DiG-Nav: Learning Differentiable Goal Allocation for Multi-Object Navigation

Anonymous authors
Paper under double-blind review

Abstract

Multi-object navigation requires an agent to prioritize unvisited targets while learning continuous actions under partial observability. We introduce DiG-Nav, which encodes the remaining goal set with masked self-attention and computes a differentiable allocation over goals. The weighted goal representation conditions the action policy, while a short-horizon latent rollout supplies an auxiliary value objective for learning target priorities. Alternating model fitting and policy improvement preserves gradients through imagined actions without optimizing the predictor toward inflated values. We characterize the sensitivity of allocation probabilities and distinguish temperature smoothing from changes in the highest-ranked goal. On Gibson, Matterport3D, and Flightmare, DiG-Nav achieves episode success rates of 83.6%, 67.1%, and 69.5%, respectively, exceeding the strongest compared baseline by 3.1, 5.3, and 4.2 percentage points. Component ablations support the contributions of learned allocation and predictive supervision. Real-world quadrotor demonstrations illustrate continuous execution across multiple targets.

1 Introduction

An agent asked to find several objects must reason about both movement and task order. A nearby object can require a long detour, while a more distant target may lie along a route useful for the remaining mission. Multi-object navigation therefore connects semantic search, memory, and control. MultiON studies navigation through multiple goals (Wani et al., 2020); sequence-agnostic navigation also allows the agent to choose their order (Gireesh et al., 2023). We consider the latter setting: all requested categories must be found within a common episode budget, and their completion order is a decision of the policy.

Existing approaches provide several ways to structure this decision. Explicit exploration and semantic planning use object or room information (Zhou et al., 2023; Zhu et al., 2026), while demand-driven navigation reasons about which objects satisfy a request (Wang et al., 2024). These tasks differ in their goal specifications and learning regimes. We focus on the following question: *does an explicit, differentiable allocation over the remaining goals provide a useful interface between target prioritization and predictive policy learning?*

DiG-Nav represents this allocation as a probability vector. The vector weights goal embeddings supplied to the actor and goal-conditioned values evaluated along an imagined trajectory. Consequently, a predicted improvement can change both the action distribution and the relative weights of candidate goals. The soft weights are a policy representation, not a discrete route or a guarantee that the controller follows the most highly weighted target. This distinction matters when interpreting target-switch diagnostics.

Our formulation builds on attention and differentiable learning (Vaswani et al., 2017; Amos & Kolter, 2017; Hafner et al., 2020). Its contribution is an explicit remaining-goal allocation that couples target prioritization to both action generation and predictive policy learning. First, masked goal-set encoding and a soft allocation provide a permutation-equivariant interface for a variable number of unvisited categories. Second, alternating predictor/critic fitting and actor improvement enables direct and indirect allocation gradients through imagined futures. Third, a sensitivity analysis characterizes probability smoothing without assuming a reduction in goal switching. Experiments compare task completion and path efficiency across ground and aerial settings, with component ablations on Matterport3D.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

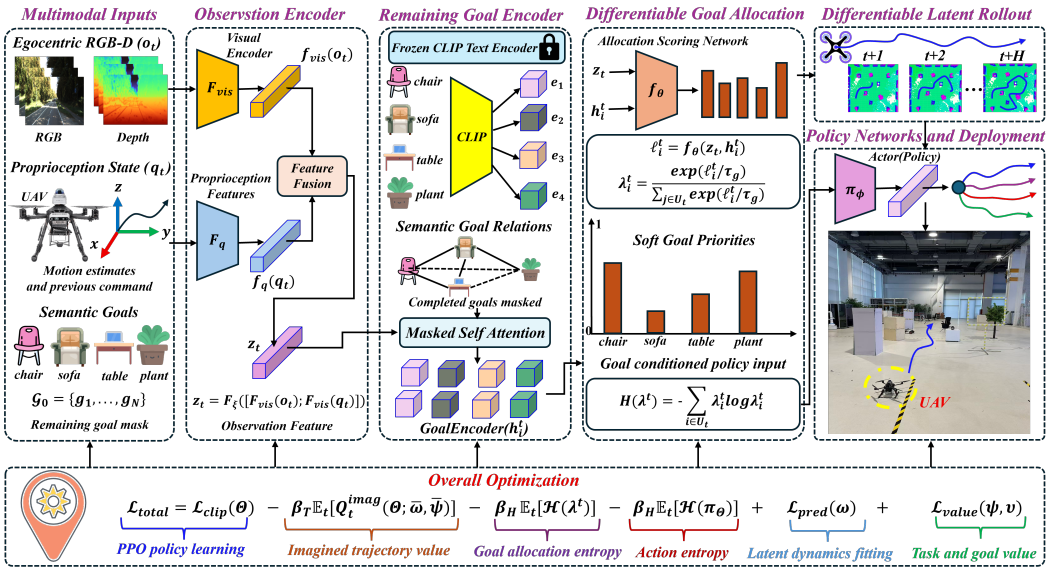


Figure 1: DiG-Nav: differentiable allocation connects remaining-goal encoding, continuous actions, and imagined values. Frozen CLIP features pass through masked self-attention; the scorer combines them with the observation feature to obtain λ^t. The actor uses (z_t, b_t) at deployment. Latent rollouts supply a training auxiliary, not a test-time path selector; map thumbnails illustrate navigation context, not predicted images. The bottom objective groups alternating policy and model/value updates, with predictor and critic weights fixed during policy improvement (Section 3.5).

Figure 1 summarizes the information flow. Observation features and the remaining goal set meet at the allocation scorer. The weighted goal representation conditions the actor directly; imagined trajectories provide an additional training signal through a fitted model and goal critic.

2 Related Work

Tasks with multiple semantic goals. MultiON (Wani et al., 2020) emphasizes semantic map memory for multiple targets. Sequence-Agnostic Multi-Object Navigation (Gireesh et al., 2023) directly addresses learned ordering and is an essential comparison for our problem. MO-DDN (Wang et al., 2024) additionally addresses object attributes and demand satisfaction. More recent work extends multi-object missions in different directions: DM^3-Nav studies decentralized coordination between robots (Kashiri et al., 2026), and LifelongCrossNav maintains persistent semantic memory for ordered queries across floors (Li et al., 2026). Neither setting is interchangeable with single-agent, unordered completion. DiG-Nav focuses on the internal representation of goal priority within a continuous navigation policy.

Semantic and generalist navigation. ZSON transfers multimodal goal embeddings to zero-shot object navigation (Majumdar et al., 2022). ESC converts commonsense information into a training-free navigation procedure (Zhou et al., 2023). STRIVE combines structured environment representations and vision-language model reasoning (Zhu et al., 2026); OctoNav develops a generalist navigation system with multiple stages of training (Gao et al., 2026). MOON is an optimization-driven object-navigation technical report (Nakajima et al., 2025). These are relevant design alternatives, but their published single-goal scores and training procedures are not measurements on our multi-goal task. Multi-goal comparisons therefore concern task-adapted systems (Appendix E.3).

Differentiable prediction and control. Differentiable optimization layers (Amos & Kolter, 2017) and latent imagination for policy learning (Hafner et al., 2020) establish that learning signals can pass through structured computations and predicted futures. DiG-Nav applies this principle to a goal-set allocation interface. For aerial operation, rigid-body dynamics and geometric tracking

control provide a physical reference (Lee et al., 2010); a learned prediction model does not inherit the controller's guarantees. Recent multi-drone transport work uses control barrier functions and explicit tracking-error analysis (Choi et al., 2026). Its safety results concern a different system and do not establish safety of DiG-Nav.

