Appendix for
"FGPrompt: Fine-grained Goal Prompting for Image-goal Navigation"

In the appendix, we provide more implementation details and experimental results of our FGPrompt. We organize the appendix as follows.

- In Sec. A, we provide more architecture details on three different types of goal-prompting methods.
- In Sec. B, we provide more experimental details, *i.e.*, training and evaluation settings.
- In Sec. C, we provide more ablation results on the skip fusion mechanism.
- In Sec. D, we provide more ablation results on the mid fusion mechanism.
- In Sec. E, we provide a direct comparison with the memory-based methods.
- In Sec. F, we provide additional visualization results.

A Architecture details

Skip fusion mechanism. As discussed in the previous sections, we design a skip fusion mechanism that utilizes keypoint detection and matching method to provide the agent with low-level prompting. In order to increase the training speed, we abandon the handcrafted local feature descriptor and detector, instead of a deep learning-based keypoint matching pipeline [2, 10], as shown in Figure 1. A convolution-based keypoint detector is adopted to extract keypoints from both the goal image and observation image. After that, a 9-layer graph neural network with attention modules is applied to find the paired points that share similar features.

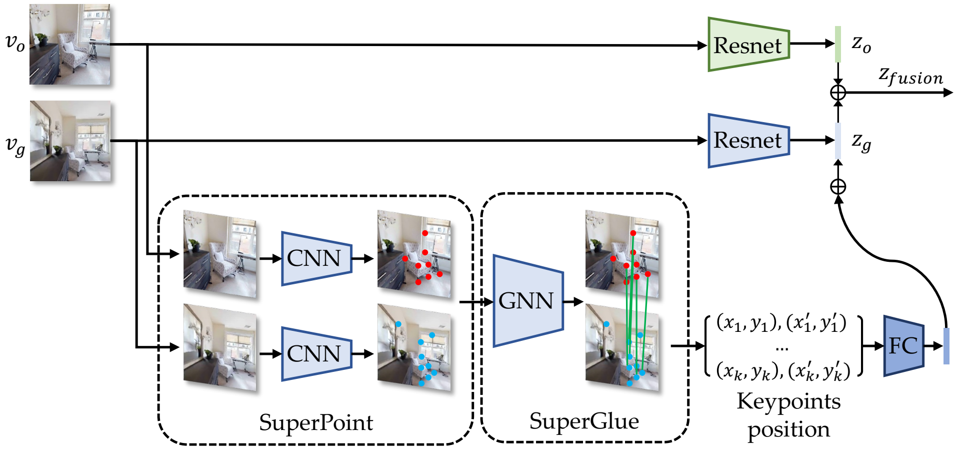


Figure 1: Architecture details of the skip fusion mechanism.

Mid fusion mechanism. We illustrate the detailed architecture of the proposed mid fusion mechanism in Figure 2. For a Resnet9 backbone, we map the intermediate activation maps of each resnet block (ResBlock) into affine factor β and γ using a 1×1 convolution and a fully connected layer. Then the β and γ are injected into the correspondent ResBlock in the observation encoder, guiding the model to focus on goal-relevant regions in the observation.

Early fusion mechanism. The early fusion mechanism is initialized as a single Resnet9 [12] encoder. The stem convolution layer takes a $128 \times 128 \times 6$ image as input. The following layers have no difference from a standard Resnet encoder. We conduct experiments using both Resnet9 and Resnet50 encoders.

Navigation policy. We initialize the navigation policy network as a 2-layer GRU with an embedding size of 128.

