Language-Guided Semantic Map Navigator

Zehao Wang*[a], Mingxiao Li*[b], Minye Wu[a], Marie-Francine Moens[b], Tinne Tuytelaars[a]

[a] *ESAT, KU Leuven, Kasteelpark Arenberg 10, Leuven, 3001, Belgium*
[b] *Computer Science Department, KU Leuven, Celestijnenlaan 200a, Leuven, 3001, Belgium*

Abstract

The goal of vision-language navigation (VLN) is to enable an agent to comprehend human-like instructions and to execute them by finding their way through the environment. This task requires the integration of language understanding, visual perception, and decision-making capabilities. In this paper, we propose a novel approach to VLN that incorporates an additional pre-exploration stage, a setting that, although not common in the literature, we believe to be quite natural in many practical applications. Specifically, we propose our instruction-aware Path Proposal and Discrimination model (iPPD) to leverage pre-extracted 3D semantic information. iPPD generates **instruction-aware path proposals**, which are instruction-aligned candidate paths that help reduce the solution space. To better align modalities and represent map observations along a path, we propose a novel **Path Feature Encoding** scheme tailored for semantic maps. Furthermore, we design an attention-based **Language Driven Discriminator** to evaluate path candidates and select the best path as the final result. Compared to single-step greedy decision methods, our method reasons about the global path, thanks to the information extracted during pre-exploration, which naturally avoids error accumulation. Extensive experiments demonstrate that the effective use of global information from the pre-exploration stage can boost the performance of the language-guided navigation task with less training effort.

Keywords:
Vision Language Navigation, Semantic Map

* indicates an equal contribution

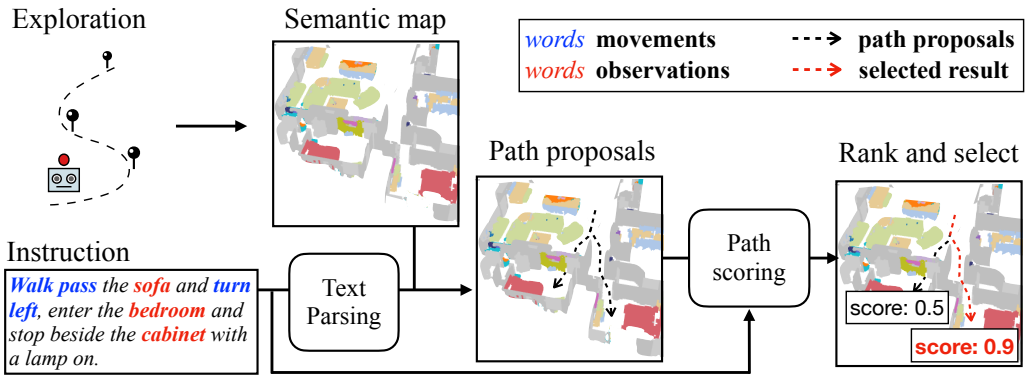

Figure 1: Overview of our proposed instruction-aware Path Proposal and Discrimination (iPPD) schema for the visual language navigation task with pre-exploration allowed. Semantic 3D metric maps are constructed during pre-exploration and are leveraged for instruction-constrained path planning.

1. INTRODUCTION

The general-purpose robot assistant of the future assists humans with daily tasks to reduce labor overhead, for instance, as a housekeeping or indoor service robot. An essential part of human-robot interaction involves *language guided navigation*, that is, enabling the robot to execute instructions given by a human to reach a target location. This requires the robot to interpret the natural language instruction, ground it in (usually visual) observations, and move accordingly.

The vision-and-language navigation (VLN) task [1] was first introduced in discrete environments, where navigable locations are predefined in a topological map known as a navigation graph. While many end-to-end deep learning-based algorithms have been shown to work well in this setting [2, 3, 4, 5], there is still a significant gap in deploying these agents directly in real-life scenarios due to the discrete environment assumption. To narrow this gap VLN in continuous environments was proposed [6]. However, end-to-end algorithms [6] designed for discrete environments do not perform satisfactorily in continuous environments, making this a significantly harder challenge.

Cross-Modal Transformer Planner (CMTP) [7] goes beyond previous research [2, 8] by breaking down navigation into two steps: map construction and planning. In their work, during map construction, agents pre-explore the whole environment and create a topological map. The agent then navigates step-by-step in the constructed map. Pre-exploration is a reasonable

assumption in many practical scenarios, such as service robots that usually work inside a fixed zone. However, we argue that the topological navigation map may be a suboptimal representation of the environment due to the loss of environmental information resulting from discretized operations. It is quite possible to miss the intersection points between a room and a corridor, which could be an important standpoint described in the language instructions. In addition, performing the subsequent path planning using only local information is short-sighted, and it easily leads to an erroneous path due to the error accumulation problem.

To tackle the aforementioned challenges, we argue that, if pre-exploration is permitted, planning a path on the constructed *global* semantic map is better than doing so only in the local area. To that end, we propose a novel modular solution for addressing the vision-language navigation task in a continuous environment. We named our solution **Instruction-aware Path Proposal and Discrimination (iPPD)**. Our solution follows a map construction, path proposing, and path scoring pipeline. We first introduce the 3D semantic map as a more informative representation of the environment. During pre-exploration, the agent takes multiple random paths in the environment and reprojects the egocentric observations to 3D point clouds. The 3D semantic metric map can be obtained by combining 2D egocentric semantic segmentation results constrained by 3D consistency. Subsequently, we perform path planning on the global 3D semantic metric map via a process consisting of path proposing and path scoring. Proposing random paths on the map could be simple, but it gets more complicated when taking constraints imposed by the language instruction into account. Inspired by the classical particle filter [9], we propose our instruction-constrained path-proposing algorithm, which yields instruction-aware candidate paths. The algorithm employs particles to simulate the movement of agents within the constructed map. The movements of all these particles are guided and constrained by the sparse action-object sequence extracted from the given instruction. Trajectories of particles are then taken as candidate paths, to be encoded and scored in the path-scoring stage. This initial stage of path proposing can be regarded as preliminary path planning, which is then followed by a subsequent stage of detailed path selection with our proposed transformer-based path scoring model. More specifically, we introduce a novel semantic feature extraction scheme tailored for the 3D semantic map to encode the environment's information associated with a point on a given path into feature representations. These extracted features of each path are then passed through a transformer-

based language-driven discriminator to obtain a path-specific score for each path. The highest-scoring candidate path is selected as the final solution.

In summary, our main contributions are three-fold:

- We introduce a high-resolution 3D semantic map as a more informative environment representation in a map-language navigation task.
- We propose a novel modular solution in a top-down manner, which accomplishes language-guided navigation tasks by proposing paths, path embedding, and scoring based on 3D and language context.
- Our approach is the first method based on global path planning on 3D semantic maps for language-guided navigation. Compared to traditional vision-based navigation methods, our approach demonstrates advantages and showcases potential applications in certain scenarios.

