DynaNav: Dynamic Feature and Layer Selection for Efficient Visual Navigation

Jiahui Wang[1] **Changhao Chen**[2]
[1]College of Design and Engineering, National University of Singapore
[2]PEAK-Lab, The Hong Kong University of Science and Technology (Guangzhou)
wjiahui@u.nus.edu changhaochen@hkust-gz.edu.cn

Abstract

Visual navigation is essential for robotics and embodied AI. However, existing foundation models, particularly those with transformer decoders, suffer from high computational overhead and lack interpretability, limiting their deployment in resource-tight scenarios. To address this, we propose **DynaNav**, a **Dyna**mic Visual **Nav**igation framework that adapts feature and layer selection based on scene complexity. It employs a trainable hard feature selector for sparse operations, enhancing efficiency and interpretability. Additionally, we integrate feature selection into an early-exit mechanism, with Bayesian Optimization determining optimal exit thresholds to reduce computational cost. Extensive experiments in real-world-based datasets and simulated environments demonstrate the effectiveness of DynaNav. Compared to ViNT, DynaNav achieves a $2.26\times$ reduction in FLOPs, 42.3% lower inference time, and 32.8% lower memory usage, while improving navigation performance across four public datasets.

1 Introduction

Visual navigation is a fundamental capability for robotics and embodied AI, enabling autonomous agents to perceive, interpret, and navigate complex 3D environments based on visual inputs [1–3]. Its applications span real-world scenarios, such as delivery and logistics, as well as virtual domains, including gaming and simulation. By bridging perception and action, visual navigation plays a crucial role in intelligent systems. Recently, there has been growing interest in developing foundation models for visual navigation [4–10, 1]. ViNT [5] is a notable example that learns from large-scale egocentric observations using transformer layers on CNN-extracted features, demonstrating strong generalization across robotic platforms and environments. NoMad [6] further builds on this by incorporating a diffusion policy and a goal-masking mechanism. PixNav [9] utilizes textual heuristics and large language models(LLMs) to explore zero-shot possibility. However, these approaches, particularly those relying on deep neural architectures such as transformer decoders, introduce significant computational overhead, posing challenges for edge deployment where efficiency is paramount.

Robotic applications demand greater efficiency than large cloud-based models. As the trend toward efficient foundation models continues [11, 12], reducing the computational burden of visual navigation models is a key challenge. Additionally, existing models function as "black boxes," raising concerns about interpretability. As humans and intelligent agents increasingly coexist, explainability becomes essential. These challenges lead to two critical research questions:

- *Is it necessary to activate all transformer layers for every navigation scenario?*
- *Which features are most important in the decoding process, and can we identify the most salient regions or pixels for navigation?*

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

Humans do not always activate all neurons for visual tasks [13]; rather, the brain dynamically recruits resources based on task complexity. Inspired by this, we propose that visual navigation models should adopt dynamic inference mechanisms, selectively utilizing features and neural layers based on scene complexity. In simpler scenarios, the model should rely on fewer features and layers for efficient computation, whereas in more complex tasks, it should allocate additional resources to ensure accurate decision-making.

To this end, we propose **DynaNav**, a highly efficient **Dyna**mic Visual **Nav**igation framework that adaptively selects relevant features and neural layers based on visual observations. Our approach employs a trainable hard feature selector to create sparse representations, enabling computationally efficient sparse operations at the feature level. This dynamic feature masking not only lowers computational overhead but also improves the understanding of which regions more relevantly influence the inference of visual navigation models, thereby enhancing explainability. Additionally, we introduce an early-exit strategy for deep Transformer layers by integrating feature selection into the early-exit mechanism, improving stability and computational efficiency. After training the decoder, Bayesian Optimization determines optimal early-exit thresholds. During inference, if a layer's feature meets its threshold, computation terminates early, significantly reducing overall computational cost. Extensive experiments on real-world datasets and in simulated environments demonstrate the effectiveness of our proposed DynaNav. Compared to ViNT [5], DynaNav achieves a $2.26\times$ reduction in FLOPs, 42.3% lower inference time, and 32.8% lower memory usage while improving navigation performance across four public datasets. To the best of our knowledge, this is the first work to introduce dynamic network mechanisms to visual navigation models. To sum up, the main contributions of our work can be summarized as follows:

- We propose DynaNav, a highly efficient and effective dynamic neural model for visual navigation, introducing a novel feature and layer selection strategy to improve efficiency without compromising performance.
- We integrate sparse feature selection into the early exit mechanism, improving the stability and success rate of dynamic layer inference, while the visualized mask enhances the interpretability of the navigation decision process.
- Extensive experiments and simulations demonstrate that DynaNav achieves more than twice the efficiency while maintaining comparable success rates.

2 Related Work

2.1 End-to-end Visual Navigation

Nowadays, conducting robot learning from diverse datasets to obtain a general model is becoming more and more popular [14–16]. Nonetheless, current approaches rely on real-world data, which is usually costly to obtain, lacks generalization, and is highly coupled with specific robot settings that are hard to transfer to different platforms [17, 18]. Instead, our paper follows the paradigm of learning navigation behavior from data collected across multiple different real-world robotic systems [19, 20, 4] while focusing on training a foundation model that can be adapted for various downstream tasks in zero-shot or with small amounts of data. To this end, models like RT-1, I2O, and GNM [21, 4, 22] provide useful insights that study broad generalization across environments and embodiments for robots deployed in real-world settings. GNM [4] demonstrates policy learning from heterogeneous RGB datasets but focuses on the singular task of reaching image goals in the zero-shot setting. ViNT [5] trains an effective visual navigation policy that can solve a range of downstream tasks, such as navigating to GPS goals [23], goal images [24], and skill-conditioned driving [25]. Building upon extensive prior work in visual navigation, ViNT combines two key elements: it uses topological graphs to keep track of how spaces are connected in the environment while employing trained policies to handle the detailed movement controls [26–29, 7, 30]. Recently, NoMaD [6] boosted the navigation task in previously unseen environments with goal masking techniques and diffusion policy.

2.2 Dynamic Network and Early Exit

Dynamic networks [31–33] tend to optimize models that can modify their architecture or parameters based on the input during the inference process. There are many techniques to achieve a dynamic

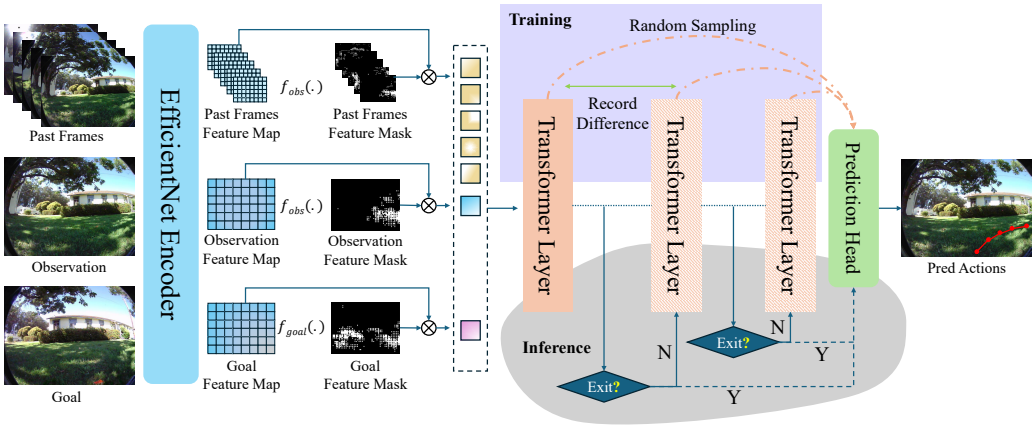

Figure 1: The architecture of our DynaNav framework. DynaNav employs two encoder instances of the same architecture: one processes the concatenated observation and historical frame sequence, while the other extracts features from the early-fused combination of the current observation and goal image. Two independent feature selectors generate masks for observations and the goal, which are then tokenized and fed into a Transformer decoder. Training incorporates stochastic early-exit triggers at intermediate decoder layers. During inference, a decision step at each layer evaluates optimized thresholds to determine whether early exit conditions are met.

network depth-wise and width-wise. For instance, layer-skipping [34], neurons-skipping [35], and low rank approximation (LoRA) [36]. Moreover, some dynamic networks focus on adjusting the shape and value of weights adaptively during the inference, such as deformable convolution [37], dynamic filter network [38], etc.. Among these techniques, early exiting gained popularity because of the prevalent Transformer-based model, which fits the inherent architecture with stacked blocks.

Early exiting is a depth-wise dynamic method for halting forward propagation at a certain layer based on intermediate predictions. This technique has been well explored in both computer vision [39–44], language processing [45–51], and multimodality [52, 53]. One challenge in implementing early-exiting models lies in finding a suitable metric to decide when to make an intermediate prediction. Commonly, metrics like probability confidence [41] or entropy value [47] are employed in traditional vision tasks. Some research also pointed out the possibility of using learning-based early exit, which relies on a trained network [54, 46, 55, 56]. Recent research [32, 34] has extended early exiting to the LLMs, which treat the autoregressive task as a classification subgoal.