3 Differentiable Goal Allocation

3.1 Task, observations, and policy state

An episode provides a set of distinct target categories $\mathcal{G}_0 = \{g_1, \dots, g_N\}$. The agent must confirm one valid instance of every category, in any order. Let $m_{i,t} \in \{0, 1\}$ indicate whether category i has been confirmed and let $U_t = \{i : m_{i,t} = 0\}$ denote the remaining indices. A simulator-side success predicate checks proximity and visibility; the proximity radius is 0.5 m. The confirmation predicate is shared across compared methods; proximity alone does not establish that a category has been observed. A terminal event occurs when every goal is confirmed, the episode's distance or step budget is exhausted, or a terminal collision or safety intervention occurs.

The environment is a POMDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T}, \Omega, r, \gamma)$. Its hidden state s_t contains scene geometry, object instances, robot configuration, and completion indicators. The observation is $x_t = (o_t, q_t, m_t)$: o_t is an egocentric RGB-D frame, q_t is available proprioception, and m_t is confirmed task progress. The transition kernel $\mathcal{T}$ describes simulator or physical dynamics, and Ω describes sensing. Ground actions are linear and angular velocity commands; aerial actions are three-dimensional velocity and yaw-rate commands. A tracking controller converts the latter to actuator inputs.

For a concrete task objective, let n_t count newly confirmed categories, c_t indicate a collision, and f_t indicate first completion of the entire mission. A sparse reward specification is

$$r_t = \beta_g n_t + \beta_f f_t - \beta_s - \beta_c c_t, \qquad J = \mathbb{E}\left[\sum_{t=0}^{T-1} \gamma^t r_t\right]. \tag{1}$$

The coefficients are nonnegative, with $\beta_g, \beta_f > 0$. The goal bonus rewards individual confirmations, the completion bonus rewards the full mission, and the step and collision costs penalize inefficient or unsafe motion. Appendix B.2 discusses the distinction between this task reward and allocation-dependent distance shaping.

We encode the current observation as

$$z_t = F_\xi\big([F_{\text{vis}}(o_t); F_q(q_t)]\big), \qquad F_{\text{vis}} \text{ is a ResNet-18 encoder} \tag{2}$$

(He et al., 2016). Brackets denote concatenation; F_q is the proprioceptive feature map, which may be the identity. We call z_t an *observation feature*, not a Bayesian belief state: a single frame and proprioception need not summarize the observation history. The completion mask provides task memory, but not a spatial map. A recurrent or map-based extension would require a corresponding architectural description and matched controls.

3.2 Encoding the remaining goal set

The frozen CLIP text encoder (Radford et al., 2021) maps each category name to $e_i \in \mathbb{R}^{d_e}$. A learned projection gives $u_i = W_e e_i \in \mathbb{R}^d$. For one attention head, the remaining goals interact through

$$\alpha_{ij}^t = \frac{\exp((W_Q u_i)^\top (W_K u_j)/\sqrt{d_k})}{\sum_{\ell \in U_t} \exp((W_Q u_i)^\top (W_K u_\ell)/\sqrt{d_k})}, \quad i, j \in U_t, \tag{3}$$

$$h_i^t = \text{LayerNorm}\left(u_i + \text{MLP}\left(\sum_{j \in U_t} \alpha_{ij}^t W_V u_j\right)\right). \tag{4}$$

The multihead version concatenates head outputs before the MLP. The matrices have compatible dimensions so that the residual in Equation (4) is d-dimensional. The graph is fully connected over unvisited goals; α_{ij}^t is a learned directed weight, not a thresholded spatial edge. Completed nodes are excluded as both queries and keys. These inputs encode category semantics and remaining-set context. They do not by themselves identify object locations, distances, or a scene topology. Scene information enters through z_t in the allocation scorer.

3.3 SOFT ALLOCATION AND CONTINUOUS ACTIONS

A scoring network maps the current feature and each goal embedding to a scalar:

$$\ell_i^t = f_\theta(z_t, h_i^t), \qquad \lambda_i^t = \frac{\exp(\ell_i^t/\tau_g)}{\sum_{j\in U_t}\exp(\ell_j^t/\tau_g)}, \qquad b_t = \sum_{i\in U_t}\lambda_i^t h_i^t. \tag{5}$$

Here $\lambda_i^t = 0$ for completed goals, and the procedure terminates before computing a softmax if U_t is empty. The positive temperature is $\tau_g = \mathrm{softplus}(\eta_g) + \tau_{\min}$, where η_g is learned and $\tau_{\min} > 0$. The policy takes (z_t, b_t) as input. Thus changing the allocation can change the executed action without choosing a discrete intermediate goal. The highest-weight goal is only a diagnostic unless an explicit hard-selection controller is introduced.

The allocation entropy is

$$\mathcal{H}(\lambda^t) = -\sum_{i\in U_t}\lambda_i^t \log \lambda_i^t. \tag{6}$$

Its coefficient β_H is a training-loss weight. It is *not* part of the denominator in Equation (5). Entropy rewards spread in the allocation; whether that spread improves task completion is an empirical question.

3.4 PREDICTION AND TRAJECTORY SCORING

Let g_ω be a differentiable predictor in the same feature coordinates as the actor, with predicted state initialized as $\tilde{z}_0 = z_t$. A rollout holds the current goal embeddings and allocation fixed for H imagined steps. For independent $\epsilon_k \sim \mathcal{N}(0, I)$,

$$\tilde{a}_k = \mu_\phi(\tilde{z}_k, b_t) + \sigma_\phi(\tilde{z}_k, b_t) \odot \epsilon_k, \tag{7}$$

$$\tilde{z}_{k+1} = g_\omega(\tilde{z}_k, \tilde{a}_k), \qquad Q_t^{\mathrm{imag}} = \sum_{k=1}^{H}\gamma^k \sum_{i\in U_t}\lambda_i^t V_\psi(\tilde{z}_k, h_i^t). \tag{8}$$

V_ψ is a scalar goal-conditioned value predictor. Its architecture consists of separate state/goal projections, concatenation, an MLP, and a scalar head; Imagined goal completions are not used to change U_t in this specification. The resulting score is an auxiliary lookahead objective, not an exact Bellman expansion of the multi-goal return. Summing state values at several horizons can count overlapping future rewards repeatedly. We use this objective during training. At deployment, the actor generates a command directly from the observed feature and current allocation; no imagined trajectory is selected for execution.

In the aerial setting, a physical consistency term can constrain a decoded motion state (p, v, R, ω_b) against rigid-body dynamics (Lee et al., 2010):

$$\dot{p} = v, \quad \dot{v} = -ge_3 + (f/m)Re_3, \quad \dot{R} = R\widehat{\omega_b}, \quad J_b\dot{\omega}_b = M - \omega_b \times J_b\omega_b. \tag{9}$$

Here f is scalar thrust, M is body torque, m is mass, J_b is inertia, e_3 is the upward vertical unit vector, and $R \in \mathrm{SO}(3)$ maps body to world coordinates. These inputs differ from the policy's velocity command: connecting the two requires a controller model and a decoder from latent features to motion coordinates. Equation (9) provides a physical reference for this interface; our latent predictor does not impose full six-degree-of-freedom dynamics as a hard constraint. No manifold-preservation, collision-avoidance, or closed-loop stability guarantee follows from using these equations as an inductive bias.