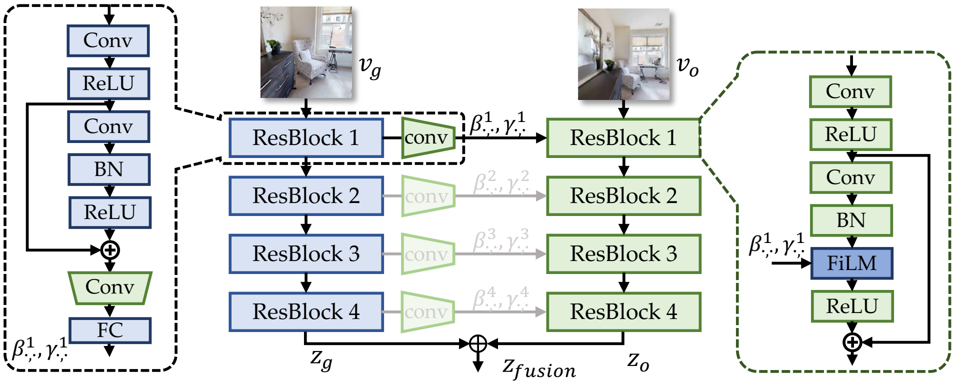

Conv
ReLU
Conv
BN
ReLU
Conv
FC
$\beta^1_{\cdot,\cdot}, \gamma^1_{\cdot,\cdot}$
v_g
v_o
ResBlock 1
conv
$\beta^1_{\cdot,\cdot}, \gamma^1_{\cdot,\cdot}$
ResBlock 1
ResBlock 2
conv
$\beta^2_{\cdot,\cdot}, \gamma^2_{\cdot,\cdot}$
ResBlock 2
ResBlock 3
conv
$\beta^3_{\cdot,\cdot}, \gamma^3_{\cdot,\cdot}$
ResBlock 3
ResBlock 4
conv
$\beta^4_{\cdot,\cdot}, \gamma^4_{\cdot,\cdot}$
ResBlock 4
z_g
z_{fusion}
z_o
Conv
ReLU
Conv
BN
$\beta^1_{\cdot,\cdot}, \gamma^1_{\cdot,\cdot}$
FiLM
ReLU

Figure 2: Architecture details of the mid fusion mechanism.

B Experimental details

Dataset details. We train our agent on the Gibson dataset and validate the agent on the Gibson, MP3D, HM3D datasets respectively. For the training dataset, there are 72 scenes in total, each scene has 9k episodes, resulting in 648k episodes. The 9k episodes in each scene are evenly divided into three levels according to the distance from the start location to the goal location: easy (1.5 - 3m), medium (3 - 5m), and hard (5 - 10m). For evalution on Gibson, we use two split, in which split A [8] has 14 scenes with 1.4k episodes per level and split B [3] has 14 scenes with 1k episodes per level. For evalution on MP3D and HM3d, we use the same test splits as [12], which has 100 scenes with 1k episodes per level and 18 scenes with 1k episodes per level respectively.

Training & validation details. We use the Habitat simulator to train our model on the Gibson dataset using 20 environments running in parallel with 8×3090 GPUs. We set the total training time steps to 500M. For one episode, we set the maximum time steps to 500 when performing validation. Other detailed hyperparameters of DD-PPO training follow the recipe of ZER [12].

Reward We use the reward formulation proposed by [12] that consists of three parts, including dense shaping reward r_{ds}, dense slack reward γ, and sparse success reward r_{ss}. The dense shaping reward is defined as:

$$r_{ds} = r_d(d_t, d_{t-1}) + [d_t \leq d_s] r_\alpha(\alpha_t, \alpha_{t-1}), \tag{1}$$

$$\text{where } [A] = \begin{cases} 1, & \text{if A is True} \\ 0, & \text{if A is False} \end{cases} \tag{2}$$

where r_d is the reduced distance to the goal from the current position relative to the previous one, and r_α is the reduced angle in radians to the goal view from the current view relative to the previous one. This reward function not only encourages the agent to approach the goal as much as possible, but also encourages the agent to rotate to a view as similar as possible to the goal view when the agent is close enough to the goal. At each time step t, the agent receives a reward r_t composed of shape reward and slack reward:

$$r_t = r_{ds} - \gamma \tag{3}$$

where $\gamma = 0.01$ is the slack penalty that encourages planning a shorter path to the goal. Once predicted a STOP action, the agent will receive a sparse success reward r_{ss} which is determined by its distance and angle to the goal:

$$r_{ss} = 5 \times ([d_t \leq d_s] + [d_t \leq d_s \, \text{and} \, \alpha_t \leq \alpha_s]). \tag{4}$$

Following [12], we set success distance $d_s = 1\text{m}$ and $\alpha_s = 25°$. As proven by [12, 7, 11], this reward enables the agent to learn to associate between observation v_o and goal image v_g. draw the association between its observation ot and the goal IG. Specifically, The agent will get a sparse reward $r_{ss} = 5$ if it is within d_s=1m from the goal, and 10 points if it is also within α_s=25° from the goal view. Otherwise, it will get a zero reward.

4 RGB setting. To compare with some methods that take panoramic images as input, we equip our agent with 4 pairwise orthogonal cameras to obtain a panoramic view. To reduce the computation cost, we only take the front image of these cameras as the goal image. During training and inference, each RGB image is combined with the goal image and input to our FGPrompt-EF model, and we concatenate all outputs as the visual-motor feature.