In this work, we are the first to utilize the idea of early exiting on an end-to-end visual navigation model. We further develop the current method [11] with the integration of sparse feature selection to the Bayesian optimization process in obtaining the desirable metric.

3 Dynamic Visual Navigation Model

3.1 Framework Overview

To enable efficient and effective visual navigation, we propose DynaNav, a dynamic navigation pipeline illustrated in Figure 1. Unlike previous end-to-end models with static network inference, DynaNav integrates a dynamic feature selector and an early-exit mechanism, reducing computational costs while enhancing explainability and robustness. DynaNav begins with an EfficientNet backbone [57] to extract features from RGB image sequences. Building on ViNT [5], we introduce a feature selector module that generates masks before feature processing in the Transformer decoder, allowing for sparse computation. Additionally, we implement a dynamic Transformer decoder, enabling predictions at intermediate layers to improve efficiency. Finally, Bayesian optimization [11] determines the optimal early-exit thresholds for our jointly trained model, further minimizing computational overhead.

Feature Extraction. We chose EfficientNet-B0 as our visual encoder due to its innovative compound scaling method, which optimally balances network width, depth, and resolution. Mathematically, the encoder processes an input sequence consisting of consecutive visual observations $\mathbf{o}_i$, where $i \in [t-p, t]$, along with a goal image $\mathbf{o}_s$. Each observation is first mapped to a latent space representation by the encoder, denoted as $\psi(\mathbf{o}_i) \in \mathbb{R}^{H \times W \times C}$, where $\psi(\cdot)$ represents the network, and H, W, C correspond to the height, width, and number of channels in the feature map. To enhance the connection between the current observation and the goal, we adopt an early fusion strategy. Specifically, the goal image is processed separately by another EfficientNet instance, producing a feature representation denoted as $\phi([\mathbf{o}_t; \mathbf{o}_s]) \in \mathbb{R}^{H \times W \times C}$, where $[;]$ represents concatenation. The detailed hyperparameter settings are provided in Section B.1.

Dynamic Feature Selector. However, when the embedded feature map is large [11], computing such a tensor in a transformer incurs significant computational costs [58], which limits the navigation model's efficiency. Moreover, not all pixels in the observations and goal are essential; some redundant pixels can be ignored to improve processing efficiency [59]. To address this, we introduce a dynamic hard feature selector that generates a mask to filter out pixels with minimal relevance to the final prediction.

Transformer Decoder. After the feature selection process, a transformer decoder $\mathbf{D}$ is employed to extract contextual features for action prediction, as illustrated in Figure 1. The stacked multi-head self-attention (MHSA) layers continuously refine the contextual information of visual tokens. Formally, we define the intermediate token as:

$$\mathbf{x}_i = D_{1:i}\left([\mathbf{m}_{t-p:t} * \psi(\mathbf{o}_{t-p:t}); \mathbf{m}_s * \phi([\mathbf{o}_t; \mathbf{o}_s])]\right), \tag{1}$$

where $\mathbf{m}_{t-p:t}$ represents the generated masks for $\mathbf{o}_i$, $i \in [t-p, t]$, and $D_{1:i}$, $i \in [1, l]$, denotes the first i layers of the decoder. The process for obtaining $\mathbf{m}$ is detailed in Section 3.2.

Navigational Action Prediction. Finally, a head network is trained to predict both the action $\mathbf{a}_t$ and the waypoint distance vector $\mathbf{w}_t$. When feature selection is applied, this prediction process can be formulated as $\mathbf{a}_t, \mathbf{w}_t = h(\mathbf{x}_l)$, where h represents the prediction head. In our implementation, h consists of a 4-layer transformer followed by an MLP with a single hidden layer.

Training Objective. During training, we sample a sequence of visual images from the dataset to construct the observation $\mathbf{o}_{t-p:t}$. A goal image $\mathbf{o}_s$ is randomly selected for a valid prediction length, where $\mathbf{o}_s = \mathbf{o}_{t+d}$ and $d \in [t_{\min}, t_{\max}]$. The corresponding action sequence $\mathbf{a}_t^{\text{gt}} = \mathbf{a}_{t:t+d}$ and waypoint $\mathbf{w}_t^{\text{gt}}$ serve as ground truth. The objective of training is to maximize the likelihood of the predicted outputs aligning with the ground truth, formulated as:

$$\mathcal{L} = \mathbb{E}\left[\log p\left(\mathbf{a}_t^{\text{gt}} \mid \mathbf{a}_t\right) + \lambda \log p\left(\mathbf{w}_t^{\text{gt}} \mid \mathbf{w}_t\right)\right]. \tag{2}$$

3.2 Dynamic Sparse Feature Selection

End-to-end visual navigation models often operate as black boxes [4, 5, 60, 7], making it unclear which parts of an observation contribute most to action prediction. Understanding these key elements could enable targeted preprocessing to enhance model performance. This leads to a fundamental question: *should all pixels be treated equally?* Intuitively, the answer is noemphasizing only relevant pixels improves robustness. In real-world scenarios, indiscriminate reliance on all pixels can lead to failures, especially when obstacles obstruct a robots camera. To address this, we introduce a feature selection approach based on the Gumbel-Softmax mechanism [61], dynamically prioritizing critical features. This improves performance and adaptability across diverse environments and provides meaningful insights into the model's region of interest. The feature selector

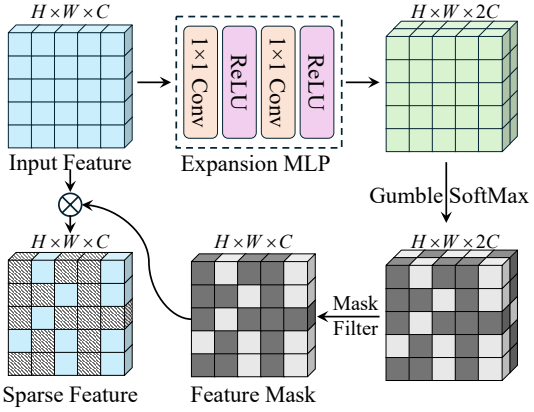

Figure 2: Architecture of the dynamic feature selector. A trainable MLP projects the input features to twice their original dimension. A pixel-wise Gumbel-Softmax operation is then applied to compute selection probabilities.

functions as a classification network, assigning each pixel a probability score to generate masks for different input features. As shown in Figure 2, the feature selector $f(\cdot)$ takes encoded features as input and outputs corresponding masks as follows,

$$\mathbf{m}_i = f(\psi(\mathbf{o}_i)); \;\; \mathbf{m}_s = f(\phi([\mathbf{o}_t; \mathbf{o}_s])) \in \mathbb{R}^{H \times W}. \tag{3}$$

Within the feature selector, the latent feature $\psi(\mathbf{o}_i)$ is projected into a higher-dimensional space:

$$\mathbf{Z}_i = MLP(\psi(\mathbf{o}_i)) \in \mathbb{R}^{H \times W \times C \times 2}, \tag{4}$$

where $MLP(\cdot)$ denotes a multi-layer perceptron. Here, $z_i^{n,c,k}$ represents the unnormalized log probability of the k-th category for the n-th pixel and c-th channel. To obtain the one-hot mask, we utilize the Gumble-SoftMax trick, which first adds a log term to each element and then conducts the SoftMax. The logarithm term is defined as:

$$g_i^{n,c,k} = -\log\left(-\log\left(u^{n,c,k}\right)\right); \quad u^{n,c,k} \sim U(0,1), \tag{5}$$

and the processed value in $\mathbf{Z}_i$ is $\bar{z}_i^{n,c,k} = z_i^{n,c,k} + g_i^{n,c,k}$. Then the SoftMax is applied on $\mathbf{Z}_i$ along the last dimension:

$$\hat{z}_i^{n,c,k} = \frac{\exp(\frac{\bar{z}_i^{n,c,k}}{\tau})}{\sum_{k'=1}^{2} \exp(\frac{\bar{z}_i^{n,c,k'}}{\tau})}, \quad k = 1, 2, \tag{6}$$

where τ is a temperature hyperparameter. At last, we manually define the last dimension of $\hat{\mathbf{Z}}_i$ as the generated mask, i.e. $m_i^{n,c} = \hat{z}_i^{n,c,2}$. The feature selector will gradually filter out the undesired features as the training continues.

Figure 3 presents the visualized input and its gradients that are processed through the feature selector. The spatial importance weights are visualized through saliency maps, where the attention mask is upsampled to match the input dimensions. The brightness intensity of each pixel in the visualization corresponds to its selection probability by the feature selection mechanism. The results indicate redundancy within the input data, while the navigation model does not specifically focus on the largest common object between the observation and the goal. This finding not only supports the feasibility of filtering pixels but also enhances the interpretability of the navigation process. After selection, we can utilize data sparselization techniques [62, 63] to save space.

