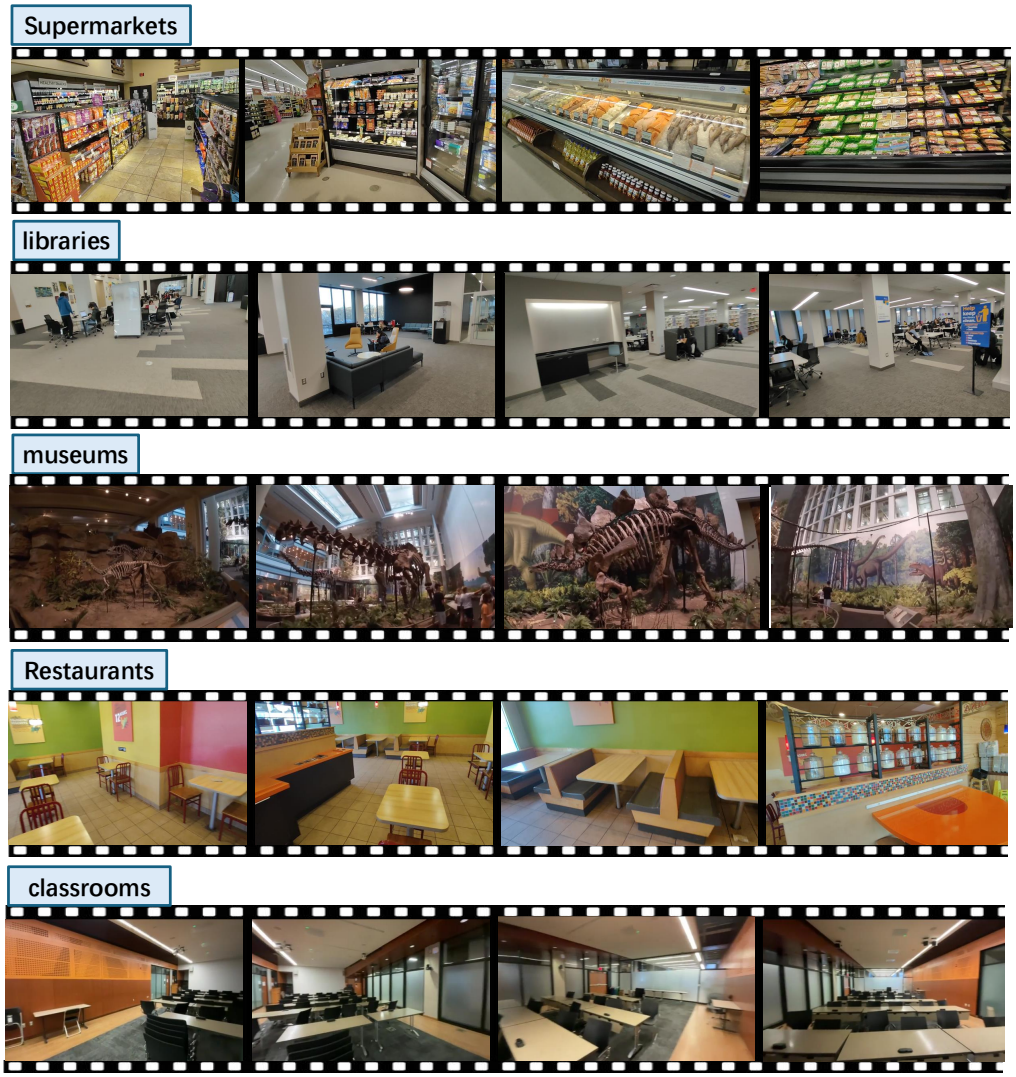


Figure 19: Data samples of real-world indoor scenes we used in evaluations