C More ablation study on skip fusion mechanism

As discussed in previous sections, we introduce an image feature matching module to provide the agent with fine-grained low-level goal prompts. A straightforward approach is applying the handcrafted local descriptors [6, 1] to detect and match paired image regions. It first detects the representative keypoints in the image and then matches the paired keypoints. The matched keypoints represent similar regions in two images that are scale-invariant, for example, the corner of a table or a part of a unique texture on a closet. However, computing these handcrafted features is time-consuming, as it requires computing Gaussian differences on different pyramid scales. In practice, this operation does not support high concurrency when training in the simulator and results in low FPS. To tackle this issue, we utilize a deep learning-based keypoint detector and matcher, called SuperPoint [2] and SuperGlue [10], in order to achieve batch inference on GPU devices. We provide the speed comparison in Table 1. To comprehensively explore how can we leverage this low-level information to perform goal prompting, we provide detailed ablation on this module. In Table 2, we compare different representation methods of the matched keypoints, where position denotes combining the normed pixel coordinate of each paired keypoint and descriptors means averaging the 256-dimension feature of each paired keypoint. We found that simply providing the agent with the location of matched points works the best.

Matching method	Device	FPS
SIFT	CPU	20
SuperPoint + SuperGlue	GPU	400

Table 1: **Comparing the forward speed of different image matching methods.** We report the frame per second (FPS) metric during training in the simulator.

Method	SPL	SR
Position	**37.1%**	**52.5%**
Descriptors	22.9%	38.1%
Descriptors + Position	24.2%	43.2%

Table 2: **Comparing different representations of the matched keypoints.** Directly combining the position of paired keypoints performs the best.

D More ablation study on mid fusion mechanism

Fusion layer. We have verified the effectiveness of fusing low-level information using low-level handcrafted descriptors in previous studies. As discussed in previous literature, the intermediate features in the earlier layer of deep convolution networks contain low-level information (*e.g.*, shape, texture, color, *etc.*). In the above ablation studies, we found that fusing these intermediate features using FiLM layers into observation encoder layers works. We further provide a detailed ablation study on the choice of the fusion layer. From Table 3, fusing later layers with coarse-grained contents performs worse than the first layer. These results show the importance of our proposed fine-grained goal prompting method.

Comparing more mapping schemes. In the previous sections, we have shown the priority of our proposed fine-grained goal prompting that mapping the intermediate high-resolution activation maps into the affine factors. To further verify the necessity of fine-grained and high-resolution mapping in the mid fusion mechanism, we provide detailed ablation on the semantic mapping methods, where we shift the source of the semantic goal prompt from the average pooled feature of each activation map to a high-dimension feature in the last layer. The poor performance of these two variants in Table 4 further indicates the importance of fine-grained and high-resolution mapping.

Comparing FiLM with self-attention. We conduct an experiment that replaces the FiLM module with a self-attention module. Specifically, we project the flattened feature map from the first layer of the goal encoder into the query and the correspondent sequence from the observation encoder

Layer	SPL	SR
1	**50.4%**	**77.3%**
2	44.4%	69.2%
3	45.9%	67.3%
4	37.1%	52.5%

Table 3: **Choice of fusion layer.** Early layers contain informative clues for prompting the observation encoder.

Mapping Method	SPL	SR
FG/HR	**50.4%**	**77.3%**
Semantic (each layer)	24.0%	32.0%
Semantic (last layer)	24.4%	32.3%

Table 4: **How to perform semantic mapping?** Neither mapping the global mean of each activation layer or semantic-level feature works.

Methods	SPL	SR
Self-attention	12.2%	13.9%
Ours	**50.4%**	**77.3%**

Table 5: **How to perform mid-fusion?** Self-attention performs significantly worse than FiLM layers.

Setting	SPL	SR
Ours w/o background	45.2%	64.4%
Ours w/ background	**50.4%**	**77.3%**

Table 6: **Importance of background context.** Removing background context in goal image leads to a performance decrease.

into key and value. Then, we utilize the self-attention operation to merge the goal and observation features. Experiment results are shown in Table 5.

Importance of environment context in goal image. We leverage the semantic annotation from 145 scenes in HM3D v2 dataset and set the pixel of background (e.g., uncountable object such as wall and floor) to zero according to the ground truth segmentation map. Numbers are reported in Table 6. From the experimental results, our method suffered from a slight degradation when the environmental context was removed from the goal image. These results indicate that environmental context in the image background provides useful but limited clues. We believe that objects with their arrangement in each room play a critical role in our FGPrompt.