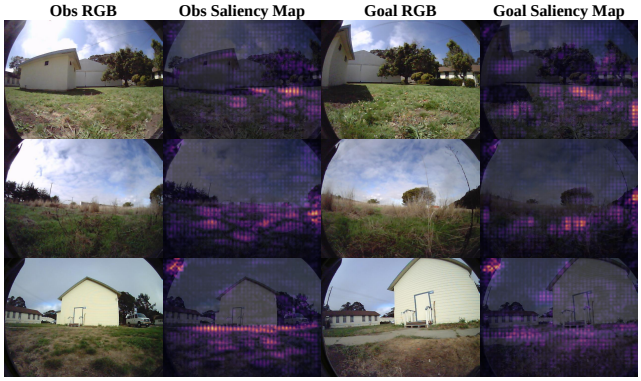

Figure 3: Visualization of saliency maps for observation and goal images.

3.3 Dynamic Transformer Layer Inference

3.3.1 Feature-Aware Early Exit Strategy

Transformer-based decoders are highly effective in visual navigation, leveraging long-range dependencies and flexible adaptation [64–67]. However, models like ViNT [5] employ a scene-agnostic decoder that activates all layers indiscriminately, disregarding scene complexity and task requirements. While beneficial for large-scale training, this approach imposes excessive computational demands on edge devices. We argue that activating every layer is often unnecessarysimilar to how humans selectively engage neurons for cognitive tasks [13, 68]. To address this, we propose a ***dynamic*** navigation decoder with an early-exit mechanism, allowing the model to halt computation based on scene complexity and navigation needs. By leveraging intermediate features for action prediction, this methodthe first to introduce early exiting in visual navigationeliminates redundant computations. Additionally, we enhance efficiency and robustness by integrating feature selection as

an initial step in the early-exit strategy. Our approach significantly reduces computational overhead while maintaining performance, making it well-suited for resource-constrained deployment.

Figure 1 outlines our early-exiting strategy workflow. DeeR-VLA [11] proposes a metric for robotic tasks, arguing that transformer layer features are inherently distinct. It uses an action consistency condition, measuring the difference in action outputs from an action head h:

$$|h(\mathbf{x}_i) - h(\mathbf{x}_{i-1})|_2 \leq \eta_i, \quad \forall i \in \{1, 2, \ldots, l\}. \tag{7}$$

However, this still requires activating multiple layers, limiting computational savings. To improve efficiency, we introduce an aggressive early-exit strategy. When the L2 difference between the goal state and the current observation falls below a training-derived threshold (based on masked pixel counts), we bypass the transformer decoder entirely and compute actions directly from the encoded tokens and a prediction head.

3.3.2 Adaptive Threshold Optimization

To determine the optimal early-exit threshold, we employ Bayesian Optimization [69, 11] to iteratively search for the best value under given constraints. Specifically, we consider the predicted action $\mathbf{a}_t$ and waypoint $\mathbf{w}_t$ alongside their respective ground truth values, $\mathbf{a}_t^{\text{gt}}$ and $\mathbf{w}_t^{\text{gt}}$. Our objective is to maximize the cosine similarity between predictions and ground truth by optimizing the early-exiting thresholds, denoted as $\eta = \{\eta_1, \eta_2, \ldots, \eta_N\}$. Consequently, the objective function is formulated as:

$$\max_{\eta} J(\eta) = \frac{1}{T} \sum_{t=1}^{T} \Big(\text{Sim}(\mathbf{a}_t, \mathbf{a}_t^{\text{gt}}; \eta) + \lambda \cdot \text{Sim}(\mathbf{w}_t, \mathbf{w}_t^{\text{gt}}; \eta) \Big), \tag{8}$$

where $\text{Sim}(\mathbf{u}, \mathbf{v}; \eta)$ represents the cosine similarity between two vectors with a given early exit threshold η. $\lambda > 0$ is a hyperparameter that balances waypoint and action prediction, and T is the total number of time steps in the task. To optimize this objective function, we introduce a penalty function $P(\eta)$ that enforces the required constraints. This function assigns a positive value when η violates any constraint and remains zero otherwise. Incorporating this penalty into the optimization framework, we reformulate the problem as:

$$\max_{\eta} V(\eta) = J(\eta) - P(\eta), \tag{9}$$

where the penalty term, $P(\eta) = \sum_k \xi_k \cdot \max(0, g_k(\eta))$, captures the weighted sum of constraint violations. Here, $g_k(\eta)$ quantifies the extent to which the k-th constraint is violated, while ξ_k represents its associated weight (remain constant). The specific constraints that the model must satisfy are outlined below.

Inference Time Constraint. Let $\text{Time}(\eta)$ denote the average inference time over the entire test set, where η represents the early exit decision parameters. To ensure the efficiency of the network, we impose a constraint that the average inference time remains below a predefined threshold $T_{\max}$. Mathematically, this can be formulated as:

$$\text{Time}(\eta) = \frac{1}{T} \sum_{t=1}^{T} \text{Time}_t(\eta), \text{ s.t. } \text{Time}(\eta) \leq \mathcal{T}_{\max}. \tag{10}$$

T is the total number of test samples, $\mathcal{T}_{\max}$ is the maximum time, and $\text{Time}_t(\eta)$ denotes the inference time for the t-th sample under the given early exit strategy. This constraint ensures that the optimization selects an early exit criterion that balances computational efficiency, predictive performance, and real-time or application-specific latency requirements.

GPU Memory Constraint.To ensure efficient deployment under limited GPU resources, we define $\text{Mem}(\eta)$ as the GPU memory usage when applying the early exit strategy. Since memory consumption can fluctuate during inference, we consider the peak memory usage across all inference steps and enforce an upper bound constraint:

$$\text{Mem}(\eta) = \max_{t=1,\ldots,T} \text{Mem}_t(\eta), \text{ s.t. } \text{Mem}(\eta) \leq G_{\max} \tag{11}$$

where $\text{Mem}_t(\eta)$ represents the memory consumption at time step t, and $G_{\max}$ denotes the maximum allowable GPU memory. This constraint ensures that Bayesian optimization selects an early exit criterion that not only improves efficiency but also maintains feasibility within hardware limitations.

Table 1: Quantitative Comparison on Benchmarks. We highlight our method with the colored font , the best and the second best value of each metric are reported with **bold** and underlined fonts, respectively.

Dataset	Method	Sim$(\mathbf{a}_t, \mathbf{a}_t^{\text{gt}})\uparrow$	Sim$(\mathbf{w}_t, \mathbf{w}_t^{\text{gt}})\uparrow$	$\mathcal{L}_{action}\downarrow$	$\mathcal{L}_{dist}\downarrow$	FLOPs (10^9)	Time(s/traj)	Memory (Gb)
Recon [5]	ViNT [5]	94.49	96.20	0.285	6.94	4.37	0.379	19.07
	NoMad [6]	-	**96.64**	0.207	6.44	7.46	1.118	21.36
	Ours	**94.92**	96.53	**0.191**	**6.26**	**1.93**	**0.228**	**13.35**
Go-Stanford [4]	ViNT [5]	88.50	93.47	0.531	15.80	4.37	0.379	19.07
	NoMad [6]	-	93.51	0.507	**12.93**	7.46	1.118	21.36
	Ours	**89.07**	**93.66**	**0.449**	14.23	**1.68**	**0.209**	**12.27**
SACSoN [71]	ViNT [5]	89.66	93.16	0.686	10.95	4.37	0.379	19.07
	NoMad [6]	-	93.69	0.501	9.66	7.46	1.118	21.36
	Ours	**90.54**	**93.72**	**0.493**	**9.62**	**1.68**	**0.209**	**12.27**
SCAND [70]	ViNT [5]	95.43	96.89	0.141	16.08	4.37	0.379	19.07
	NoMad [6]	-	**97.79**	**0.121**	**13.05**	7.46	1.118	21.36
	Ours	**96.85**	97.03	0.130	14.41	**1.93**	**0.228**	**13.25**

FLOPs Constraint. One of the most critical considerations in optimizing the early exit strategy is controlling the computational cost, particularly in the transformer decoder. To achieve this, we define FLOPs(η) as the average floating point operations (FLOPs) required per trajectory. Our goal is to ensure that the computational complexity remains within a predefined upper bound, $F_{\max}$, while maintaining the models performance. Formally, we express this constraint as:

$$\text{FLOPs}(\eta) = \frac{1}{T}\sum_{t=1}^{T}\text{FLOPs}_t(\eta) \text{ s.t. FLOPs}(\eta) \leq F_{\max}. \tag{12}$$

Here, FLOPs$_t(\eta)$ represents the computational cost at each exit point t. By integrating this constraint into our Bayesian optimization framework, we explore the trade-off between computational efficiency and model accuracy, enabling us to identify the most effective early exit criteria within the computational limits.