2. RELATED WORK

Vision-and-Language Navigation. In the VLN [1] task, an agent needs to navigate to a goal location in a photorealistic virtual environment following a given natural language instruction. Research on the VLN task has made significant progress in the past few years. Attention mechanisms across different modalities are widely used to learn the alignment between vision and language, hence boosting the performance [6, 2, 10, 3, 5, 11, 12, 13]. The improvements of VLN also come from new learning approaches, such as imitation learning (IL) [4] and reinforcement learning (RL) [2, 14]. Another set of work, such as VLNBERT-CE [15], [16] and [17], focuses on leveraging existing, well-studied discrete environment solutions. These approaches typically discretize the action space into topological graphs and encode panoramic views at each point to represent the surrounding semantics. At the same time, another line of work focuses on exploiting an actual map of the environment in the navigation task [18, 19, 20, 21, 22]. Previous works follow the pipeline of first building a 2D map using RGB and depth images on the fly, and then integrate the built map into the model as additional visual input. The 2D map inevitably drops some important environmental information; for example, when two objects are stacked at one location, only one will be on the map. Moreover, prior works only treated maps as an auxiliary modality, there is a lack of study on the map modality itself. In

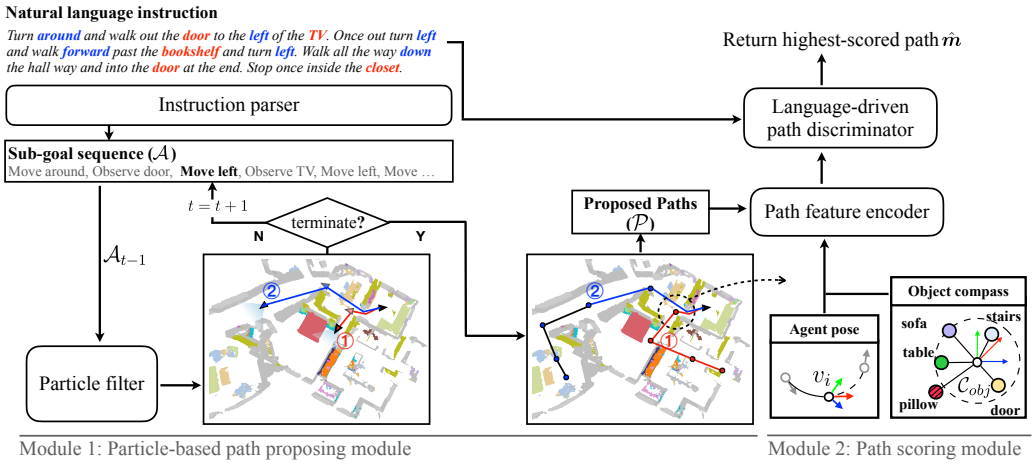

Figure 2: **Language-guided path planning model (iPPD)**: The architecture consists of two modules. In the particle-based path proposing module, the language instruction is processed by GPT3.5 and leveraged to step the particle filter. Particles follow the extracted actions with a certain amount of uncertainty to sample the next navigable location and are weighted by observations. The movement of two particles is marked as trajectory 1 and trajectory 2 with red and blue lines, respectively. All valid paths will be encoded and scored by the path-scoring module. The highest-scored path $\hat{m}$ is returned as the model decision.

this work, we introduce a more informative 3D semantic map and propose a novel solution to conduct path planning directly on the map.

Multi-modal Transformer The transformer model has achieved great success in natural language processing and vision-language tasks. [23] first pre-train the BERT model on large-scale text data and achieve state-of-the-art performance on a wide range of natural language processing (NLP) tasks. Inspired by the great success of BERT on NLP, a lot of research extends the transformer model to process both vision and language information. [23, 24] propose a two-stream BERT model to first separately encode texts and images and then fuse the two modalities via a cross-modal attention layer. Another line of work introduces single-stream multimodal BERT, which directly processes both vision and language information simultaneously using cross-modal attention layers [25, 26, 27]. Different from previous works which mainly focus on studying transformer architectures that handle vision and language data, in this work we further exploit the transformer model and extend it to process language and 3D map information.

3. OVERVIEW

We begin by establishing our notation and settings. We follow the setting of [7], where pre-exploring the environment is allowed. Our pipeline has two stages: semantic map reconstruction (Sec. 4.1) and language-guided path planning (Sec. 4.2, Sec. 4.3, Sec. 4.4).

In the semantic map reconstruction stage, an agent takes random walks in each scene. The collected RGBD observation at each time step t with camera pose p_t is segmented and aggregated into a global 3D semantic map $\mathbf{M}$ on the fly as shown in Fig. 3.

Our language-guided path planning model iPPD can be divided into two main modules. The first module proposes candidate paths $\mathcal{P} = \{\mathbf{p}_1, \cdots, \mathbf{p}_n\}$ given a natural language instruction T and the start agent pose v_0. In particular, the path proposals are constrained based on the sub-goal sequence $\mathcal{A}$ extracted from T (further explained in Sec. 4.2). Then, we use the second module F_{score} to encode, score, and select the best-ranked path as the final prediction. We formulate it as:

$$\hat{\mathbf{p}} = \underset{\mathbf{p}_i \in \mathcal{P}}{\arg\max}\, F_{\text{score}}(\mathbf{M}, T, \mathcal{P}) \tag{1}$$

The main components of stage two are demonstrated in Figure 2. In real applications, the final best-ranked path can be optimized into an executable trajectory by a kinodynamic solver. In the Habitat simulator [28], this process can be simplified and managed by the greedy geodesic planner. We go through the details of each individual component in the following section.

4. METHOD

4.1. Semantic metric map reconstruction

We build 3D semantic maps $\mathbf{M}$, as shown in Figure 3 with a voxel-based representation. The resolution is set to 0.1 meters. At each time step t, the wandering agent receives an RGB observation I_t and a depth observation D_t. We use a pre-trained segmentation model Mask2Former [29] to perform semantic segmentation on I_t. The segmentation results describe the label distribution of each pixel in the egocentric view. Each pixel can be projected to an egocentric point cloud with the help of D_t and camera intrinsics K. Together with the camera pose v_t, we can obtain the reprojection transformation $\mathbf{T}_w$ from pixel space to world coordinates. We transfer each pixel-wise