3.5 TRAINING AND GRADIENT ROUTING

We use PPO (Schulman et al., 2017) for learning from executed transitions. Let Θ collect actor, scorer, learned observation/goal encoders, and temperature parameters. For stored observations and masks, the likelihood ratio must recompute *both* the feature and allocation under the new and behavior policies:

$$\rho_t(\Theta) = \frac{\pi_\Theta(a_t \mid x_t, \mathcal{G}_0)}{\pi_{\Theta_{\mathrm{old}}}(a_t \mid x_t, \mathcal{G}_0)}, \qquad L_{\mathrm{clip}} = -\mathbb{E}_t \min\left\{\rho_t\hat{A}_t,\ \mathrm{clip}(\rho_t, 1-\epsilon, 1+\epsilon)\hat{A}_t\right\}. \tag{10}$$

$\hat{A}_t$ is an advantage estimate from executed task rewards and a task-value baseline. The negative sign makes this a minimization objective.

The update separates model/value fitting from policy improvement. Fit g_ω to observed transitions and fit the task and goal-conditioned critics to observed returns (Appendix B). During the actor and allocation update, use fixed predictor and critic parameters $\bar{\omega}, \bar{\psi}$, while preserving derivatives with respect to their inputs:

$$L_{\text{actor}}(\Theta) = L_{\text{clip}} - \beta_T \mathbb{E}_t Q_t^{\text{imag}}(\Theta; \bar{\omega}, \bar{\psi}) - \beta_H \mathbb{E}_t \mathcal{H}(\lambda^t) - \beta_\pi \mathbb{E}_t \mathcal{H}(\pi_\Theta). \tag{11}$$

$\beta_T, \beta_H, \beta_\pi \geq 0$ control lookahead, goal entropy, and action entropy, respectively. Fixing model parameters does not detach the imagined states or actions. It prevents the auxiliary maximization from directly rewarding the critic for increasing its outputs or the dynamics model for inventing high-value transitions. The alternating schedule separates supervised prediction from policy improvement while retaining the pathwise derivatives needed to learn allocation.

The figure summarizes these blocks as

$$L_{\text{total}} = L_{\text{actor}}(\Theta; \bar{\omega}, \bar{\psi}) + L_{\text{pred}}(\omega) + L_{\text{value}}(\psi, \nu). \tag{12}$$

This is shorthand for alternating updates, not an unrestricted joint gradient step. Model and value losses use detached encoder features; the actor loss retains input derivatives through fixed model and critic weights. Algorithm 1 specifies this distinction operationally.

For fixed rollout states and embeddings, let $v_i = \sum_{k=1}^{H} \gamma^k V_{\bar{\psi}}(\tilde{z}_k, h_i^t)$. The direct allocation derivative is

$$\left.\frac{\partial Q_t^{\text{imag}}}{\partial \ell_j^t}\right|_{\tilde{z},h} = \frac{\lambda_j^t}{\tau_g}\left(v_j - \sum_{i \in U_t} \lambda_i^t v_i\right). \tag{13}$$

There is also an indirect derivative through b_t, reparameterized actions, and predicted states. Equation (13) isolates why explicit allocation can receive a signal about relative predicted value. Appendix B.3 gives the full chain rule. Differentiability of this training computation does not require the real sensor, detector, environment, or deployed low-level controller to be differentiable.

4 Allocation Sensitivity and Its Limits

For unvisited goals i, j, define $\Delta_{ij}^t = \ell_i^t - \ell_j^t$ and $p_{ij}^t = \lambda_i^t/(\lambda_i^t + \lambda_j^t)$. The following standard softmax properties describe the implemented allocation.

Proposition 1 (Probability sensitivity at a fixed temperature)**.** *For $\tau_g > 0$ and a fixed active goal set,*

$$p_{ij}^t = \sigma(\Delta_{ij}^t/\tau_g), \quad \left|\log \frac{\lambda_i^t}{\lambda_j^t}\right| = \frac{|\Delta_{ij}^t|}{\tau_g}, \quad |2p_{ij}^t - 1| = \tanh\left(\frac{|\Delta_{ij}^t|}{2\tau_g}\right). \tag{14}$$

If both goals remain active, the temperature is fixed across steps, and $|\Delta_{ij}^{t+1} - \Delta_{ij}^t| \leq \varepsilon_t$, then

$$|p_{ij}^{t+1} - p_{ij}^t| \leq \frac{\varepsilon_t}{4\tau_g}. \tag{15}$$

A pairwise ranking reversal is impossible if $|\Delta_{ij}^t| > \varepsilon_t$.

Here $\sigma(u) = (1 + e^{-u})^{-1}$. The identities follow by dividing softmax weights; the bound follows from $\sigma'(u) \leq 1/4$. Appendix C gives a proof and the variable-temperature case. Neither result contains β_H.

Smoothing is not a switching guarantee. For every positive scalar temperature, $\arg\max_i \lambda_i^t = \arg\max_i \ell_i^t$, with a fixed tie rule. A two-goal gap $\Delta^t = (-1)^t\delta$, $\delta > 0$, produces a ranking change at every step for every temperature. Raising τ_g reduces the probability difference but leaves the rankings unchanged. Reduced switching after retraining must therefore be established from score trajectories and completion events. Entropy regularization may change those trajectories through learning; the proposition does not prove that it will do so.

5 EXPERIMENTS

5.1 EVALUATION DESIGN

We evaluate three aspects of the allocation interface: completion of all requested categories, path efficiency for the unordered task, and the contributions of allocation and predictive supervision. The simulation comparison includes semantic-search systems and aerial planning systems. SAM (Gireesh et al., 2023) directly addresses sequence-agnostic multi-object navigation, while PONI (Ramakrishnan et al., 2022) represents potential-based semantic search. ESC, MOON, OctoNav, and STRIVE provide additional object-navigation comparisons. EGO-Planner and FUEL represent local trajectory optimization and frontier-based aerial exploration, respectively. Appendix E.3 explains their task roles.

These comparisons assess complete navigation systems, whose perception, training objectives, and planning modules can differ. We therefore use within-model ablations to examine individual components of DiG-Nav. Uniform allocation tests learned prioritization, removal of the imagined-value objective tests predictive supervision, and removal of allocation entropy tests its regularizing effect. Changing the rollout horizon tests the amount of predicted context used by the auxiliary objective.

5.2 ENVIRONMENTS, METRICS, AND UNCERTAINTY

Gibson (Xia et al., 2018) and Matterport3D (Chang et al., 2017) provide indoor scenes; Flightmare (Song et al., 2021) is an aerial simulator, not a standardized multi-goal episode set. We use $N = 3$ for Gibson and $N = 5$ for MP3D and Flightmare. DiG-Nav uses a 50M-step PPO training budget, five seeds, and a default rollout horizon of $H = 8$. Ground policies output linear and angular velocity; aerial policies output three-dimensional velocity and yaw rate. The remaining-goal interface is shared across these action spaces.

For episode e, let $y_{e,i}$ indicate confirmation of category i, $S_e = \prod_i y_{e,i}$, L_e be the executed path length, and d_e^* the shortest valid path completing the same unordered task:

$$\mathrm{PSR} = \frac{1}{E}\sum_{e=1}^{E}\frac{1}{N_e}\sum_{i} y_{e,i}, \quad \mathrm{ESR} = \frac{1}{E}\sum_{e=1}^{E} S_e, \quad \mathrm{mSPL} = \frac{1}{E}\sum_{e=1}^{E} S_e \frac{d_e^*}{\max(d_e^*, L_e)}. \tag{16}$$

The oracle must use the evaluator's valid instances, visibility, and success regions. Held–Karp dynamic programming (Held & Karp, 1962) is exact for fixed service vertices. Category goals with multiple instances or service poses require a richer oracle (Appendix D). The all-goal success factor makes mSPL sensitive to complete missions, whereas PSR also credits partial completion. All three metrics lie in $[0, 1]$ and are reported as percentages.