E Comparison with memory-based methods

In Table 7, we directly compare our FGPrompt-EF with two different memory-based image navigation methods on the evaluation split B [3]. The reported results are averaged on both straight and curved path types. Although we do not use an additional memory module to store past agent states and model their relationship using the graph neural network, our method still shows priority over the memory-based methods. In our future work, we will discuss the effectiveness of our goal prompting module by involving it with the memory-based methods.

Method	Backbone	Pretrain	Sensor(s)	Memory	Split	SPL	SR
VGM [5]	ResNet18	N/A	4 RGB	✓	B	55.1%	75.3%
TSGM [4]	ResNet18	N/A	4 RGB	✓	B	76.9%	85.4%
FGPrompt-EF (Ours)	ResNet9	N/A	4 RGB	✗	B	**78.3%**	**96.4%**

Table 7: **Comparison with memory-based methods on eval split B.**

F Additional visualization results

Training curve vs. validation curve. We visualize the training and validation curve in both Success Rate and SPL metrics. As shown in Figure 3, our agent does not overfit the training scenes and has a consistent performance on both the training and validation episodes.

Visualizing FGPrompt-SF. In Figure 5, we visualize the matching results of the skip fusion mechanism of our proposed FGPrompt-SF. The first row shows a successfully matched example that numerous keypoints are detected and paired. The second and last rows show failure cases that the keypoint matching module makes incorrect predictions or failed to find corresponding regions, respectively. In the case of the agent's observation completely different from the goal image, this

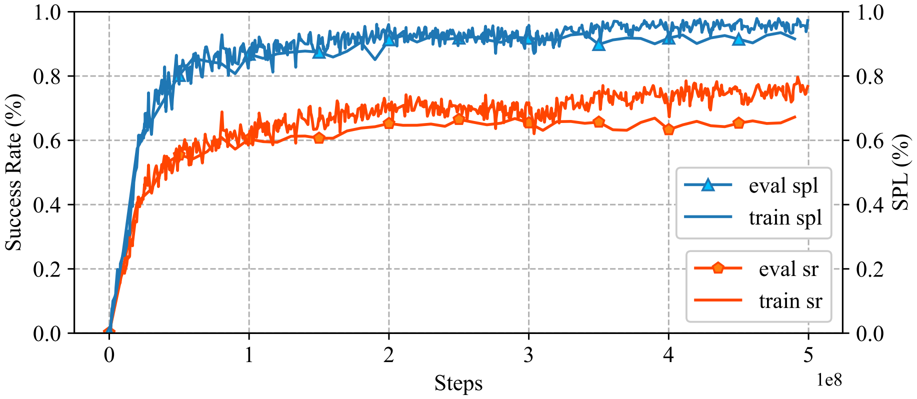
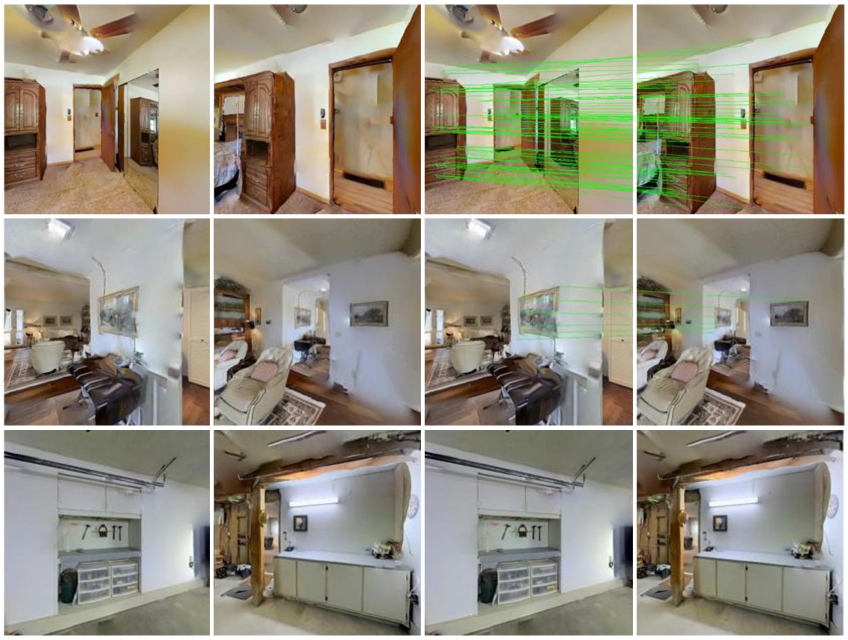