4 Experiment

4.1 Experimental Setups

4.1.1 Datasets

We evaluated our method in two experimental settings: benchmark datasets and simulated environments. For the benchmark datasets, we select four diverse datasets to assess the performance of our approach under various conditions. These include the Recon dataset [5], which provides medium-speed ($2m/s$) outdoor data to evaluate our method in real-world, dynamic outdoor settings, and the SCAND dataset [70], a medium-speed dataset featuring environmental interactions. Additionally, we include the Go-Stanford dataset [30] and the SACSoN dataset [71], representing low-speed ($0.5m/s$) and medium-speed indoor scenarios, respectively. These datasets allow us to test our method across environments with varying speed characteristics. All datasets are pre-processed following the ViNT method [5] to ensure consistency across experiments. For each dataset, we randomly split the data into training (80%) and testing (20%) sets. The implementation detail is illustrated in Section B in the Appendix.

4.1.2 Evaluation Metrics

For the benchmark comparison, we report the cosine similarity between action angles and predicted waypoints, denoted as Sim$(\mathbf{a}_t, \mathbf{a}_t^{\text{gt}})$ and Sim$(\mathbf{w}_t, \mathbf{w}_t^{\text{gt}})$, respectively (in percentage). Since NoMad [6] only outputs waypoints through the diffusion process, we omit the action angle term for this model. To highlight the efficiency advantages of our approach, we report the FLOPs and memory usage of each model on the entire evaluation set. For inference, we measure the time required to predict a single trajectory for each model. Additionally, we report the loss values for the action vector and distance.

For the CARLA [72] simulation, we track the progress of our model-driven agent until it either reaches the target or encounters a collision. The success rate is calculated as the mean of the ratio of progress length to total trajectory length.

4.2 Evaluation in Real-world Benchmarks

Table 1 illustrates the performance of our method on RECON [5], Go-Stanford [4], SACSoN [71], and SCAND [70] datasets. We compare the performance with ViNT [5] and NoMad [6]. All models are trained from scratch. Our model saves about 58% FLOPs across all benchmarks compared to ViNT [5] while maintaining comparable accuracy. Figure 4 depicts the efficiency advantages of our model compared to ViNT [5]. Our approach achieves a 0.83% improvement in $\text{Sim}(\mathbf{a}_t, \mathbf{a}_t^{\text{gt}})$ and a 0.28% increase in $\text{Sim}(\mathbf{w}_t, \mathbf{w}_t^{\text{gt}})$ compared to ViNT [5] across four benchmarks.

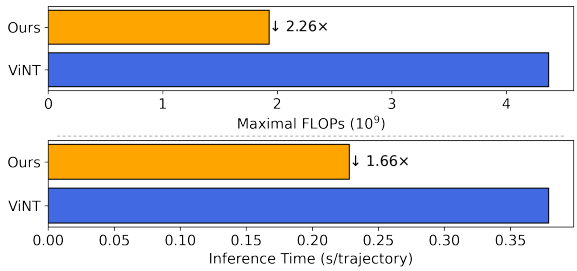

Figure 4: Efficiency comparison of **upper:** FLOPs, **bottom:** inference time between ours and ViNT on RECON dataset.

In terms of time efficiency, we save 0.16 seconds compared to ViNT [5] and 0.89 seconds compared to NoMaD [6]. Despite this, NoMaD [6], due to its diffusion refinement procedure, achieves an average performance that is 0.2% higher than ours. However, NoMaD [6] requires approximately four times the FLOPs of our method, making it less efficient. Notably, the average FLOPs of our dynamic model in RECON [5] and SCAND [70] are higher than those in SACSoN [71] and Go-Stanford [4]. One reason for this is that the former two datasets are from outdoor environments, while the latter two consist of indoor scenarios. The indoor datasets benefit from lower speeds, more controllable environments, and less complex lighting conditions. This finding also validates the assumption that for a more complex scene, activating more layers for accurate navigation.

4.3 Real-time Robotic Navigation in CARLA Simulation Environment

For the simulation, we first collected 200 trajectories of inline navigation data from CARLA [72] Town01 to fine-tune the pre-trained model. The data was gathered using an RGB camera and various sensors mounted on an autopilot agent operating at a frequency of 4Hz. To ensure consistency, we standardized the image size to 640×480 pixels with a 90° field of view (FOV). We evaluated our method across three distinct CARLA environments: Town02 (Scene A), Town03 (Scene B), and Town10 (Scene C). Scene A, with a small-town layout and simple residential-commercial mix, represents an "easy task" for the agent. Scene B, a larger urban map with roundabouts and large junctions, is considered a "medium task." Scene C, a downtown area filled with skyscrapers, residential buildings, and parked cars, presents a highly dynamic and complex environment, making it a "hard task." For each scene, we collected 20 trajectories, which were used in the subsequent testing phase. The car was driven by a BehaviorAgent [72], maintaining a consistent maximum speed of 20 km/h across all environments. More details are illustrated in Appendix Section B.

Figure 5: The simulation result in CARLA Town02 environment. The green dots represent the discrete goals, and the red dots represent the predicted waypoints.

Figure 5 illustrates the visualized navigation performance of baselines and ours. GNM [4] lacks enough generalization ability to achieve the target task. ViNT [5] and our method can successfully reach the target points. Note that the trajectory of ViNT [5] has some drift. This is due to ViNT [5] utilizing all features and decoder layers, potentially overfitting to training data and producing sub-

Table 3: Ablation study of the effectiveness of individual modules on the RECON dataset.

Dynamic decoder	Feature selector	$\text{Sim}(\mathbf{a}_t, \mathbf{a}_t^{\text{gt}})$	$\text{Sim}(\mathbf{w}_t, \mathbf{w}_t^{\text{gt}})$	$\mathcal{L}_{action}$	$\mathcal{L}_{dist}$	FLOPs (10^9)	Time	Memory
Half layers	-	91.05	93.28	0.332	7.53	2.61	0.306	17.48
Half channel	-	89.70	92.41	0.390	7.71	2.19	0.270	12.11
-	-	94.49	96.20	0.285	6.94	4.37	0.379	19.07
✓	-	93.68	95.42	0.274	7.08	2.41	0.251	16.49
-	✓	94.81	96.44	0.205	6.30	4.06	0.377	18.22
✓	✓	**94.92**	**96.53**	**0.191**	**6.26**	**1.93**	**0.228**	**13.35**

optimal trajectories. Our approach dynamically activates transformer layers and selectively filters features, resulting in superior trajectory performance.

Table 2 presents the success rate of different models on the CARLA [72] simulation. NoMad [6] is unable to achieve agile real-time simulation on our test platform due to the computationally intensive nature of its diffusion process. The results show that, although our model has higher FLOPs than GNM [4], its success rate shows a 38% improvement, demonstrating the effectiveness of our approach. Compared to ViNT [5], our method not only achieves comparable performance but also reduces FLOPs by more than a factor of two. Furthermore, as the simulation environment becomes more challenging (Scene A→Scene C), the FLOPs required by our model increase. This is because we use a unified early exit metric across all three simulation environments. As the visual discrepancy between the observation and goal increases, our model needs more decoder blocks to extract contextual information effectively.

Table 2: The comparison of our model with baselines in the CARLA under various environments. The best and the second best values of each metric are reported with **bold** and underlined fonts, respectively.

Environment	Model	Successful Rate	FLOPs (10^9)
Scene A	GNM [4]	0.297	1.09
	ViNT [5]	<u>0.724</u>	4.37
	Ours	**0.727**	1.58
Scene B	GNM [4]	0.288	1.09
	ViNT [5]	<u>0.659</u>	4.37
	Ours	**0.664**	1.70
Scene C	GNM [4]	0.251	1.09
	ViNT [5]	**0.589**	4.37
	Ours	<u>0.588</u>	1.93

4.4 Ablation Study

Ablation into Individual Modules: Table 3 illustrates the effectiveness of our proposed module. The dynamic decoder column represents whether we are using early exit on the transformer decoder. The first row shows the result when we simply deactivate half of the decoder layers. Similarly, the second row presents the result when we deliberately reduce the hidden channel size from C to $\frac{C}{2}$. Although these settings can improve efficiency, they usually lead to decreased performance and poor generalization (i.e., high accuracy on the training set but low accuracy on the testing set).. The rest of Table 3 elaborates that without the dynamic decoder, the efficiency does not vary too much compared to the baseline. Moreover, by using the feature selector, the performance will be better, and the efficiency will also be boosted. This is because our proposed feature selector sparsifies the features and stabilizes the early exit process.