Let $i_t^* = \arg\max_{i \in U_t} \lambda_i^t$ with a deterministic tie rule. Unnecessary switches are

$$C_e = \sum_{t=0}^{T_e-2} \mathbb{I}\{i_t^* \in U_{t+1}\}\mathbb{I}\{i_{t+1}^* \neq i_t^*\}, \tag{17}$$

on nonterminal transitions with nonempty active sets. Completing a goal does not count as switching away from it. Normalizing C_e by the number of decisions separates switching frequency from episode duration. These are allocation diagnostics, not direct measurements of a discrete controller's commitments.

Table 1 reports means and standard errors over five seeds, with $\mathrm{SE} = s/\sqrt{K}$ for K independent runs. Episode matching and scene clustering affect uncertainty in pairwise comparisons; pooled episodes are not independent training replicates. We interpret the reported differences descriptively and do not infer statistical significance from error-bar overlap. Appendix E discusses the corresponding aggregation units.

5.3 SIMULATION RESULTS

Table 1 compares category completion, full-episode success, and path efficiency. DiG-Nav achieves the highest mean on all three metrics in each setting. Compared with STRIVE, the strongest baseline

in the table, ESR increases by 3.1 percentage points on Gibson, 5.3 on MP3D, and 4.2 on Flightmare. The corresponding mSPL improvements are 2.6, 3.1, and 2.1 points. The gains therefore extend beyond partial category completion to both completion of the entire mission and the efficiency of successful routes.

Table 1: Multi-object navigation in simulation. Values are mean ± standard error over five seeds. All metrics are percentages; higher is better. Baselines are evaluated in the indicated multi-goal setting.

Setting	Method	PSR (%)	ESR (%)	mSPL (%)
Gibson (N=3, Ground)	ESC (Zhou et al., 2023)	84.2 ± 0.7	71.1 ± 1.2	62.6 ± 1.0
	SAM (Gireesh et al., 2023)	86.8 ± 0.5	74.2 ± 0.8	65.3 ± 0.9
	PONI (Ramakrishnan et al., 2022)	87.1 ± 1.2	75.0 ± 0.7	66.1 ± 0.5
	MOON (Nakajima et al., 2025)	88.4 ± 0.4	77.6 ± 1.2	68.1 ± 0.8
	OctoNav (Gao et al., 2026)	89.3 ± 0.6	79.0 ± 0.6	69.4 ± 1.1
	STRIVE (Zhu et al., 2026)	90.1 ± 1.3	80.5 ± 0.5	70.6 ± 1.3
	DiG-Nav (Ours)	$\mathbf{92.1 \pm 1.3}$	$\mathbf{83.6 \pm 0.4}$	$\mathbf{73.2 \pm 0.8}$
Matterport3D (N=5, Ground)	ESC (Zhou et al., 2023)	69.1 ± 0.8	47.5 ± 1.3	41.8 ± 1.1
	SAM (Gireesh et al., 2023)	72.1 ± 0.7	50.8 ± 1.0	48.5 ± 0.8
	PONI (Ramakrishnan et al., 2022)	74.5 ± 1.3	54.2 ± 0.9	51.0 ± 0.7
	MOON (Nakajima et al., 2025)	76.7 ± 0.8	57.4 ± 0.8	52.9 ± 1.2
	OctoNav (Gao et al., 2026)	78.4 ± 0.5	59.6 ± 0.7	54.3 ± 0.4
	STRIVE (Zhu et al., 2026)	80.2 ± 0.4	61.8 ± 1.3	55.7 ± 0.5
	DiG-Nav (Ours)	$\mathbf{83.5 \pm 0.4}$	$\mathbf{67.1 \pm 0.7}$	$\mathbf{58.8 \pm 0.3}$
Flightmare-UAV (N=5, Aerial)	PONI (Ramakrishnan et al., 2022)	73.4 ± 0.9	48.2 ± 1.4	44.5 ± 1.2
	EGO-Planner (Zhou et al., 2021b)	76.2 ± 0.8	55.0 ± 1.5	54.7 ± 0.9
	FUEL (Zhou et al., 2021a)	79.8 ± 0.6	58.4 ± 1.1	57.1 ± 0.6
	MOON (Nakajima et al., 2025)	80.9 ± 0.3	60.2 ± 1.0	58.5 ± 0.8
	OctoNav (Gao et al., 2026)	82.7 ± 1.2	63.1 ± 0.9	60.4 ± 0.5
	STRIVE (Zhu et al., 2026)	84.1 ± 0.5	65.3 ± 0.7	61.7 ± 0.7
	DiG-Nav (Ours)	$\mathbf{86.2 \pm 0.5}$	$\mathbf{69.5 \pm 0.9}$	$\mathbf{63.8 \pm 1.2}$

PSR measures category completion, ESR requires all categories, and mSPL weights complete success by path efficiency. Citations identify the base methods; Appendix E.3 describes their roles in the multi-goal task. Gray shading highlights DiG-Nav.

The gap between PSR and ESR captures a remaining challenge: finding some categories does not ensure completion of the whole set. For DiG-Nav this gap is 8.5 points on Gibson, 16.4 on MP3D, and 16.7 on Flightmare. The larger gaps in the five-goal settings emphasize the importance of retaining progress across target visits. Because scene distributions and action spaces also differ, these cross-setting differences do not isolate goal count as a cause.

The aerial comparison additionally separates semantic search from local motion generation. EGO-Planner and FUEL are useful planning references, but their base algorithms do not themselves define category-level allocation. Their rows measure the corresponding complete navigation stacks. Accordingly, we attribute system-level differences to the evaluated configurations and use the controlled component changes below for a narrower analysis.

Figure 2 shows performance over training on MP3D and the final ESR and mSPL values. DiG-Nav maintains an advantage in both metrics at the end of training, consistent with Table 1. The curves complement the final comparison by showing how task completion and path efficiency develop with additional environment interaction.

5.4 Component Ablations

Table 2 examines the allocation, imagined-value objective, entropy term, and prediction horizon on MP3D. Uniform allocation yields the largest decrease: ESR falls from 67.1% to 53.4% and mSPL from 58.8% to 48.2%. Learning unequal priorities among unvisited categories is therefore useful in

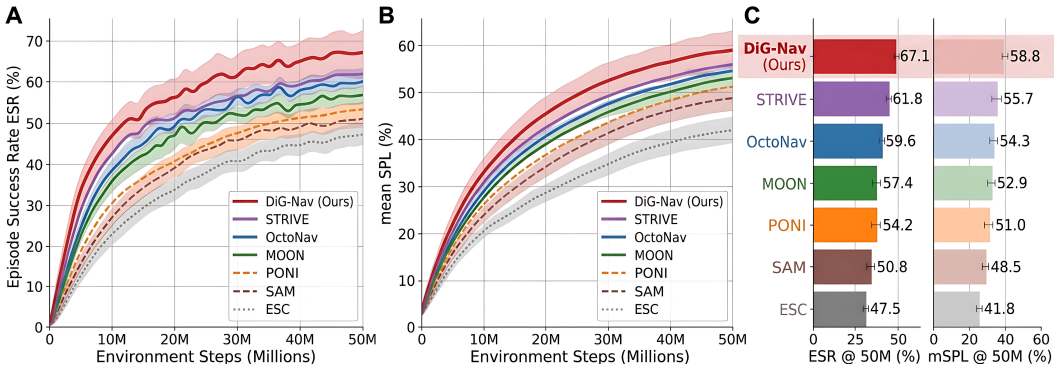


Figure 2: Training performance on MP3D ($N = 5$). (A) Episode success rate and (B) mSPL as a function of environment steps. (C) Final ESR and mSPL after 50M steps. Method names denote the multi-goal systems compared in Table 1; higher values indicate better performance.

this setting. Removing the imagined-value objective reduces ESR by 11.0 points, while removing allocation entropy reduces it by 7.2 points.