Figure 3: **Training and validation curve in Success Rate and SPL.**



(a) goal image

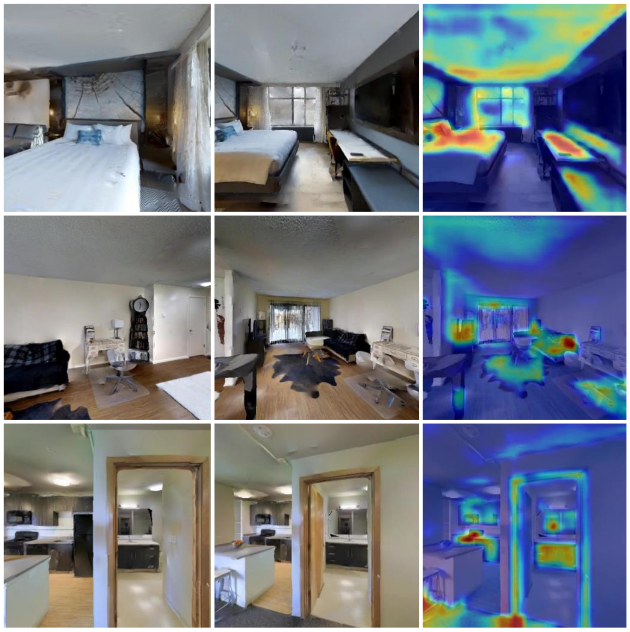
(b) observation image

(c) paired keypoints in goal image and observation image

Figure 4: **Visualization of the matching results of FGPrompt-SF.** Paired keypoints are connected with green lines in the last two columns.

matching module does not contribute to the navigation policy, which is particularly significant in the longer episodes that start far from the goal position.

Visualizing FGPrompt-EF. We show the visualization result of the early fusion mechanism of our proposed FGPrompt-SF in Figure 5. The last column present EigenCAM [9] visualized activation maps from the first layer of the joint encoder backbone. Guided by the fine-grained goal prompts in the early fusion scheme, the visual backbone focuses more on regions related to the goal image.



(a) goal image

(b) observation image

(c) Joint encoder act. (after fusion)

Figure 5: **Visualization of the activation map of FGPrompt-EF.** We use EigenCAM [9] to reveal where the model pays more attention.

References

[1] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005. 3

[2] D. DeTone, T. Malisiewicz, and A. Rabinovich. Superpoint: Self-supervised interest point detection and description. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 3

[3] M. Hahn, D. S. Chaplot, S. Tulsiani, M. Mukadam, J. M. Rehg, and A. Gupta. No rl, no simulation: Learning to navigate without navigating. In *Neural Information Processing Systems (NeurIPS)*, 2021. 2, 4

[4] N. Kim, O. Kwon, H. Yoo, Y. Choi, J. Park, and S. Oh. Topological semantic graph memory for image-goal navigation. In *Conference on Robot Learning (CoRL)*, pages 393–402. PMLR, 2023. 4

[5] O. Kwon, N. Kim, Y. Choi, H. Yoo, J. Park, and S. Oh. Visual graph memory with unsupervised representation for visual navigation. In *IEEE International Conference on Computer Vision (ICCV)*, pages 15890–15899, 2021. 4

[6] D. G. Lowe. Object recognition from local scale-invariant features. In *IEEE International Conference on Computer Vision (ICCV)*, 1999. 3

[7] A. Majumdar, G. Aggarwal, B. Devnani, J. Hoffman, and D. Batra. Zson: Zero-shot object-goal navigation using multimodal goal embeddings. In *Neural Information Processing Systems (NeurIPS)*, 2022. 2

[8] L. Mezghani, S. Sukhbaatar, T. Lavril, O. Maksymets, D. Batra, P. Bojanowski, and K. Alahari. Memory-augmented reinforcement learning for image-goal navigation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022. 2

[9] M. B. Muhammad and M. Yeasin. Eigen-cam: Class activation map using principal components. In *International Joint Conference on Neural Networks (IJCNN)*, 2020. 5, 6

[10] P. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich. Superglue: Learning feature matching with graph neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 3

[11] K. Yadav, R. Ramrakhya, A. Majumdar, V.-P. Berges, S. Kuhar, D. Batra, A. Baevski, and O. Maksymets. Offline visual representation learning for embodied navigation. In *International Conference on Learning Representations (ICLR)*, 2022. 2

[12] S. K. R. Ziad Al-Halah and K. Grauman. Zero experience required: Plug & play modular transfer learning for semantic visual navigation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2