Ablation into Threshold Optimization: Table 4 shows the different performances of whether we implement an extra Bayesian Optimization (BO) after training as DeeR-VLA [11]. Moreover, it reports the influence of whether we allow an early exit before the decoder. Results show that without BO, the early exit process can be impaired due to a suboptimal threshold. Besides, if we allow the early exit before the transformer decoder, although it can gain efficiency improvement, the over-

Table 4: Ablation study on whether using post-training Bayesian Optimization (BO) and allowing exit before the decoder. The best and the second best values of each metric are reported with **bold** and underlined fonts, respectively.

BO	Pre-decoder Exit	$\text{Sim}(\mathbf{w}_t, \mathbf{w}_t^{\text{gt}})$	Successful Rate	FLOPs (10^9)
-	-	96.30	0.725	2.46
-	✓	96.22	0.719	2.27
✓	-	**96.58**	**0.732**	<u>2.11</u>
✓	✓	<u>96.53</u>	<u>0.727</u>	**1.93**

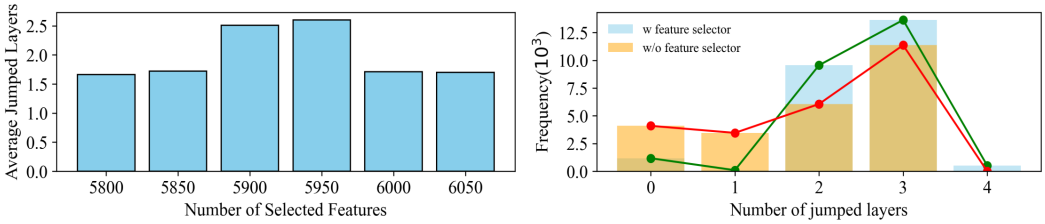

Figure 6: Visualized relationship between the number of selected features and the skipped layers (**Left**) and the frequency of different numbers of skipped layers in the context of with or without the feature selector (**right**).

all accuracy will slightly decrease. Therefore, such a technique ought to be a trade-off that requires careful design.

Ablation into Feature Selection for Early Exit: To assess the impact of our proposed feature selector on the early exit mechanism, we evaluate the model on the RECON [5] test set with a batch size of 1, both with and without the feature selector. For each sample, we record the MEAN value of the action difference between each layer (0-1,1-2,2-3). Moreover, we record the early exit index, referring to it as the number of skipped layers. In Figure 6, we observe that within a certain range of selected feature numbers, the average number of skipped layers remains high. This finding suggests that our feature selector helps determine an appropriate early exit threshold, thereby enhancing the frequency of early exiting. Additionally, Figure 6 shows that our proposed feature selector increases the frequency of 2-to-4 layer jumps, leading to improved efficiency. Therefore, by integrating the feature selector with early exit, our method achieves both more stable and more efficient performance.

5 Conclusion

In this work, we propose DynaNav, a novel, highly efficient visual navigation model. We first introduce a dynamic feature selector that filters observations and goals to extract robust, memory-efficient features. We also introduce feature-aware early exit criteria for the transformer decoders, using action consistency metrics optimized via Bayesian techniques. Our experimental results show a significant reduction in computational overhead compared to existing foundation navigation models while maintaining high performance across standard benchmarks and the CARLA simulation environment. The empirical evidence validates the effectiveness of our approach in achieving efficient and robust visual navigation.

To achieve optimal performance, our model requires an additional optimization process. Although the Bayesian optimization helps fine-tune the model and determine optimal thresholds, the added labor cost is non-negligible. Future work could involve implementing these optimization techniques concurrently with training to create a more streamlined end-to-end system. Furthermore, the proposed feature selection mechanism can be integrated with various CNN-based encoder models to improve their overall efficiency.

6 Acknowledgement

This research is supported by the National University of Singapore under the NUS College of Design and Engineering Industry-focused Ring-Fenced PhD Scholarship programme. Changhao Chen is funded by the Young Elite Scientist Sponsorship Program by CAST (No. YESS20220181) and the National Natural Science Foundation of China (NFSC) under the Grant Number 62573370.

References

[1] Y. Zhang, Z. Ma, J. Li, Y. Qiao, Z. Wang, J. Chai, Q. Wu, M. Bansal, and P. Kordjamshidi, "Vision-and-language navigation today and tomorrow: A survey in the era of foundation models," *arXiv preprint arXiv:2407.07035*, 2024.

[2] H. Li, M. Li, Z.-Q. Cheng, Y. Dong, Y. Zhou, J.-Y. He, Q. Dai, T. Mitamura, and A. Hauptmann, "Human-aware vision-and-language navigation: bridging simulation to reality with dynamic human interactions," *Advances in Neural Information Processing Systems*, vol. 37, pp. 119 411–119 442, 2025.

[3] A. Bar, G. Zhou, D. Tran, T. Darrell, and Y. LeCun, "Navigation world models," in *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, 2025, pp. 15 791–15 801.

[4] D. Shah, A. Sridhar, A. Bhorkar, N. Hirose, and S. Levine, "Gnm: A general navigation model to drive any robot," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 7226–7233.

[5] D. Shah, A. Sridhar, N. Dashora, K. Stachowicz, K. Black, N. Hirose, and S. Levine, "Vint: A foundation model for visual navigation," in *Conference on Robot Learning (CoRL)*. PMLR, 2023, pp. 711–733.

[6] A. Sridhar, D. Shah, C. Glossop, and S. Levine, "Nomad: Goal masked diffusion policies for navigation and exploration," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 63–70.

[7] D. Shah, B. Eysenbach, G. Kahn, N. Rhinehart, and S. Levine, "Ving: Learning open-world navigation with visual goals," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 13 215–13 222.

[8] D. Shah, B. Osiński, S. Levine *et al.*, "Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action," in *Conference on robot learning*. PMLR, 2023, pp. 492–504.

[9] W. Cai, S. Huang, G. Cheng, Y. Long, P. Gao, C. Sun, and H. Dong, "Bridging zero-shot object navigation and foundation models through pixel-guided navigation skill," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 5228–5234.

[10] G. Zhou, Y. Hong, and Q. Wu, "Navgpt: Explicit reasoning in vision-and-language navigation with large language models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 7, 2024, pp. 7641–7649.

[11] Y. Yue, Y. Wang, B. Kang, Y. Han, S. Wang, S. Song, J. Feng, and G. Huang, "Deer-vla: Dynamic inference of multimodal large language models for efficient robot execution," in *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[12] X. Sun, P. Zhang, P. Zhang, H. Shah, K. Saenko, and X. Xia, "Dime-fm: Distilling multimodal and efficient foundation models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 15 521–15 533.

[13] A. A. Bharath and M. Petrou, *Next generation artificial vision systems: Reverse engineering the human visual system*. Artech House, 2008.

[14] C. Devin, A. Gupta, T. Darrell, P. Abbeel, and S. Levine, "Learning modular neural network policies for multi-task and multi-robot transfer," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 2169–2176.

[15] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn, "Robonet: Large-scale multi-robot learning," in *Conference on Robot Learning (CoRL)*. PMLR, 2020, pp. 885–897.

[16] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 2636–2645.

[17] A. Kadian, J. Truong, A. Gokaslan, A. Clegg, E. Wijmans, S. Lee, M. Savva, S. Chernova, and D. Batra, "Sim2real predictivity: Does evaluation in simulation predict real-world performance?" *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6670–6677, 2020.

[18] P. Anderson, A. Shrivastava, J. Truong, A. Majumdar, D. Parikh, D. Batra, and S. Lee, "Sim-to-real transfer for vision-and-language navigation," in *Conference on Robot Learning (CoRL)*. PMLR, 2021, pp. 671–681.

[19] N. Hirose, D. Shah, A. Sridhar, and S. Levine, "Exaug: Robot-conditioned navigation policies via geometric experience augmentation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 4077–4084.

[20] A. Loquercio, A. I. Maqueda, C. R. Del-Blanco, and D. Scaramuzza, "Dronet: Learning to fly by driving," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1088–1095, 2018.

[21] J. Truong, A. Zitkovich, S. Chernova, D. Batra, T. Zhang, J. Tan, and W. Yu, "Indoorsim-to-outdoorreal: learning to navigate outdoors without any outdoor experience," *IEEE Robotics and Automation Letters*, 2024.

[22] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu *et al.*, "Rt-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022.

[23] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik *et al.*, "Habitat: A platform for embodied ai research," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 9339–9347.

[24] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3357–3364.

[25] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4693–4700.

[26] N. Savinov, A. Dosovitskiy, and V. Koltun, "Semi-parametric topological memory for navigation," in *International Conference on Learning Representations (ICLR)*, 2018.

[27] J. Bruce, N. Sunderhauf, P. Mirowski, R. Hadsell, and M. Milford, "Learning deployable navigation policies at kilometer scale from a single traversal," in *Conference on Robot Learning (CoRL)*. PMLR, 2018, pp. 346–361.

[28] A. Faust, K. Oslund, O. Ramirez, A. Francis, L. Tapia, M. Fiser, and J. Davidson, "Prm-rl: Long-range robotic navigation tasks by combining reinforcement learning and sampling-based planning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 5113–5120.