Table 2: Component ablations on MP3D ($N = 5$). Subscripts give the decrease from full DiG-Nav in percentage points. All metrics are percentages; higher is better.

Variant	Component changed	PSR	ESR	mSPL
Full DiG-Nav ($H = 8$)	Allocation, prediction, and entropy	**83.5**	**67.1**	**58.8**
(A) Uniform allocation	$\lambda_i = 1/\lvert U_t \rvert$ on remaining goals	$75.2_{(\downarrow 8.3)}$	$53.4_{(\downarrow 13.7)}$	$48.2_{(\downarrow 10.6)}$
(B) No trajectory value	$\beta_T = 0$	$76.8_{(\downarrow 6.7)}$	$56.1_{(\downarrow 11.0)}$	$50.5_{(\downarrow 8.3)}$
(C) No allocation entropy	$\beta_H = 0$; no entropy objective	$78.4_{(\downarrow 5.1)}$	$59.9_{(\downarrow 7.2)}$	$52.3_{(\downarrow 6.5)}$
(D) Shorter horizon	$H = 4$	$79.5_{(\downarrow 4.0)}$	$63.0_{(\downarrow 4.1)}$	$55.1_{(\downarrow 3.7)}$
(E) Longer horizon	$H = 16$	$81.1_{(\downarrow 2.4)}$	$64.2_{(\downarrow 2.9)}$	$56.4_{(\downarrow 2.4)}$

Each variant changes the indicated component of full DiG-Nav. The default prediction horizon is $H = 8$; uniform allocation is applied only to unvisited goals.

The horizon comparison is nonmonotonic: $H = 4$ gives 63.0% ESR, $H = 8$ gives 67.1%, and $H = 16$ gives 64.2%. The intermediate horizon performs best among the tested choices. A longer rollout offers more predicted context, but can also expose the actor to accumulated model error; the table measures the performance tradeoff without directly measuring prediction error.

These ablations establish component-level associations within the evaluated architecture. They do not separate every mechanism: removing the auxiliary changes both supervision and the gradient path, while removing entropy can change score trajectories without directly constraining argmax switches. A capacity-matched PPO control and a detached-imagination control would further distinguish additional model capacity from allocation gradients. The probability bound in Section 4 is consistent with this interpretation and does not convert the ESR changes into a switching result.

5.5 Physical deployment and RViz demonstrations

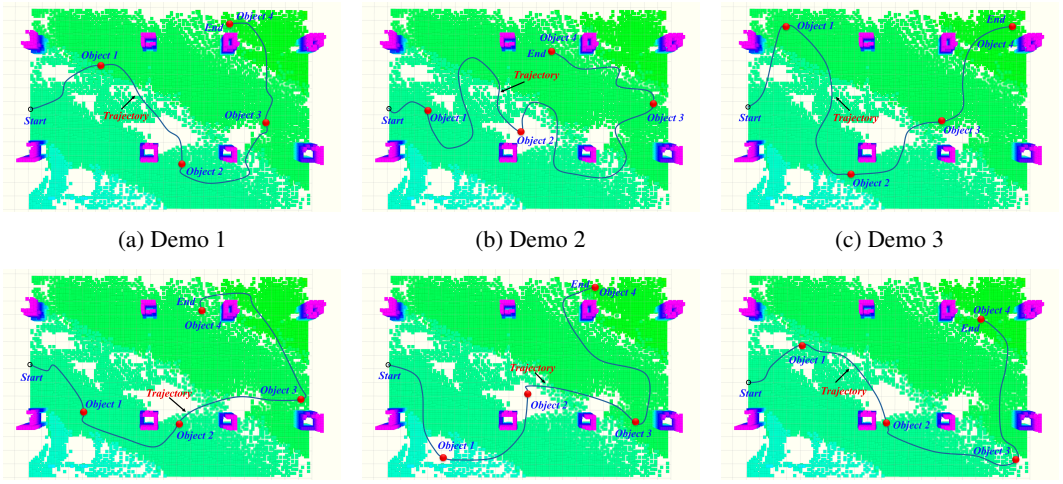
Figure 3 shows the quadrotor platform, an indoor test site, and an in-flight deployment view. The platform uses a Pixhawk 4, Jetson Orin NX, and RealSense D435i; a low-level controller tracks the navigation policy's velocity commands.

Six missions from 100 real-world trials. Figure 4 presents six selected missions in RViz, with four target points per displayed mission. Red circles indicate target points and the blue trace is the executed flight trajectory over the displayed spatial reconstruction. The traces illustrate how several target visits are connected during continuous flight. RViz is a visualization interface; its map display does not imply that the policy consumes a reconstructed map.



(a) Quadrotor platform (b) Test site (c) In-flight execution

Figure 3: Real-world deployment: (a) the onboard quadrotor platform, (b) an indoor test site, and (c) an in-flight deployment view. These photographs document the hardware and operating setting; Figure 4 shows six recorded mission visualizations.



(a) Demo 1 (b) Demo 2 (c) Demo 3

(d) Demo 4 (e) Demo 5 (f) Demo 6

Figure 4: Six real-world quadrotor missions from a 100-trial campaign. Red circles mark four targets per mission; blue curves show executed trajectories. Panels illustrate route geometry and are indexed by demonstration.

These qualitative demonstrations show onboard execution across multiple targets, rather than aggregate success or collision clearance over all 100 trials. Appendix E discusses their interpretation.

6 Limitations and Conclusion

The formulation is limited by observation aliasing, category-only goal features, prediction error, and the gap between high-level commands and modeled physics. Fixed goal masks inside imagined rollouts miss predicted task completion. Dense attention costs $O(N^2d + Nd^2)$; scoring N goals at H rollout states adds an HN factor to critic evaluation. Shared goal-wise networks need not increase parameter count with N. This analysis characterizes scaling, rather than device-level latency.

DiG-Nav connects remaining-goal representations to continuous actions and imagined values through a differentiable allocation. Its analysis distinguishes probability sensitivity from discrete ranking changes. Across three simulation settings, the method improves completion and path efficiency over the compared systems; MP3D ablations support the roles of learned allocation, prediction, and entropy regularization, with direct actor execution at deployment. Extending this evaluation to larger goal sets and open-vocabulary benchmarks such as HM3D-OVON (Yokoyama et al., 2024) remains future work.

AI USE STATEMENT

Generative AI tools were used only for language polishing, grammar checking, and wording improvements, as well as assistance with understanding code and troubleshooting minor issues. The authors take full responsibility for the content of this manuscript.

ETHICS STATEMENT

Physical navigation can cause collisions and capture incidental visual information. Deployment requires operator supervision and protection of recorded data. The allocation and prediction modules provide no formal safety guarantee. Safety interventions must be recorded rather than excluded from evaluation.

REFERENCES

Brandon Amos and J. Zico Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 136–145. PMLR, 2017.

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Nießner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. In *International Conference on 3D Vision*, 2017.

Jaeyoun Choi, Oswin So, Songyuan Zhang, Cooper Taylor, and Chuchu Fan. Safe and scalable multi-drone payload transport via CBF-based reinforcement learning with zero-shot sim-to-real transfer. *IEEE Robotics and Automation Letters*, 2026. doi: 10.1109/LRA.2026.3715346. Early access; author manuscript arXiv:2607.20665.

Chen Gao, Liankai Jin, Xingyu Peng, Jiazhao Zhang, Yue Deng, Annan Li, He Wang, and Si Liu. OctoNav: Towards generalist embodied navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 40074–40084, 2026.