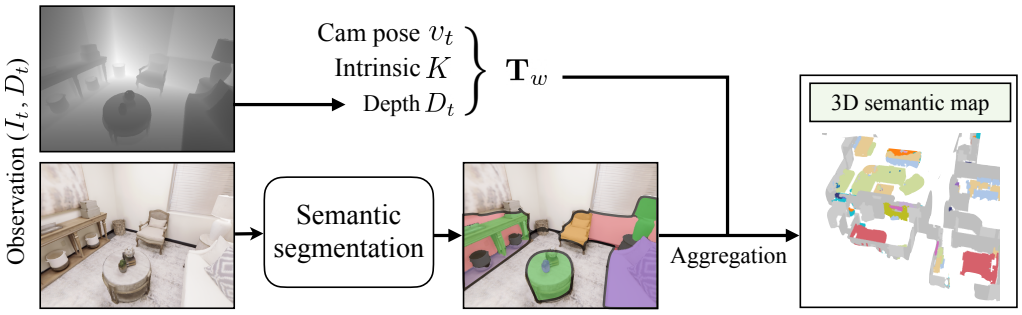

Figure 3: **Semantic map reconstruction:** We construct a 3D semantic map while exploring the environment. At each time step, the agent processes RGBD observations and dynamically updates the map content.

semantic prediction to its corresponding voxel and accumulate the predictions. The semantic map is dynamically updated during exploration. To tolerate the error of semantic prediction, we threshold out those voxels that have fewer than 5 observations and max-pool the results accumulated at each voxel to obtain the final semantic map $\mathbf{M}$. In real-world applications, camera poses can be estimated using SLAM [30], but since map reconstruction is not our main focus, we leverage ground truth camera poses returned from the Habitat [28] simulator.

4.2. Particle-Based Path Proposals

The goal of language-guided navigation is to enable an agent to follow given instructions and navigate to a specified location. Crucially, actions such as turning left, turning right, and moving forward serve as critical cues for determining the appropriate trajectory. Meanwhile, key objects observed along the path can serve as a constraints to further reduce the solution space. However, due to the intricate and ambiguous nature of natural language, precise extraction of action and key object information from the given instructions can be challenging. This implies that if one wants to use language instruction as a guide for trajectory proposing, it is necessary to deal with its uncertainty. In response to this challenge, we design an instruction-constrained random walk-based path proposing strategy, drawing inspiration from the particle filter algorithm [9]. Our algorithm aims to generate a set of potential paths $\mathcal{P} = \{\mathbf{p}_1, \ ..., \mathbf{p}_n\}$ by utilizing a sparse sub-goal sequence $\mathcal{A}$.

Sub-goal sequence extraction by LLM As previously elucidated, the actions and objects mentioned in the instruction serve as critical indicators

Algorithm 1: Particle Filter Step

Input: $\mathcal{X}_{t-1}$, $\mathcal{A}_{t-1}$ `// last particle states, sub-goal`
Output: $\mathcal{X}_t$ `// estimated particles states`

$\mathbf{A}_{trigger}, \mathbf{A}_{content} = \mathcal{A}_{t-1}$
$\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
foreach x_{t-1} **in** $\mathcal{X}_{t-1}$ **do**
  if $\mathbf{A}_{trigger}$ **is** MOVE **then**
    $x_t = \text{MOVE}(x_{t-1}, \mathbf{A}_{content})$
    $w_t = 1$
  else if $\mathbf{A}_{trigger}$ **is** OBSERVE **then**
    $w_t = \text{OBSERVE}(x_{t-1}, \mathbf{A}_{content})$
    $x_t = x_{t-1}$
  $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t, w_t \rangle$
$P_t = \textbf{KDE}(\bar{\mathcal{X}}_t)$
resample $\mathcal{X}_t$ from P_t

for finding the correct path. We define the movement and observation descriptions as sub-goals $\mathcal{A}$ of instruction T. In our algorithm, the extraction of sub-goals $\mathcal{A}$ from a given instruction T is accomplished through prompting a *Large Language Model* (LLM). Specifically, we embed the current instruction T into a predefined prompt and input it into LLM. The LLM constructs and returns the sub-goal sequence in the required format as described in the prompt. We use GPT3.5-turbo with manually defined prompts. An example is listed in the appendix. The output from the LLM contains two kinds of sub-goals, **OBSERVE([object])** and **MOVE([direction])**. These will serve as the observation model and motion model in the particle filter.

Proposing paths by particle filter The path proposing strategy follows the standard procedure of the particle filter [9]. Our algorithm is illustrated in Algorithm 1. We initialize N particles at the starting position. Then the particles will follow the sub-goal sequence $\mathcal{A}$. The main components of the algorithm are defined as follows:

1. **MOVE** sub-goal will direct all the particles using the manually designed motion models. For instance, **MOVE(right)** will trigger each particle to turn a random angle between 30° to 150° clockwise. Details of other motion models can be found in the appendix.

2. **OBSERVE** sub-goal will trigger the observation model to assign an importance weight to each particle. The importance is measured by distance to the closest *object* mentioned in the current sub-goal within a radius of five meters.
3. **Resampling** is the procedure to estimate the sampling distribution and resample particles to the most certain area. The sampling distribution is estimated from all weighted particles by weighted Gaussian Kernel Density Estimation (KDE). Each surviving particle contains its history waypoints which form the trajectory.[1]

After executing the sub-goal sequence $\mathcal{A}$ with a particle filter, we obtain a sequence of waypoints $\mathbf{m}_i = \{m_0, ..., m_n\}$ for each surviving particle in $\mathcal{X}_t$. The navigable path $\mathbf{p}_i$ is then generated by connecting each subsequent waypoint using A* algorithm on the map. For simplicity, we use the same symbol $\boldsymbol{v} = \{v_0, ..., v_n\}$ to denote each candidate path $\mathbf{p}_i$ (a set of agent poses) in the later section. In the case that we cannot find any navigable waypoints given one instruction, possibly because no actions can be extracted, we choose to draw multiple random paths as alternatives.

It is worth noting that we do not apply collision detection while moving the particle. The reason for this is that an action sequence is usually very sparse in VLN tasks. If we avoid obstacle-crossing at the proposing stage, candidate paths tend to be located in a very local region, which limits the diversity of path proposals. Experiments discussing the effects of including obstacle avoidance can be found in section 5.8.

4.3. Path Feature Encoding

Next, we propose a path feature encoding scheme and apply it to each candidate path in the semantic map $\mathbf{M}$. Ideally, the path feature $\mathbf{\Gamma}$ represents the environmental context along the path and can be aligned with the language representation to verify if this path matches the given instruction. Since nearby positions may have a large overlap when observed, iPPD discretizes each path by a fixed distance. It then encodes the local context of each position along the path to form a feature sequence in temporal order. Specifically, for each position feature, besides the agent pose at this position,

[1] In the actual implementation, we performed an equivalent operation for better efficiency. We chose to initialize a large number of particles $N = 10,000$ at the start position and directly drop the particles with zero weights during the execution of the algorithm.