[29] X. Meng, N. Ratliff, Y. Xiang, and D. Fox, "Scaling local control to large-scale topological navigation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 672–678.

[30] N. Hirose, F. Xia, R. Martín-Martín, A. Sadeghian, and S. Savarese, "Deep visual mpc-policy learning for navigation," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3184–3191, 2019.

[31] Y. Han, G. Huang, S. Song, L. Yang, H. Wang, and Y. Wang, "Dynamic neural networks: A survey," *TPAMI*, vol. 44, no. 11, pp. 7436–7456, 2021.

[32] L. Del Corro, A. Del Giorno, S. Agarwal, B. Yu, A. Awadallah, and S. Mukherjee, "Skipdecode: Autoregressive skip decoding with batching and caching for efficient llm inference," *arXiv preprint arXiv:2307.02628*, 2023.

[33] D. Raposo, S. Ritter, B. Richards, T. Lillicrap, P. C. Humphreys, and A. Santoro, "Mixture-of-depths: Dynamically allocating compute in transformer-based language models," *arXiv preprint arXiv:2404.02258*, 2024.

[34] M. Elhoushi, A. Shrivastava, D. Liskovich, B. Hosmer, B. Wasti, L. Lai, A. Mahmoud, B. Acun, S. Agarwal, A. Roman *et al.*, "Layer skip: Enabling early exit inference and self-speculative decoding," *arXiv preprint arXiv:2404.16710*, 2024.

[35] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *arXiv preprint arXiv:1308.3432*, 2013.

[36] A. Davis and I. Arel, "Low-rank approximations for conditional feedforward computation in deep neural networks," *arXiv preprint arXiv:1312.4461*, 2013.

[37] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 764–773.

[38] X. Jia, B. De Brabandere, T. Tuytelaars, and L. V. Gool, "Dynamic filter networks," *Advances in neural information processing systems*, vol. 29, 2016.

[39] Y. Wang, R. Huang, S. Song, Z. Huang, and G. Huang, "Not all images are worth 16x16 words: Dynamic transformers for efficient image recognition," *Advances in neural information processing systems*, vol. 34, pp. 11 960–11 973, 2021.

[40] T. Bolukbasi, J. Wang, O. Dekel, and V. Saligrama, "Adaptive neural networks for efficient inference," in *ICML*, 2017.

[41] G. Huang, D. Chen, T. Li, F. Wu, L. Van Der Maaten, and K. Q. Weinberger, "Multi-scale dense networks for resource efficient image classification," *ICLR*, 2018.

[42] Y. Han, D. Han, Z. Liu, Y. Wang, X. Pan, Y. Pu, C. Deng, J. Feng, S. Song, and G. Huang, "Dynamic perceiver for efficient visual recognition," in *ICCV*, 2023.

[43] L. Yang, H. Jiang, R. Cai, Y. Wang, S. Song, G. Huang, and Q. Tian, "Condensenet v2: Sparse feature reactivation for deep networks," in *CVPR*, 2021.

[44] Y. Han, Z. Liu, Z. Yuan, Y. Pu, C. Wang, S. Song, and G. Huang, "Latency-aware unified dynamic networks for efficient image recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–17, 2024.

[45] M. Elbayad, J. Gu, E. Grave, and M. Auli, "Depth-adaptive transformer," *ICLR*, 2020.

[46] J. Xin, R. Tang, Y. Yu, and J. Lin, "Berxit: Early exiting for bert with better fine-tuning and extension to regression," in *ACL*, 2021.

[47] J. Xin, R. Tang, J. Lee, Y. Yu, and J. Lin, "Deebert: Dynamic early exiting for accelerating bert inference," *arXiv preprint arXiv:2004.12993*, 2020.

[48] W. Liu, P. Zhou, Z. Zhao, Z. Wang, H. Deng, and Q. Ju, "Fastbert: a self-distilling bert with adaptive inference time," *arXiv preprint arXiv:2004.02178*, 2020.

[49] S. Mangrulkar, A. MS, and V. Sembium, "Be3r: Bert based early-exit using expert routing," in *KDD*, 2022.

[50] Y. Chen, X. Pan, Y. Li, B. Ding, and J. Zhou, "Ee-llm: Large-scale training and inference of early-exit large language models with 3d parallelism," *arXiv preprint arXiv:2312.04916*, 2023.

[51] T. Schuster, A. Fisch, J. Gupta, M. Dehghani, D. Bahri, V. Tran, Y. Tay, and D. Metzler, "Confident adaptive language modeling," *NeurIPS*, 2022.

[52] Z. Fei, X. Yan, S. Wang, and Q. Tian, "Deecap: Dynamic early exiting for efficient image captioning," in *CVPR*, 2022, pp. 12 216–12 226.

[53] S. Tang, Y. Wang, Z. Kong, T. Zhang, Y. Li, C. Ding, Y. Wang, Y. Liang, and D. Xu, "You need multiple exiting: Dynamic early exiting for accelerating unified vision language model," in *CVPR*, 2023.

[54] A. Ghodrati, B. E. Bejnordi, and A. Habibian, "Frameexit: Conditional early exiting for efficient video recognition," in *CVPR*, 2021.

[55] Z. Ni, Y. Wang, R. Zhou, R. Lu, J. Guo, J. Hu, Z. Liu, Y. Yao, and G. Huang, "Adanat: Exploring adaptive policy for token-based image generation." in *ECCV*, 2024.

[56] C. Fang, C. He, F. Xiao, Y. Zhang, L. Tang, Y. Zhang, K. Li, and X. Li, "Real-world image dehazing with coherence-based label generator and cooperative unfolding network," *arXiv preprint arXiv:2406.07966*, 2024.

[57] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *ICML*, 2019.

[58] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, 2017.

[59] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 16 000–16 009.

[60] D. Shah and S. Levine, "ViKiNG: Vision-Based Kilometer-Scale Navigation with Geographic Hints," in *Proceedings of Robotics: Science and Systems*, 2022.

[61] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.

[62] B. Wheatman and H. Xu, "Packed compressed sparse row: A dynamic graph representation," in *2018 IEEE High Performance extreme Computing Conference (HPEC)*. IEEE, 2018, pp. 1–7.

[63] S. Ruiter, S. Wolfgang, M. Tunnell, T. Triche, E. Carrier, and Z. DeBruine, "Value-compressed sparse column (vcsc): Sparse matrix storage for redundant data," in *2024 Data Compression Conference (DCC)*. IEEE, 2024, pp. 580–580.

[64] L. Yuan, Y. Chen, T. Wang, W. Yu, Y. Shi, Z.-H. Jiang, F. E. Tay, J. Feng, and S. Yan, "Tokens-to-token vit: Training vision transformers from scratch on imagenet," in *ICCV*, 2021.

[65] A. Awadalla, I. Gao, J. Gardner, J. Hessel, Y. Hanafy, W. Zhu, K. Marathe, Y. Bitton, S. Gadre, S. Sagawa *et al.*, "Openflamingo: An open-source framework for training large autoregressive vision-language models," *arXiv preprint arXiv:2308.01390*, 2023.

[66] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," *Advances in neural information processing systems (NeurIPS)*, vol. 36, pp. 34 892–34 916, 2023.

[67] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby *et al.*, "Dinov2: Learning robust visual features without supervision," *Transactions on Machine Learning Research Journal*, pp. 1–31, 2024.

[68] M. Ramamurthy and V. Lakshminarayanan, "Human vision and perception," *Handbook of advanced lighting technology*, pp. 1–23, 2015.

[69] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, 2015.

[70] H. Karnan, A. Nair, X. Xiao, G. Warnell, S. Pirk, A. Toshev, J. Hart, J. Biswas, and P. Stone, "Socially compliant navigation dataset (scand): A large-scale dataset of demonstrations for social navigation," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 807–11 814, 2022.

[71] N. Hirose, D. Shah, A. Sridhar, and S. Levine, "Sacson: Scalable autonomous control for social navigation," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 49–56, 2023.

[72] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[73] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint arXiv:2312.00752*, 2023.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We develop a novel method for efficient visual navigation. Our proposed method effectively reduces the computational and time cost by using the developed feature selector to obtain sparse features and utilizing dynamic early-exiting to skip decoder layers.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: In the conclusion section of the main body, we elaborate on some limitations of our work. These limitations can be further investigated to find suitable solutions.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: In this work, we develop a novel architecture of the visual navigation model. There are no additional newly proposed theories, such as optimization or representation, etc.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: In the experiment section and appendix, we provide detailed settings for training and data processing.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: All data we use in this paper comes from publicly available datasets. Upon the situation of acceptance, we will consider releasing the code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so No is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.

- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: In the experiment section and appendix, we provide detailed settings for training and data processing.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We use the mean value as the justification metric. For each dataset, the final metric value is the mean across all samples. As the parameters of the model are frozen during the inference, therefore, the standard deviation of the same model inference on the same dataset is small enough.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We illustrate all hardware resources we used in the experimental section and appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: Our work conforms to it in every respect.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: Our work is a standard vision task and does not involve societal issues.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: All data and models used by this work are publicly available and tested in many applications. There are no such risks for this paper.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We use publicly available code resources.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: The paper does not release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.

- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: Our work focuses on visual navigation, which does not involve language models.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Appendices of
DynaNav: Dynamic Feature and Layer Selection for Efficient Visual Navigation

A Overview

Section B illustrates the hyperparameters and details for training and inference. Section C shows more experimental results. Section D illustrates more visualized results of the saliency heatmaps on observations and goals.

B Implementation Details

For pre-training, we adopt the same parameter settings as ViNT [5] to ensure a fair comparison. A notable difference, however, is that we directly use the encoded features from EfficientNet-B0 [57] as tokens to the transformer layer, bypassing the MLP projection used by ViNT to reduce the dimensionality from 1280 to 512. This modification helps save computational resources and time. The input RGB images are resized to a resolution of 85×64, with a batch size of 256. Both ViNT and our models are trained for 100 epochs under these conditions. For fine-tuning, we set the learning rate to 1e-4 and train for 80 epochs, deactivating the warm-up stage during this process.

Table 5: Hyperparameter Settings.

Hyperparameter	Value
General	
Train Epochs	100
Fine-tuning Epochs	80
Input Resolution	85×64
Training LR	0.0005
Fine-tuning LR	0.0001
Warmup Epochs	3
Optimizer	AdamW
LR Scheduler	Cosine Annealing
Batch Size	256
λ in loss	0.5
Backbone	
Type	EfficientNet-b0
Hidden Dim	1280
Data	
Length of past frames	5
Length of predicted waypoints	5
Max obs-goal distance(meter)	20
Min obs-goal distance(meter)	0
Transformer Decoder	
Number of layers	4
Attention Heads	4
Bayesian Optimization	
$\text{Sim}(\mathbf{a}_t, \mathbf{a}_t^{\text{gt}})$ Constraint	0.950
$\text{Sim}(\mathbf{w}_t, \mathbf{w}_t^{\text{gt}})$ Constraint	0.960
FLOPs Constraint (10^9)	2.0
Time Constraint (sec)	0.3
Memory Constraint (GB)	14
Optimization Epochs	20
Constraint of Masked Pixels (obs)	2770
Constraint of Masked Pixels (goal)	3400
ξ	[0.8,0.5,1.0]
CARLA Realted	
Max Speed	20km/h
Max Distance	900m
Capture Frequency	4Hz

B.1 Hyper-parameter Setting

The detailed hyper-parameter settings for our training and fine-tuning are in Table 5.

C More Experiment Result

C.1 Results with Mamba Decoder

We also test the performance of the Mamba [73] block. Table 6 illustrates the results of substituting mamba. As the resolution of our feature maps is not large, the advantage of Mamba [73] can not be fully explored. On the other hand, the Mamba's [73] core computing structure - the state space model (SSM) and its high-order recursive calculations- causes its calculation volume to increase rapidly under high-dimensional features. Comparing the $\text{Sim}(\mathbf{w}_t, \mathbf{w}_t^{\text{gt}})$, the performance using Mamba [73] blocks is lower than ours. This may result from the fact that Mamba [73] has advantages on long sequences but may not be able to fully utilize its recursive modeling capabilities on short sequences. Transformer [58] is more suitable for capturing global dependencies. Even if the sequence is short, it can still use the self-attention mechanism to efficiently model the relationship between features.

During the CARLA [72] simulation, the models predictions are normalized to compute the necessary waypoint offsets. These offsets, combined with the vehicles current location, determine the target waypoint. A PID controller is employed to generate control signals based on the target waypoint. To ensure smooth trajectory generation within the CARLA environment, we use an image captured six timestamps ahead of the current observation as the objective, carefully tracking the models progress over each run. As ViNT [5] processes the waypoints in relative coordinates, represented as follows:

$$\mathbf{w}_t = (\mathbf{P}_{t+h} - \mathbf{P}_t) \otimes \mathbf{R}(\theta_t), \tag{13}$$

where $\mathbf{P}_{t+h}$ and $\mathbf{P}_t$ denote the position vectors of the goal and current points in world coordinates, respectively, and $\otimes$ indicates matrix multiplication. θ_t represents the vehicle's yaw, and $\mathbf{R}$ is the rotation matrix. Thus, the final target point is calculated as: $\mathbf{P}_{t+h} = \mathbf{P}_t + \hat{\mathbf{w}}_t \otimes \mathbf{R}(\theta_t)^\top$, where $\hat{\mathbf{w}}_t$ is the predicted waypoint offset.

Table 6: Quantitative Comparison on Benchmarks of ours and Mamba blocks.

Dataset	Method	$\text{Sim}(\mathbf{w}_t, \mathbf{w}_t^{\text{gt}})$	FLOPs(10^9)
RECON [5]	Mamba [73]	95.09	4.41
	Ours	96.53	1.93
Go-Stanford [4]	Mamba [73]	93.34	4.41
	Ours	93.66	1.68
SacSoN [71]	Mamba [73]	92.92	4.41
	Ours	93.72	1.68
SCAND [70]	Mamba [73]	97.28	4.41
	Ours	97.43	1.93

C.2 Goal Image Viewpoint Investigation

In real-world scenarios, goal images often come from diverse sourcessuch as human-captured photosand may not exactly align with the agents ego-centric view. Understanding how such domain and viewpoint differences impact performance is critical.

To investigate this, we conducted additional experiments in CARLA under three challenging conditions:

- **Same location, different angle:** Goal image is taken from the same waypoint but with a camera orientation offset (within $\pm 15^\circ$).

- **Nearby location, same angle:** Image is captured from a nearby position (within 5 meters), keeping the same orientation.
- **Nearby location, different angle:** Goal is from a nearby waypoint (within 5 meters) and a different orientation.

Table 7 illustrates the quantitative results. Our model exhibits graceful degradation as the domain gap increasesi.e., greater viewpoint or positional differences. However, it consistently outperforms the ViNT baseline across all settings, highlighting the robustness and generalization of our approach to goal images with moderate domain shifts.

Table 7: Comparison of our model and ViNT under varying goal image settings in CARLA Scene A

Setting	**Model**	**Success Rate**	**FLOPs** ($\times 10^9$)
Same Position, Same Angle	ViNT	0.724	4.37
	Ours	**0.727**	**1.58**
Nerby Position, Same Angle	ViNT	0.723	4.37
	Ours	**0.725**	**1.58**
Same Position, Different Angle	ViNT	0.694	4.37
	Ours	**0.708**	**1.74**
Nearby Position, Different Angle	ViNT	0.688	4.37
	Ours	**0.691**	**1.79**

C.3 Timestep-wise Consistency

To investigate the potential timestep-wise inconsistency, we conducted an in-depth analysis on a 700-frame trajectory. We segmented the trajectory into 100-frame intervals and computed the average FLOPs and inference time for each segment, comparing our model against the baseline ViNT. As shown in Table 8, our model consistently reduces both computational cost and inference time across all intervals, while maintaining or improving action similarity $\text{Sim}(\mathbf{a}_t, \mathbf{a}_t^{gt})$. In over 96% of the evaluated trajectories, our approach is more efficient than ViNT without any degradation in navigation accuracy.

These results demonstrate that, despite the dynamic nature of early exiting, our model exhibits stable, consistent, and efficient performance over time in practice.

Table 8: Timestep-wise results compared with ViNT

	No. of Frame	100	200	300	400	500	600	700
ViNT	FLOPs (10^9)	4.37	4.37	4.37	4.37	4.37	4.37	4.37
	Avg Time (s)	0.218	0.218	0.218	0.218	0.218	0.218	0.218
	$\text{Sim}(\mathbf{a}_t, \mathbf{a}_t^{gt})$	94.41	94.46	94.49	94.48	94.52	94.51	94.49
Ours	FLOPs (10^9)	2.02	1.95	1.92	1.96	1.85	1.93	1.93
	Avg Time (s)	0.194	0.190	0.189	0.191	0.185	0.190	0.190
	$\text{Sim}(\mathbf{a}_t, \mathbf{a}_t^{gt})$	94.76	94.88	94.92	94.90	94.92	94.92	94.92

C.4 Additional Ablation Study of Constraints

Our adaptive threshold optimization incorporates three constraints designed to jointly enhance model efficiency across FLOPs, time, and memory usage. To evaluate their individual contributions, we conducted an ablation study by removing each constraint separately.

As shown in Table 9, enforcing the FLOPs constraint encourages more frequent layer skipping, effectively reducing inference time and memory consumption. However, removing either the time or memory constraint results in noticeable degradation across all efficiency metrics. This confirms that jointly optimizing all three constraints achieves the best overall performance and balanced resource utilization.