Nandiraju Gireesh, Ayush Agrawal, Ahana Datta, Snehasis Banerjee, Mohan Sridharan, Brojeshwar Bhowmick, and Madhava Krishna. Sequence-agnostic multi-object navigation. In *IEEE International Conference on Robotics and Automation*, pp. 9573–9579, 2023. doi: 10.1109/ICRA48891.2023.10160259.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Michael Held and Richard M. Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, 10(1):196–210, 1962. doi: 10.1137/0110015.

Amin Kashiri, Atharva Jamsandekar, and Yasin Yazıcıoğlu. DM^3-Nav: Decentralized multi-agent multimodal multi-object semantic navigation. *arXiv preprint arXiv:2604.22014*, 2026. Accepted to IROS 2026, according to the authors' arXiv record.

Taeyoung Lee, Melvin Leok, and N. Harris McClamroch. Geometric tracking control of a quadrotor UAV on SE(3). In *49th IEEE Conference on Decision and Control*, pp. 5420–5425, 2010. doi: 10.1109/CDC.2010.5717652.

Zehui Li, Zihao Sun, Jiawei Xu, Zheqi He, Xiaoqiang Zhang, Jing-Shu Zheng, Lu Liu, Dahui Gao, and Xiuwan Chen. LifelongCrossNav: Persistent 3D semantic memory for cross-floor multi-object navigation. *arXiv preprint arXiv:2608.07079*, 2026.

Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. ZSON: Zero-shot object-goal navigation using multimodal goal embeddings. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Daigo Nakajima, Kanji Tanaka, Daiki Iwata, and Kouki Terashima. MOON: Multi-objective optimization-driven object-goal navigation using a variable-horizon set-orienteering planner. *arXiv preprint arXiv:2505.12752*, 2025. Technical report; version 3 revised January 2026.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021.

Santhosh Kumar Ramakrishnan, Devendra Singh Chaplot, Ziad Al-Halah, Jitendra Malik, and Kristen Grauman. PONI: Potential functions for ObjectGoal navigation with interaction-free learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18890–18900, 2022.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Yunlong Song, Selim Naji, Elia Kaufmann, Antonio Loquercio, and Davide Scaramuzza. Flightmare: A flexible quadrotor simulator. In *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pp. 1147–1157. PMLR, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Hongcheng Wang, Peiqi Liu, Wenzhe Cai, Mingdong Wu, Zhengyu Qian, and Hao Dong. MO-DDN: A coarse-to-fine attribute-based exploration agent for multi-object demand-driven navigation. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Saim Wani, Shivansh Patel, Unnat Jain, Angel X. Chang, and Manolis Savva. MultiON: Benchmarking semantic map memory using multi-object navigation. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Fei Xia, Amir R. Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson Env: Real-world perception for embodied agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9068–9079, 2018.

Naoki Yokoyama, Ram Ramrakhya, Abhishek Das, Dhruv Batra, and Sehoon Ha. HM3D-OVON: A dataset and benchmark for open-vocabulary object goal navigation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2024.

Boyu Zhou, Yichen Zhang, Xinyi Chen, and Shaojie Shen. FUEL: Fast UAV exploration using incremental frontier structure and hierarchical planning. *IEEE Robotics and Automation Letters*, 6 (2):779–786, 2021a.

Kaiwen Zhou, Kaizhi Zheng, Connor Pryor, Yilin Shen, Hongxia Jin, Lise Getoor, and Xin Eric Wang. ESC: Exploration with soft commonsense constraints for zero-shot object navigation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 42829–42842. PMLR, 2023.

Xin Zhou, Zhepei Wang, Hongkai Ye, Chao Xu, and Fei Gao. EGO-Planner: An ESDF-free gradient-based local planner for quadrotors. *IEEE Robotics and Automation Letters*, 6(2):478–485, 2021b. doi: 10.1109/LRA.2020.3047728.

Haokun Zhu, Zongtai Li, Zhixuan Liu, Wenshan Wang, Ji Zhang, Jonathan Francis, and Jean Oh. STRIVE: Structured representation integrating VLM reasoning for efficient object navigation. In *IEEE International Conference on Robotics and Automation*, 2026.

A Notation and Architecture

Table 3 defines the main symbols, and Table 4 summarizes the architecture and evaluation settings. Predictor fitting, critic fitting, and actor improvement use distinct objectives.

Table 3: Notation. Parameters for the predictor and critic are kept distinct.

Symbol	Meaning
$s_t, x_t = (o_t, q_t, m_t)$	Hidden environment state and available observation
$\mathcal{G}_0, U_t, N$	Initial category set, remaining indices, initial goal count
z_t, e_i, h_i^t	Observation feature, frozen text feature, contextual goal feature
$\ell_i^t, \lambda_i^t, b_t$	Allocation score, soft weight, weighted goal representation
$\tau_g, \eta_g, \tau_{\min}$	Positive temperature, its free parameter, positive floor
$\Theta, \phi, \theta, \xi$	Policy-side parameters; actor, scorer, observation encoder
ω, ψ, ν	Dynamics predictor, goal critic, task critic parameters
$H, \tilde{z}_k, \tilde{a}_k$	Imagined horizon, predicted feature, reparameterized action
V_ψ, U_ν	Goal-conditioned auxiliary critic and task-return baseline
$\beta_T, \beta_H, \beta_\pi$	Imagined-value, allocation-entropy, action-entropy weights
d_e^*, L_e, S_e	Episode oracle length, executed length, all-goal success

Table 4: Architecture and evaluation settings.

Component	Specification
Observations	Egocentric RGB-D, proprioception, and goal-completion mask
Visual features	ResNet-18 observation encoder
Goal features	Frozen CLIP text features with learned projection
Goal interaction	Fully connected self-attention over unvisited categories
Allocation	Masked softmax with learned positive temperature
Actor input	Observation feature and weighted goal representation
Prediction	Differentiable latent dynamics; default horizon $H = 8$
Optimization	PPO with imagined value and separate goal/action entropy terms
Training budget	50M environment steps; five seeds
Goal counts	Gibson: $N = 3$; MP3D and Flightmare: $N = 5$
Confirmation	Proximity and visibility; 0.5 m proximity radius
Physical platform	Pixhawk 4, Jetson Orin NX, RealSense D435i
Physical demonstrations	Six four-target missions from 100 trials

The predictor parameters ω and goal-critic parameters ψ are distinct. The coefficients β_T and β_H weight imagined value and allocation entropy, respectively; neither is the softmax temperature τ_g.

B Training and Gradient Computation

This section details the alternating learning procedure, return targets, and gradient paths used by the allocation interface.

Observed data and fitted quantities. Store observations, remaining-goal masks, commands, task rewards, goal confirmation events, terminal flags, behavior log-probabilities, and policy version identifiers. A basic latent prediction loss is

$$L_{\text{pred}}(\omega) = \mathbb{E}_t \left\| g_\omega(\text{sg}(z_t), a_t) - \text{sg}(z_{t+1}) \right\|_2^2, \tag{18}$$

where sg stops gradients and the predictor's input must include every variable required by its chosen transition model. A physical residual may be added only after defining a decoder, controller-to-actuation map, and discretization. The observation encoder is learned from policy supervision in this formulation; prediction fitting alone does not guarantee sufficient state or prevent representation aliasing.