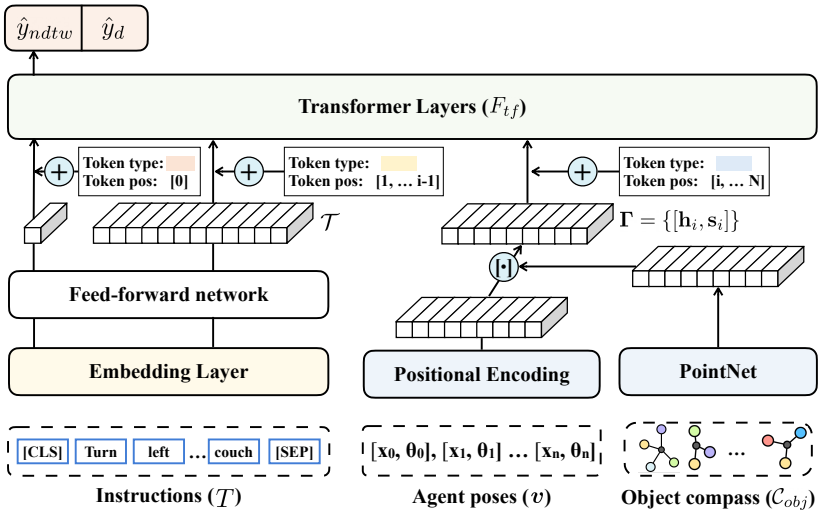

Figure 4: **Path scoring model.** The model scores a path with the concatenation of the $[CLS]$ special token, instruction token sequence, agent poses, and object compasses as input.

we further introduce an object compass to perceive the local environment, as illustrated in Figure 2.

The object compass $\mathcal{C}_{\text{obj}} = \{(x_i, y_i, z_i, \mathbf{w}_i)\}$ contains semantic voxels near the agent within 5 meters radius. We record voxel positions (x_i, y_i, z_i) in the egocentric coordinate system centered on the agent. The object compass is treated as a 3D point cloud in a 3D egocentric view. For each point in the cloud, we embed the voxel's semantic classes with the GLOVE embedding [31] $\mathbf{w}_i$ based on its distinguishability. To obtain a permutation-invariant representation of the points' order, we use PointNet [32] to encode the object compass:

$$\mathbf{h} = \text{PointNet}(\mathcal{C}_{\text{obj}}), \tag{2}$$

where $\mathbf{h} \in \mathbb{R}^d$ is the d-dimensional feature of the local object's context at a given position. In practice, due to GPU memory restrictions and to increase the diversity of observations, at each position, we randomly sample 200 observed points within the compass range each time the model processes the data during training.

We also incorporate low-level information, specifically, agent poses $\boldsymbol{v} = \{v_0, ..., v_n\}$, into the feature sequence. An agent pose feature $\mathbf{s}$ aggregates

location $\mathbf{x}$ and orientation θ information at v_i as follows:

$$\mathbf{s} = \text{FFN}_{ap}([PE(\mathbf{x}), PE(\theta)]), \tag{3}$$

where $[\cdot]$ represents concatenation, $PE(\cdot)$ is the agent pose positional encoding function [33], and FFN_{ap} is a single-layer feed-forward network projecting the agent position into the d-dimensional feature space. By gathering features from all locations $\boldsymbol{v}$, we obtain the path feature $\mathbf{\Gamma} = \{[\mathbf{h}_i, \mathbf{s}_i]\}$.

4.4. Path scoring and ranking

We deploy a transformer-based model [34] as a *Language-driven Discriminator* F_{tf} for scoring paths. The natural language instruction T is tokenized and embedded using CLIP [35] embeddings $\psi(\cdot)$. We project the word embeddings to feature vectors using a single-layer feed-forward network $\text{FFN}_{lang}(\cdot)$.

$$\mathcal{T} = \text{FFN}_{lang}(\psi(T)), \tag{4}$$

where $\mathcal{T} \in \mathbb{R}^{N \times d}$ is the word sequence representation of length N and dimension d.

The F_{tf} has 12 transformer layers. It takes a concatenated sequence of $\mathcal{T}$ and $\mathbf{\Gamma}$ as input and predicts a score for the corresponding path. A learnable special token embedding $[CLS]$ is placed at the front of the input sequence to aggregate context information. The *Language-driven Discriminator* is formulated as

$$\hat{y}_{ndtw},\ \hat{y}_d = F_{tf}([[\text{CLS}], \mathcal{T}, \mathbf{\Gamma}]) \tag{5}$$

where $\hat{y}_{ndtw}$ and $\hat{y}_d$ are two scores projected from the $[CLS]$ token embedding. Specifically, $\hat{y}_{ndtw}$ is the predicted nDTW score and indicates how well the candidate path shape matches the ground truth path, and $\hat{y}_d$ is the predicted distance to the goal rearranged to [0, 1], which measures how well the agent stops at the correct location. During inference, we combine the two scores with a weighted term $\lambda = 0.5$ as a uniform metric for path ranking.

$$\hat{y} = \lambda \cdot \hat{y}_{ndtw} + (1 - \lambda) \cdot \hat{y}_d \tag{6}$$

The highest-scoring path is picked as the predicted path $\hat{\mathbf{p}}$.

In addition, we apply the Masked Language Prediction task [23] to enhance the model's context awareness. We randomly mask 15% of the tokens

Model	Explore	Local view	Map	Val_Seen				Val_Unseen			
				nDTW ↑	OS ↑	SR ↑	SPL ↑	nDTW ↑	OS ↑	SR ↑	SPL ↑
Seq2Seq	✗	✓	no map	0.49	0.40	0.30	0.28	0.53	0.39	0.27	0.25
CMA	✗	✓	no map	0.46	0.38	0.28	0.26	0.56	0.38	0.33	0.31
Waypoint	✗	✓	no map	-	0.51	0.44	0.42	-	0.38	0.34	0.32
VLNBERT-CE	✗	✓	topological	0.58	0.59	0.50	0.44	0.55	0.53	0.44	0.39
Dreamwalker	✗	✓	topological	-	0.66	0.59	0.48	-	0.59	0.49	0.44
ETPNav	✗	✓	topological	-	0.67	0.59	0.56	-	0.57	0.50	0.46
CMTP	✓	✓	topological	-	0.45	0.36	0.31	-	0.38	0.26	0.22
CM2	✗	✗	semantic	-	0.50	0.42	0.34	-	0.41	0.34	0.27
WS-MGMap	✗	✗	semantic	-	0.51	0.46	0.43	-	0.47	0.38	0.17
iPPD	✓	✗	semantic	0.66	0.59	0.51	0.48	0.63	0.52	0.42	0.39
iPPD + sp	✓	✗	semantic	0.67	0.66	0.57	0.54	0.64	0.58	0.45	0.42

Table 1: **Results**: Performance of models under different setups (input modalities, w/ or w/o pre-exploration) evaluated on VLN-CE dataset.

in the instruction and train the model to predict the masked tokens during training. The total training loss $\mathcal{L}_{\text{total}}$ is formulated as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{MLM}} + \mathcal{L}_{\text{MSE}}^{(ndtw)} + \mathcal{L}_{\text{MSE}}^{(ed)}, \tag{7}$$

where $\mathcal{L}_{\text{MLM}}$ is the Masked Language Prediction loss, and $\mathcal{L}_{\text{MSE}}$ is the mean squared error between predicted scores and ground truth (nDTW and distance to goal).