Table 9: Ablation Study On RECON Dataset

Setting	FLOPs (10^9)	Time (s/traj)	Memory (GB)
Ours	1.93	0.228	13.35
w/o FLOPs constrain	2.84	0.291	15.62
w/o Time constrain	2.55	0.273	15.09
w/o Memory constrain	2.16	0.255	14.75

C.5 Study on Robustness

We conducted each navigation trajectory in CARLA 10 times to evaluate the robustness of our method. Table 10 reports the FLOPs, average execution time, and average successful rate for selected trajectories across these runs. As shown, the results exhibit minimal variance, indicating strong consistency and low randomness. This stability is attributed to the synergy between our feature selector and Bayesian optimization, which together enable adaptive yet reliable behavior across diverse scenarios.

Table 10: Results of different separation simulations.

No. of Trajectory	1	2	3	4	5	6	7	8	9	10
FLOPs (10^9)	1.91	1.93	1.92	1.91	1.90	1.92	1.94	1.93	1.93	1.91
Avg Time (s)	0.258	0.260	0.258	0.257	0.260	0.257	0.262	0.260	0.257	0.258
Success Rate	0.725	0.727	0.727	0.726	0.726	0.726	0.728	0.727	0.726	0.727

D More Visualizations

In this section, we added more visualizations of saliency maps. Such a saliency map helps to identify the interest area after being processed by our proposed feature selector. From Figure 7 to Figure 10, we can tell that the region of interest is not always located in the biggest common object between observation and goal images. The model "considers" more spatial information, which results in higher "attention" along the target direction. These findings support our claims in Section 1 that there is redundant information in the observation and goal. In other words, it proves the rationality of using the proposed feature selector to filter features.

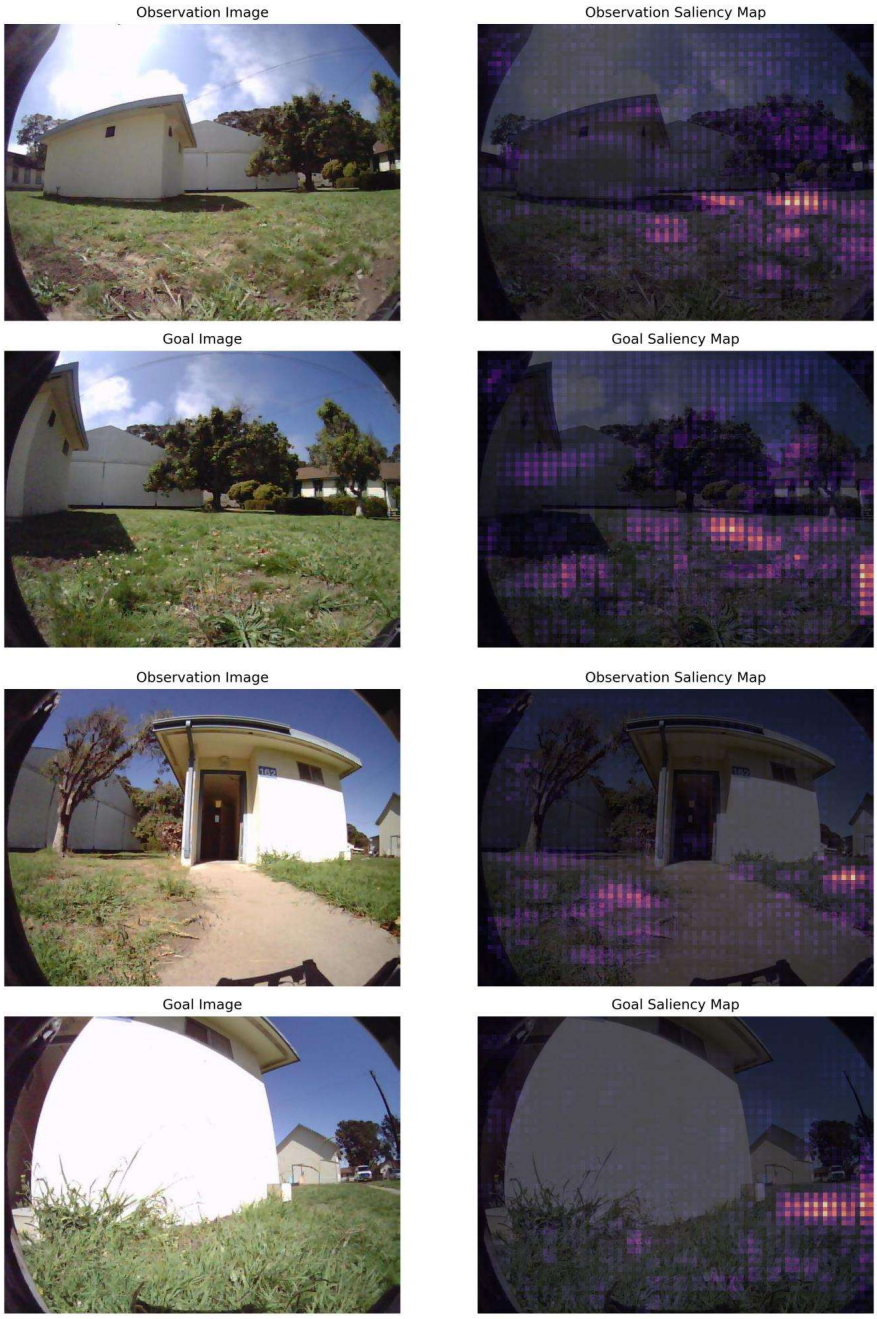

Figure 7: Salieny map of observation and goal images.

Figure 8: Salieny map of observation and goal images.

Figure 9: Salieny map of observation and goal images.

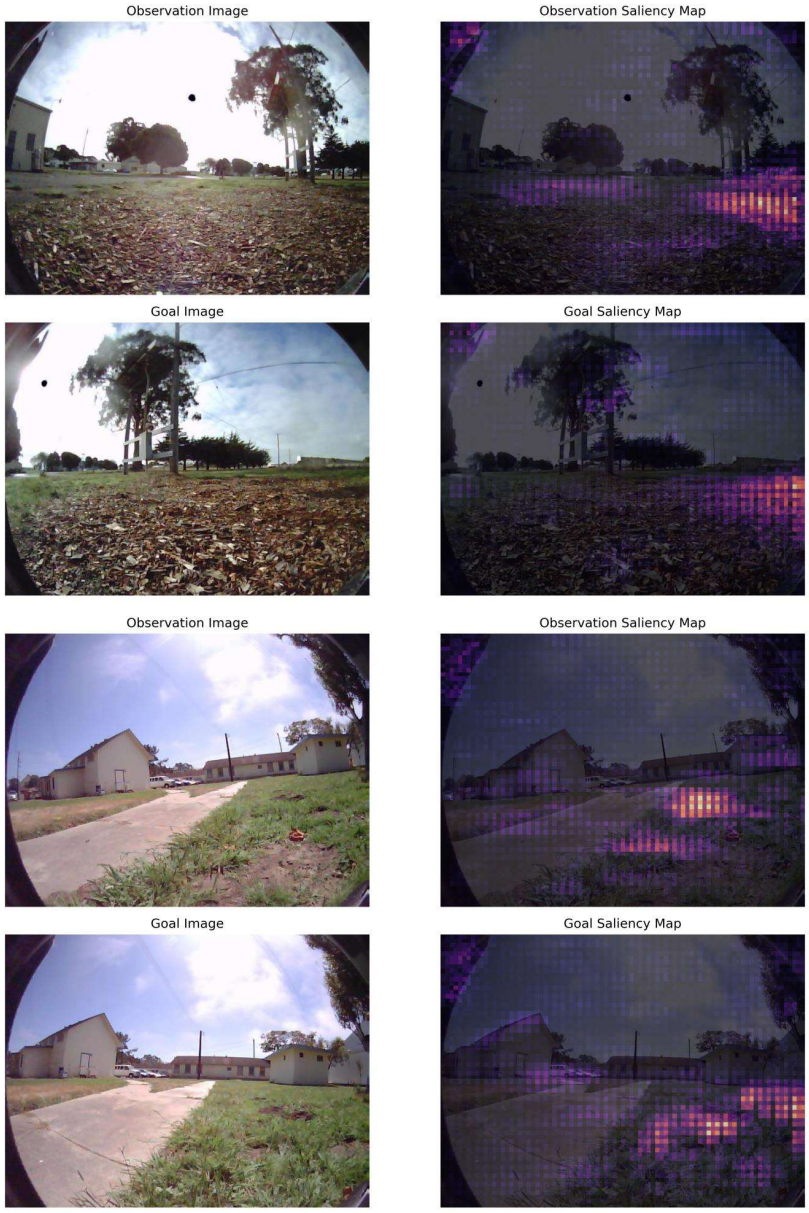

Figure 10: Salieny map of observation and goal images.