Fit a task baseline $U_\nu(z_t, b_t)$ to task-return targets $\hat{R}_t$. For each remaining goal i, define an auxiliary reward $r_{i,t} = \mathbb{I}\{i \in U_t,\ i \notin U_{t+1}\}$ and a discounted return $G_{i,t} = \sum_{j=t}^{T-1} \gamma^{j-t} r_{i,j}$. A concrete

fitted-value loss is

$$L_{\text{value}} = \mathbb{E}_t(U_\nu(z_t, b_t) - \text{sg}(\hat{R}_t))^2 + \mathbb{E}_t \frac{1}{|U_t|} \sum_{i \in U_t} (V_\psi(z_t, h_i^t) - \text{sg}(G_{i,t}))^2. \tag{19}$$

This auxiliary goal critic predicts discounted completion, rather than an unspecified mixture of task value and geometric distance. Full-episode returns are well-defined; a truncated rollout needs a declared bootstrap rule. Changing auxiliary targets changes the algorithm and needs validation. Observation features and goal embeddings are treated as fixed during these critic-fitting steps.

Actor improvement. Freeze predictor and critic parameters for Equation (11), but keep their input Jacobians. The auxiliary can still exploit extrapolation error in the fitted model, so model validation and an appropriate β_T remain necessary. The PPO clipping term does not constrain the separate imagined-value objective and supplies no trust-region guarantee for their sum. Gaussian actions in the paper denote the pre-transform parameterization; bounds, squashing, and their log-density corrections must agree between the behavior policy, executed controls, and imagined controls.

Table 5: One training iteration and one deployment step. Rollout scoring is a training objective in this formulation; test-time search is not implied.

Stage	Operation
Collect	Freeze a behavior-policy snapshot. Encode observations and remaining goals; compute allocation and b_t; sample and execute one command. Record the transition and confirmation events.
Fit	Compute task and goal-return targets. Update dynamics and critics from observed transitions, using the stated detached targets.
Improve	Recompute the current allocation and full PPO likelihood ratio. Roll out H steps through fixed model parameters, preserving input gradients. Update actor/allocation/encoders using Equation (11).
Deploy	Read the next RGB-D/proprioceptive input and confirmation mask. If no goal remains, terminate. Otherwise encode, allocate, and execute the actor command through the platform controller; then repeat.

B.1 An executable update order

Algorithm 1 details the gradient routing. The behavior log-probability is stored at collection time. Reusing its denominator is valid, whereas replacing the numerator's current allocation with a stored allocation would optimize a different policy. Completed goals are masked before both attention and softmax, and terminal observations do not enter a new actor computation.

Deployment. At each decision, update the remaining mask from confirmed observations. Terminate if it is empty or a declared episode termination condition holds. Otherwise encode, allocate, and send one policy command to the tracking controller. The next sensor observation closes the loop. A deterministic mean action and a sampled action are different evaluation protocols; the chosen convention must be shared by learned controls and recorded. No imagined trajectory is selected at deployment in Algorithm 1.

Termination and finite rollout targets. Let κ_t equal zero at a true task terminal and one otherwise. Let ζ_t equal one only if transition $t+1$ exists in the same collected trajectory segment. With frozen behavior features and baseline,

$$\delta_t = r_t + \gamma \kappa_t U_{\nu_{\text{old}}}(z_{t+1}, b_{t+1}) - U_{\nu_{\text{old}}}(z_t, b_t), \tag{20}$$

$$\hat{A}_t = \delta_t + \gamma \lambda_{\text{GAE}} \zeta_t \hat{A}_{t+1}, \qquad \hat{R}_t = \hat{A}_t + U_{\nu_{\text{old}}}(z_t, b_t). \tag{21}$$

Here $\lambda_{\text{GAE}} \in [0, 1]$ is a trace parameter, distinct from the goal-allocation vector. At a collector cutoff, bootstrap from the final observation but stop the trace; never bootstrap from an auto-reset observation. An episode budget that defines task failure is a true task terminal, even if an API labels it as a time limit. When no goal remains, the bootstrap value is zero and an empty-set encoder need not be evaluated.

Algorithm 1 DiG-Nav training

Require: Policy parameters Θ; predictor ω; critics ψ, ν; horizon H; declared optimizer and return settings

while the environment-step budget is not exhausted **do**
  Snapshot behavior parameters Θ_{old} and task baseline ν_{old}.
  Collect transitions with $\pi_{\Theta_{\text{old}}}$ through the common controller.
  Store $x_t, \mathcal{G}_0, a_t, r_t$, per-goal confirmations, terminal/truncation flags, and the behavior log-probability of the executed policy command.
  Construct detached task advantages/returns and per-goal targets.
  Encode collected observations with the current encoder; detach fitting features.
  Fit ω with Equation (18); fit ψ, ν with Equation (19).
  Freeze fitted predictor/goal-critic weights as $\bar{\omega}, \bar{\psi}$.
  for each declared policy minibatch/epoch **do**
    Recompute $z_t, h_i^t, \lambda_i^t, b_t$ from stored observations and masks.
    Form the full likelihood ratio in Equation (10).
    Initialize $\tilde{z}_0 = z_t$; hold $U_t, h_i^t, \lambda_i^t, b_t$ fixed along each rollout.
    for $k = 0, \dots, H-1$ **do**
      Sample ϵ_k; form the reparameterized command $\tilde{a}_k$.
      Predict $\tilde{z}_{k+1} = g_{\bar{\omega}}(\tilde{z}_k, \tilde{a}_k)$, retaining input gradients.
    end for
    Compute Q_t^{imag} and minimize Equation (11) over Θ only.
  end for
end while
return policy and evaluator configurations, checkpoints, and recorded evaluations.

For a segment ending at T, a declared alternative to full-episode goal returns is

$$\hat{G}_{i,t} = \sum_{j=t}^{T-1} \gamma^{j-t} r_{i,j} + \gamma^{T-t} \mathbb{I}\{\text{nonterminal cutoff},\ i \in U_T\} V_{\psi_{\text{old}}}(z_T, h_i^T). \tag{22}$$

All targets are detached. A goal already confirmed cannot receive a bootstrap term. The terminal mask and the collector cutoff play different roles: the former removes future task return, while the latter only ends the sampled trace.

Feature drift and parameter ownership. Updating the encoder changes the coordinates in which a fitted predictor and critic operate. Detaching fitting features removes competing gradients but does not solve this distribution shift. Re-encode observations before each fitting block and record encoder versions; held-out prediction/return errors should be evaluated in the same coordinates. Any target encoder, shared backbone, or model update between actor minibatches changes the specification and should be documented. Physical consistency also requires a declared latent decoder, command-to-actuation map, time step, and residual scaling.

B.2 Distance shaping and the task reward

An allocation-weighted distance-change term illustrates how reward shaping can interact with learned priorities. Consider the optional form

$$\Delta d_t = \sum_{i \in U_t} \text{sg}(\lambda_i^t)\big[d_i(p_{t+1}) - d_i(p_t)\big], \qquad r_t^{\text{shaped}} = r_t - \beta_d \Delta d_t. \tag{23}$$

Using the same start-of-transition goal set and weights at both positions avoids rewarding a change in allocation without motion. Nevertheless, learned weights make this shaping policy-dependent; it is not automatically potential-based shaping of a fixed POMDP reward. Nor does it justify exposing oracle distances at inference. A matched study must use the same declared reward rule for all controls, or isolate shaping in an additional ablation. This expression is an optional extension of Equation (1), not a prerequisite for the imagined-value objective.