5. Experiments

5.1. Dataset

We follow the pre-exploration setting of CMTP [7] applied on the VLN-CE dataset [6] with the original dataset splits for training and evaluation. The environments in the validation seen set have been observed during the training phase, but the instructions are not. In contrast, both environment and instruction have not been observed during training for episodes in the validation unseen split.

5.2. Implementation Details

Semantic Map Construction. We use Mask2Former [29] Swin-L (IN21k) pre-trained on MS-COCO [36] as our basic semantic segmentation module. Semantic labels are manually mapped and filtered to commonly appearing objects in indoor scenes. We adopt a simple random walk strategy to pre-explore the environment. We used an average of 200 agents to perform

random walks in parallel in each environment, with a maximum 1000 steps. This is a CPU-intensive job but it only needs to explore once the static environment. More efficient policies such as [20], [37] could be used for this purpose. Please refer to the supplementary materials for details.

Model Setup. The main learnable component is the path-scoring module which contains a PointNet for observation encoding and a transformer model for score prediction. We optimize the joint model using AdamW [38] optimizer with a learning rate of 1e-4 and train it on a single NVIDIA-P100 GPU.

5.3. Evaluation Metrics

We evaluate results using the standard evaluation metrics in visual navigation and visual-language navigation tasks [39, 1, 40]: normalized dynamic-time warping (nDTW), oracle success (OS) which measures the percentage of predicted trajectories that pass the target point, success rate (SR), and success weighted by the normalized inverse of the path length (SPL). We choose nDTW and SR as our primary metrics, as they cover two important aspects of the navigation task: (a) predicted and ground truth path similarity, and (b) accuracy of reaching the target location.

5.4. Main Results

Table 1 exhibits a comparative analysis of our proposed method with other models. We first review the methods developed with different input modalities and settings for the VLN-CE task. The first three models [41, 6] use RGBD observations as input and predict only the next action at each time step. Recently, constructed maps have gradually been considered an important modality while doing language-guided navigation. A branch of research, including works like VLNBERT-CE [15], [16] and [17], focuses on discretizing the action space through waypoint prediction. This approach allows the method to leverage existing, well-studied solutions in the original R2R discrete setting, such as panorama-enhanced topological map encoding and large-scale pretraining. These methods implicitly encode the representation of semantic information within each node and remain the state-of-the-art across various settings. The CMTP [7] model first introduces the pre-exploration phase and topological map reconstruction in the VLN-CE task. The reduced action space make it easier for the agent to decide its next moving position. CM2 [42] and WS-MGMap [43] include metric semantic maps as an auxiliary modality and put more focus on how to leverage dense

2D semantic metric local maps at each time step to determine a short-term trajectory. The models within the RGBD category rely on a deep learning algorithm to implicitly learn both semantic and geometric information of the environment from visual input, while models in the Map category exploit the benefit of maps to preserve explicit spatial information and conduct planning on the map.

We follow the mapping and planning roadmap [7] and take one step further arguing that when pre-exploration is permitted, the algorithm should maximize the potential benefits of the constructed 3D semantic metric map. It is natural to perform path planning on the constructed map. We present the results of our model in two different path-proposing settings in the last two rows denoted as **iPPD** and **iPPD+sp**. (1) **iPPD** is the standard setting including path proposing and scoring stages as described in the previous section. As shown in Table 1, our extended setting achieves remarkable results, especially compared to CMTP [7] which has the most similar setting. Extensive use of the map constructed in the pre-exploration phase yields a large performance gain. This result implies that the semantic map has a great potential in language-guided navigation tasks. (2) In the **iPPD+sp** setting we ignore the intermediate waypoints and only consider the endpoints of the path proposals. We then construct the trajectory as the shortest path between the start agent location and endpoint. We observe that instead of using the full path proposed by our path-proposing algorithm, only taking the end point of the proposed path results in better performance in the VLN-CE dataset. This result is underlined in Table 1. Despite exhibiting superior performance, it heavily depends on the assumption of the shortest path, which could potentially align with the annotation bias within the VLN-CE dataset.

5.5. Does the constructed map contain enough semantic information of the environment?

Due to errors in the 2D semantic segmentation, constructed semantic map inevitably contains noise. We argue that, on the one hand, our voting-based map construction strategy reduces multi-view semantic inconsistency, especially for common object categories such as "table" and "chair". On the other hand, our random sampling strategy for observations at each camera pose v_i introduces robustness to handle the noisy observations. In this experiment, we compare our model trained with the constructed semantic maps to one that uses ground truth maps. The ground truth semantic maps are

Model	Val-Seen		Val-Unseen	
	nDTW ↑	SR ↑	nDTW ↑	SR ↑
iPPD	0.66	0.51	0.63	0.42
w/ sp	0.67	0.57	0.64	0.45
w/ gt map	0.68	0.53	0.63	0.41
w/ gt map + sp	0.68	0.56	0.64	0.45

Table 2: Constructed semantic map v.s. ground truth map: navigation performance trained on maps with different qualities.

constructed from the human-annotated semantic mesh provided by Matter-Port3D [44] dataset. As shown in Table 2, the ground truth map shows a 2% better success rate in the validation seen environment without the shortest path prior. For other settings, using our constructed maps achieves comparable performance to that of using the ground truth semantic map. This does not mean that the overall quality of our constructed maps is as good as ground truth maps, but the semantic information which can be extracted and leveraged by our path-scoring module seems to be sufficient. From the visualization of both semantic maps in Figure. 5, we can observe that our reconstructed map can handle objects that have relatively high segmentation accuracy well, such as tables and beds. Affiliated objects such as pillows or small objects with fewer observations, such as bedside lamps are sometimes missed.

5.6. How effective is each component in iPPD ?

To verify the effectiveness of each component of our proposed iPPD, we conduct ablation studies as shown in Table 3. Agent pose information addresses the agent's state in the environment. This information is vital, as we can see from the second row of Table 3. Models trained without agent pose information show a drop of 11% in SR on seen environments and a 9% drop on unseen environments. Next, we verify the effectiveness of the encoded path features using our proposed object compass. The results in the last row of Table 3 show the importance of point-wise context information. Removing the object compass from the model results in a significant drop in all evaluation metrics; the SR on seen environments and unseen environments decreases by 24% and 12% respectively.

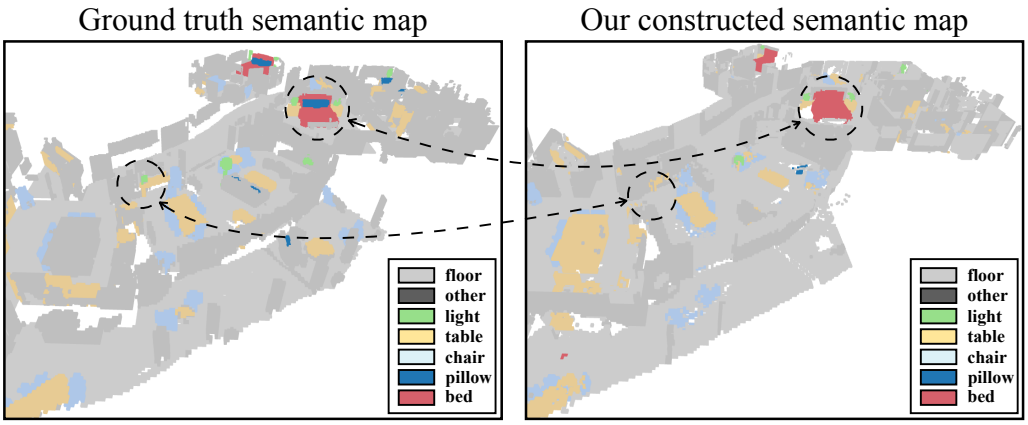

Figure 5: Comparison between our constructed map and ground truth semantic map with a subset of semantic labels rendered.

Model	Val-Seen		Val-Unseen	
	nDTW ↑	SR ↑	nDTW ↑	SR ↑
iPPD	0.66	0.51	0.63	0.42
w/o agent pose	0.62	0.40	0.57	0.33
w/o obj compass	0.54	0.27	0.56	0.30

Table 3: Ablation study for iPPD method on both the val_seen and val_unseen data splits of VLN-CE dataset.

5.7. How the object compass radius influence the performance?

In this section, we conduct an ablation study to examine the impact of the object compass radius on performance. As shown in Table 4, a radius of five meters provides the best balance between the diversity of semantic information and the distinguishability of each local region. Notably, there is a significant 3% increase in the success rate when comparing the radius of five meters to seven meters on the unseen split.

5.8. What is the influence of different trajectory-proposing strategies?

In this section, we investigate the impact of different trajectory-proposing strategies. Our particle-based path-proposing algorithm relies on the extracted sub-goal sequence. An effective path proposal algorithm should have the capability to propose candidate paths, including at least one path that

Radius	Val-Seen		Val-Unseen	
	nDTW ↑	SR ↑	nDTW ↑	SR ↑
1m	0.32	0.24	0.30	0.19
3m	0.61	0.45	0.59	0.37
5m	**0.66**	**0.51**	**0.63**	**0.42**
7m	0.63	0.50	0.61	0.39
9m	0.59	0.45	0.56	0.36

Table 4: Performance evaluation of various object compass radius using the iPPD method on both the val_seen and val_unseen data splits of the VLN-CE dataset.

can reach a nearby location of the ground truth target. Furthermore, the trajectory shape should be close to the human-annotated trajectory following the instruction. Therefore, we evaluate various path-proposing strategies using the **Recall** metric, which quantifies the proportion of data that has candidate paths with endpoints within one meter of the goal position. We also use the **average nDTW (anDTW)** metric, which evaluates the average similarity between recalled paths and the ground truth path. Additionally, we visualize the proposed paths of different trajectories to provide a visual explanation, complementing the above metrics.

We evaluate three strategies with a fixed number of initial particles, i.e., $10,000$: (a) Random walk (**random**): The simplest action space. The particle can move in any direction and cover any distance at each time step. In this experiment, we bound the maximum moving distance of each time step to 10 meters. The maximum number of action steps is limited to five to better fit the size of indoor scenes; (b) Proposed setting (**proposed**): The proposed setting described in the previous section; (c) Proposed setting with obstacle avoidance (**+ obstacle**): Particle movement is not allowed if there is any obstacle between two sampled waypoints.

The random walk strategy ignores the information in the instruction. While the algorithm may suggest potential routes, it is apparent that the resulting trajectory is chaotic and difficult to conform to the instruction-guided ground truth path, as depicted in Figure 6. Table 5 illustrates that the incorporation of the obstacle avoidance strategy detrimentally impacts the Recall metric. When obstacles are considered during path generation, particles are easily blocked, as presented in Figure 6. This results in a large

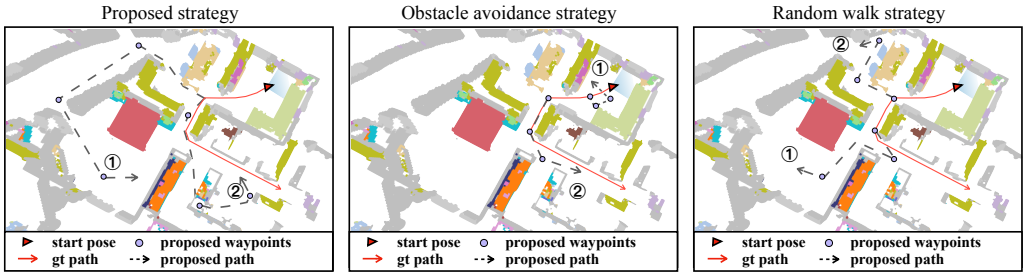

Figure 6: Visualization of classical cases for three different path-proposing strategies. Purple dots are proposed waypoints obtained from different strategies, they are later connected by A* shortest path planning in order. The demoed action sequence is [*turn around, turn left, turn left, turn left*]. Our default path proposing strategy (**proposed strategy**) can better recall the ground truth path with acceptable sampling efficiency. Obstacle avoidance strategy always failed to generate sufficient long paths due to the sparsity of directly mentioned actions in the instruction. The random walk cannot guarantee the instruction constraints and is likely to generate zig-zag trajectories.

Strategy	**Val-Seen**		**Val-Unseen**	
	Recall	anDTW	Recall	anDTW
proposed	73.65	0.695	70.96	0.726
+ obstacle	48.84	0.792	43.01	0.806
random	62.85	0.708	54.65	0.727

Table 5: Comparison of different path proposing strategies evaluated by the recall of ground truth goal location and average nDTW

percentage of short local paths, such as trajectory 1. Consequently, there is a substantial decrease in the number of proposed paths, leading to a performance drop. However, with a sufficient number of particles, this strategy has the potential to find a more accurate trajectory set. As shown in Table 5, the average nDTW of the method **proposed + obstacle** is significantly higher than that of the other two methods. Due to the balance of sampling efficiency and recall, we choose to use our current **proposed** strategy but also acknowledge the benefits of including obstacle avoidance for future research.

5.9. Time complexity

The training time for our fully supervised semantic-map-based solution is significantly more efficient than that of other methods. For example, the

classic Waypoint [41] method requires 5 days using 64 GPUs with reinforcement learning, while the more recent ETPNav [17] method, which employs a discretized action space and imitation learning, still takes 1 day with 2 GPUs. In contrast, our method only takes 12 hours with a single GPU, facilitating quick deployment of the agent in a known environment.

Regarding inference time, our solution completes the task in just 10 minutes for the unseen split of VLNCE dataset (i.e., 0.3 seconds per instruction). Other methods, which rely on sequential prediction, take much longer. Even the simplest Seq2Seq [6] method takes four times longer than our solution.

6. Discussion

In this section, we discuss the potential avenues for future research and the limitations of our method. With our experimental results, we show that, under the situation where pre-exploration is allowed, the VLN task could be better addressed using the pipeline of global map construction and path planning. Despite the significant performance gains achieved by our proposed method, there remains a substantial discrepancy between its performance and that of humans. Therefore, it would be valuable to further investigate better ways to encode visual information about the environment or design a more effective and efficient instruction-guided local or global map path planning algorithm. The limitations of our method are as follows:

1. The reconstructed semantic maps only contain rough semantic information. These rough metric maps lose the instance-level information, which makes it impossible to execute instructions containing specific objects. How to include more of the instance-level information and even their attributes in the map is an interesting future research direction. Likewise, the information we extract from the language instruction is sparse and can be suboptimal. How to include recent advancements in LLM to supplement possible intermediate steps is worth further study.
2. Since the path-proposing strategy is not accurate enough, we still need to rely on a trained scoring model to rank the paths. It is possible to include an advanced probability method with a more accurate subgoal extraction strategy to achieve good performance in the zero-shot setting. We expect to work in this direction in future work.

7. Conclusion

In this paper, we explore a new avenue for addressing the difficult vision-and-language navigation task in the continuous environment. Following the previous work [7], we allow the agent to establish a memory of the environment by pre-exploration. We then argue that when pre-exploration is permitted, the agent should maximize the benefits of using the constructed memory. To this end, we propose a particle-based path proposal algorithm and a cross-model transformer path scorer to conduct path planning on the constructed map. Experimental results illustrate that our proposed solution can boost the performance of the VLN task.

Declaration of Generative AI in Scientific Writing

During the preparation of this paper, we utilized GPT-4 and Grammarly to enhance the clarity and natural flow of the introduction section and the experiment section.

Acknowledgement

This research received funding from the Flanders AI Impuls Programme - FLAIR and from the European Research Council Advanced Grant 788506.

Appendix A. Instruction parsing

We use GPT3.5-turbo as the instruction parser. We construct the prompts with three examples as follows:

```
[{
  "role": "system",
  "content": "You are a helpful, pattern-following assistant that
     parses human instruction to a sequence of sub-goals. Each
     sub-goal can only be one of the following patterns:
    OBSERVE([object])
    MOVE([direction])"
},
{
  "role": "system", "name":"example_user",
  "content": "Exit the bedroom and turn left. Walk straight passing
     the gray couch and stop near the rug and a table."
```

```
},
{
  "role": "system", "name": "example_assistant",
  "content": "
    OBSERVE: bedroom
    MOVE: forward
    MOVE: left
    MOVE: forward
    OBSERVE: couch
    MOVE: forward
    OBSERVE: rug
    OBSERVE: table"
},
{
  "role": "system", "name":"example_user",
  "content": "Go straight past the pool. Walk between the bar and
      chairs. Stop when you get to the corner of the bar. That's
      where you will wait."
},
{
  "role": "system", "name": "example_assistant",
  "content": "
    MOVE: forward
    OBSERVE: pool
    MOVE: forward
    OBSERVE: bar
    OBSERVE: chairs
    MOVE: forward
    OBSERVE: bar"
},
{
  "role": "system", "name":"example_user",
  "content": "Go pass the white table, turn right at the second
      door, move pass the piano and stop"
},
{
  "role": "system", "name": "example_assistant",
  "content": "
    MOVE: forward
    OBSERVE: table
```

```
    MOVE: right
    OBSERVE: door
    MOVE: forward
    OBSERVE: piano
    MOVE: forward"
},
{
  "role": "user", "content": ""
}]
```

We conduct a human evaluation on the performance of our instruction parser's extraction quality. We invite three students to evaluate whether the extraction results cover the major components of the instruction. Each student is assigned 200 instructions and their parsing results. If the extracted key components cover 90% or more of the components in the instruction, the parsing result is scored 5. If it covers 20% or less, the parsing result is scored 1. The score distribution is shown in Table A.6. More than 90% of the instructions cover at least 80% of the key components. This indicates relatively high coverage, but since there are still instances where key components might be missed, our random trajectory samples can help mitigate these issues.

Score	5	4	3	2	1
Ratio	48.68%	42.31%	8.79%	0.14%	0.08%

Table A.6: Human evaluation of LLM parsing performance

Appendix B. Details of Motion models

We set the following standard actions:

1. **Move forward**: Moving the particle along the current orientation for a distance uniformly sampled from 0 to 10 meters. 0 distance is set to tolerate excessive extraction. For instance, "you can see a sofa on the left" is easily misidentified as the action "turn left". With the 0 distance, the particle has a probability of staying in its current state. Furthermore, we add Gaussian noise with distribution $\mathcal{N}(0, 1)$ to the current orientation to handle probable missed turning action.

2. **Move left/right**: The turning actions are defined as turning a certain degree followed by a moving forward action. Since the turning angle will not be specified in the language, we set a uniform distribution with the range 30° to 150° for turning left/right.
3. **Move around**: Turning around is usually mentioned at the beginning of the instruction. We set a turning direction of 180° together with Gaussian noise $\mathcal{N}(0, 1)$.

Appendix C. Details of observation model

To query the description of observations extracted from instruction on the map cannot simply based on string matching since similar meanings can have various forms of expression. We extract CLIP [35] features of both semantic map labels and extracted descriptions and calculate the cosine similarities between them. The highest-scored semantic label is treated as the observed semantics at this time stamp.

The observation model will take the observed semantics to query the map region within five meters of the particles. The weight assigned is the inverse distance towards the queried semantics.

References

[1] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, A. Van Den Hengel, Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 3674–3683.

[2] X. Wang, Q. Huang, A. Celikyilmaz, J. Gao, D. Shen, Y.-F. Wang, W. Y. Wang, L. Zhang, Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6629–6638.

[3] F. Landi, L. Baraldi, M. Cornia, M. Corsini, R. Cucchiara, Multimodal attention networks for low-level vision-and-language navigation, Computer Vision and Image Understanding 210 (2021) 103255.

[4] X. Wang, W. Xiong, H. Wang, W. Y. Wang, Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 37–53.

[5] C.-Y. Ma, J. Lu, Z. Wu, G. AlRegib, Z. Kira, R. Socher, C. Xiong, Self-monitoring navigation agent via auxiliary progress estimation, in: Proceedings of the International Conference on Learning Representations (ICLR), 2019.
URL https://arxiv.org/abs/1901.03035

[6] J. Krantz, E. Wijmans, A. Majumdar, D. Batra, S. Lee, Beyond the nav-graph: Vision-and-language navigation in continuous environments, in: European Conference on Computer Vision, Springer, 2020, pp. 104–120.

[7] K. Chen, J. K. Chen, J. Chuang, M. Vázquez, S. Savarese, Topological planning with transformers for vision-and-language navigation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 11276–11286.

[8] F. Zhu, Y. Zhu, X. Chang, X. Liang, Vision-language navigation with self-supervised auxiliary reasoning tasks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 10012–10022.

[9] P. Del Moral, Nonlinear filtering: Interacting particle resolution, Comptes Rendus de l'Académie des Sciences-Series I-Mathematics 325 (6) (1997) 653–658.

[10] H. Huang, V. Jain, H. Mehta, J. Baldridge, E. Ie, Multi-modal discriminative model for vision-and-language navigation, in: Proceedings of the Combined Workshop on Spatial Language Understanding (SpLU) and Grounded Communication for Robotics (RoboNLP), 2019, pp. 40–49.

[11] C.-Y. Ma, Z. Wu, G. AlRegib, C. Xiong, Z. Kira, The regretful agent: Heuristic-aided navigation through progress estimation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6732–6740.

[12] F. Landi, L. Baraldi, M. Corsini, R. Cucchiara, Embodied vision-and-language navigation with dynamic convolutional filters, arXiv preprint arXiv:1907.02985 (2019).

[13] Y. Hong, C. Rodriguez, Q. Wu, S. Gould, Sub-instruction aware vision-and-language navigation, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020, pp. 3360–3376.

[14] E. Wijmans, A. Kadian, A. Morcos, S. Lee, I. Essa, D. Parikh, M. Savva, D. Batra, Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames, arXiv preprint arXiv:1911.00357 (2019).

[15] Y. Hong, Z. Wang, Q. Wu, S. Gould, Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 15439–15449.

[16] H. Wang, W. Liang, L. Van Gool, W. Wang, Dreamwalker: Mental planning for continuous vision-language navigation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 10873–10883.

[17] D. An, H. Wang, W. Wang, Z. Wang, Y. Huang, K. He, L. Wang, Etpnav: Evolving topological planning for vision-language navigation in continuous environments, IEEE Transactions on Pattern Analysis and Machine Intelligence (2024).

[18] S. Gupta, D. Fouhey, S. Levine, J. Malik, Unifying map and landmark based representations for visual navigation, arXiv preprint arXiv:1712.08125 (2017).

[19] Z. Seymour, K. Thopalli, N. Mithun, H.-P. Chiu, S. Samarasekera, R. Kumar, Maast: Map attention with semantic transformers for efficient visual navigation, in: 2021 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2021, pp. 13223–13230.

[20] D. S. Chaplot, D. Gandhi, S. Gupta, A. Gupta, R. Salakhutdinov, Learning to explore using active neural slam, in: International Conference on Learning Representations (ICLR), 2020.

[21] T. Chen, S. Gupta, A. Gupta, Learning exploration policies for navigation, in: International Conference on Learning Representations, 2019.
URL https://openreview.net/pdf?id=SyMWn05F7

[22] S. Tan, M. Ge, D. Guo, H. Liu, F. Sun, Self-supervised 3d semantic representation learning for vision-and-language navigation, arXiv preprint arXiv:2201.10788 (2022).

[23] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2019, pp. 4171–4186.

[24] H. Tan, M. Bansal, Lxmert: Learning cross-modality encoder representations from transformers, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019, pp. 5100–5111.

[25] Y.-C. Chen, L. Li, L. Yu, A. El Kholy, F. Ahmed, Z. Gan, Y. Cheng, J. Liu, Uniter: Universal image-text representation learning, in: European conference on computer vision, Springer, 2020, pp. 104–120.

[26] G. Li, N. Duan, Y. Fang, M. Gong, D. Jiang, Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 11336–11344.

[27] L. H. Li, M. Yatskar, D. Yin, C.-J. Hsieh, K.-W. Chang, What does BERT with vision look at?, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Online, 2020, pp. 5265–5275. doi:10.18653/v1/2020.acl-main.469.
URL https://aclanthology.org/2020.acl-main.469

[28] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. Chaplot, O. Maksymets, A. Gokaslan, V. Vondrus, S. Dharur, F. Meier, W. Galuba, A. Chang, Z. Kira, V. Koltun, J. Malik, M. Savva, D. Batra, Habitat 2.0: Training home

assistants to rearrange their habitat, in: Advances in Neural Information Processing Systems (NeurIPS), 2021.

[29] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, R. Girdhar, Masked-attention mask transformer for universal image segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 1290–1299.

[30] H. Durrant-Whyte, T. Bailey, Simultaneous localization and mapping: part i, IEEE robotics & automation magazine 13 (2) (2006) 99–110.

[31] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation, in: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.

[32] C. R. Qi, H. Su, K. Mo, L. J. Guibas, Pointnet: Deep learning on point sets for 3d classification and segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 652–660.

[33] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, R. Ng, Nerf: Representing scenes as neural radiance fields for view synthesis, in: European conference on computer vision, Springer, 2020, pp. 405–421.

[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[35] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, I. Sutskever, Learning transferable visual models from natural language supervision, in: M. Meila, T. Zhang (Eds.), Proceedings of the 38th International Conference on Machine Learning, Vol. 139 of Proceedings of Machine Learning Research, PMLR, 2021, pp. 8748–8763.
URL `https://proceedings.mlr.press/v139/radford21a.html`

[36] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft coco: Common objects in context, in: Computer Vision–ECCV 2014: 13th European Conference, Zurich,

Switzerland, September 6-12, 2014, Proceedings, Part V 13, Springer, 2014, pp. 740–755.

[37] D. S. Chaplot, H. Jiang, S. Gupta, A. Gupta, Semantic curiosity for active visual learning, in: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16, Springer, 2020, pp. 309–326.

[38] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, in: International Conference on Learning Representations (ICLR), 2019.

[39] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, et al., On evaluation of embodied navigation agents, arXiv preprint arXiv:1807.06757 (2018).

[40] G. Magalhaes, V. Jain, A. Ku, E. Ie, J. Baldridge, Effective and general evaluation for instruction conditioned navigation using dynamic time warping, arXiv preprint arXiv:1907.05446 (2019).

[41] J. Krantz, A. Gokaslan, D. Batra, S. Lee, O. Maksymets, Waypoint models for instruction-guided navigation in continuous environments, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 15162–15171.

[42] G. Georgakis, K. Schmeckpeper, K. Wanchoo, S. Dan, E. Miltsakaki, D. Roth, K. Daniilidis, Cross-modal map learning for vision and language navigation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 15460–15470.

[43] P. Chen, D. Ji, K. Lin, R. Zeng, T. H. Li, M. Tan, C. Gan, Weakly-supervised multi-granularity map learning for vision-and-language navigation, in: Advances in Neural Information Processing Systems, 2022.

[44] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, Y. Zhang, Matterport3D: Learning from RGB-D data in indoor environments, International Conference on 3D Vision (3DV) (2017).