B.3 Full pathwise derivative

Let D denote differentiation with respect to Θ for fixed sampled noises and fixed model parameters. Let $h_i = h_i^t$ and $\lambda_i = \lambda_i^t$. Then

$$Db = \sum_i (D\lambda_i) h_i + \lambda_i Dh_i, \tag{24}$$

$$D\tilde{a}_k = \partial_\Theta A_\Theta(\tilde{z}_k, b, \epsilon_k) + \partial_z A_\Theta D\tilde{z}_k + \partial_b A_\Theta Db, \tag{25}$$

$$D\tilde{z}_{k+1} = \partial_z g_{\bar{\omega}} D\tilde{z}_k + \partial_a g_{\bar{\omega}} D\tilde{a}_k, \qquad D\tilde{z}_0 = Dz_t, \tag{26}$$

$$DQ_t^{\text{imag}} = \sum_{k=1}^{H} \gamma^k \sum_i \left[(D\lambda_i) V_{\bar{\psi}}(\tilde{z}_k, h_i) + \lambda_i \left(\partial_z V_{\bar{\psi}} D\tilde{z}_k + \partial_h V_{\bar{\psi}} Dh_i \right) \right]. \tag{27}$$

Here A_Θ denotes reparameterized action sampling, with its direct parameter derivative taken at fixed inputs. This derivative shows both the explicit allocation term and the effects through the actor and goal embeddings. A tensor-wide detach of Q_t^{imag} or $\tilde{z}_k$ would remove the relevant gradient path. Freezing $\bar{\omega}, \bar{\psi}$ is different: it retains derivatives with respect to their inputs.

C Proofs and Additional Qualifications

Proof of Proposition 1. The common softmax denominator cancels in the ratio, giving $\lambda_i/\lambda_j = \exp((\ell_i - \ell_j)/\tau_g)$. Thus $p_{ij} = (1 + \exp(-\Delta_{ij}/\tau_g))^{-1}$. Taking logarithms gives the log-odds identity; the identity $2\sigma(u) - 1 = \tanh(u/2)$ gives the margin expression. Differentiation yields $\partial p_{ij}/\partial \Delta_{ij} = p_{ij}(1 - p_{ij})/\tau_g \le 1/(4\tau_g)$. The mean-value theorem proves the drift bound. If the magnitude of a gap exceeds its allowed change, its interval of possible next values excludes zero, proving the sign claim. □

Variable temperatures. The fixed-temperature premise holds during evaluation of a frozen model with a global learned scalar. If temperature is time-varying, the correct general bound is

$$|p_{ij}^{t+1} - p_{ij}^t| \le \frac{1}{4} \left| \frac{\Delta_{ij}^{t+1}}{\tau_{t+1}} - \frac{\Delta_{ij}^t}{\tau_t} \right|. \tag{28}$$

A score-drift bound alone does not control this expression without a bound on temperature variation. Pairwise normalization still cancels other goals, but removing a goal can trivially change the global argmax and is not an unnecessary switch.

Entropy variational identity. For fixed scores, the identity

$$\text{softmax}(\ell/\tau_g) = \arg\max_{\lambda \in \Delta} \left\{ \sum_i \lambda_i \ell_i + \tau_g \mathcal{H}(\lambda) \right\} \tag{29}$$

follows from the stationarity equations. This identity does not make the training coefficient β_H an additional temperature. Substituting $\beta_H \tau_g$ in the variational problem changes the optimizer to $\text{softmax}(\ell/(\beta_H \tau_g))$, a different decision map.

Scale and full-system stability. The allocation Jacobian is $\partial \lambda_i/\partial \ell_j = \lambda_i(\mathbb{I}\{i = j\} - \lambda_j)/\tau_g$. A bound on its entries is not a proof of contraction of the full policy and environment. Moreover, (ℓ, τ_g) and $(a\ell, a\tau_g)$, $a > 0$, produce the same allocation. Temperature values need to be interpreted together with score scale, normalized score gaps, and entropy.

C.1 Entropy and temperature derivatives

For a nonempty fixed active set, write $\bar{\ell} = \sum_i \lambda_i \ell_i$. From the softmax Jacobian and $\sum_i \partial \lambda_i/\partial \ell_j = 0$,

$$\frac{\partial \mathcal{H}}{\partial \ell_j} = -\sum_i (1 + \log \lambda_i) \frac{\lambda_i(\mathbb{I}\{i = j\} - \lambda_j)}{\tau_g} = -\frac{\lambda_j}{\tau_g}(\log \lambda_j + \mathcal{H}), \tag{30}$$

$$\frac{\partial \lambda_i}{\partial \tau_g} = \frac{\lambda_i}{\tau_g^2}(\bar{\ell} - \ell_i), \qquad \left.\frac{\partial \mathcal{H}}{\partial \tau_g}\right|_\ell = \frac{\text{Var}_\lambda(\ell)}{\tau_g^3} \ge 0. \tag{31}$$

The temperature result holds at fixed scores; joint training can also change the scores and their scale. For the learned unconstrained parameter, $\partial\tau_g/\partial\eta_g = \sigma(\eta_g)$. The actor objective therefore receives separate derivatives from imagined value, allocation entropy, action entropy, and the PPO term. Equating β_H with τ_g discards this separation. A single active goal has $\lambda = 1$, zero allocation entropy, and zero allocation-score gradient; the action policy still learns. For numerical evaluation, use log-softmax over active scores, avoid $0 \log 0$ on masked entries, and stop before an all-masked softmax.

Sensitivity of the weighted goal feature. If h_i are fixed across two decisions and $\|h_i\| \le B$, then

$$\|b^{t+1} - b^t\| \le B\|\lambda^{t+1} - \lambda^t\|_1. \tag{32}$$

When embeddings also change, an additional term $\sum_i \lambda_i^{t+1}\|h_i^{t+1} - h_i^t\|$ is needed. This explains why a probability-sensitivity bound alone cannot control actor actions: the actor also depends on changing observations, goal embeddings, and its own input sensitivity. A completion event changes the active set and falls outside a fixed-set temporal bound.

D Episode Oracle and Metric Conventions

D.1 Fixed service vertices

For a static traversability graph, let v_0 be the start and let each goal have one fixed admissible service vertex v_i. Let D_{ij} be the graph's shortest path length between these vertices. The shortest *open* route visits all goals without returning to the start. For subsets A of goal indices, the Held–Karp recurrence is

$$F(\{j\}, j) = D_{0j}, \tag{33}$$

$$F(A, j) = \min_{i \in A \setminus \{j\}} \{F(A \setminus \{j\}, i) + D_{ij}\}, \tag{34}$$

$$d_e^* = \min_j F(\{1, \dots, N\}, j). \tag{35}$$

Its time complexity is $O(N^2 2^N)$ and storage is $O(N 2^N)$. No return-to-start edge is added. The goal representatives must agree with the evaluator: replacing admissible service poses with object centers can invalidate the denominator.

D.2 Category goals and service regions

For category i, define $\mathcal{C}_i$ as all graph states satisfying the same visibility and distance predicate for any valid instance of i. A state may require camera heading as well as position. A precise discrete oracle searches the product graph (v, A), where A records categories already satisfied. Traversing (v, w) changes the mask to

$$A' = A \cup \{i : w \in \mathcal{C}_i\} \tag{36}$$

and incurs the original nonnegative edge cost. Shortest-path search from the correctly initialized start mask to any state with all categories satisfied gives the exact optimum on that discretized graph. Simply using pairwise minimum distances between category sets is generally insufficient: the instance or service pose minimizing one leg may differ from the one minimizing the next.

The graph, success predicates, and discretization must be saved. A static geometric oracle is not a time-dependent or dynamically feasible oracle for moving obstacles and aircraft. If used in those settings, it must be named and interpreted as a geometric reference rather than an optimal feasible flight trajectory.

Degenerate and invalid episodes. Exclude unreachable or invalid episodes before evaluating any method, using one common manifest and documenting the exclusions. For a valid all-goals-at-start episode with $d_e^* = 0$, define the successful zero-length score as one and a successful positive-length score as zero; unsuccessful episodes score zero. Alternatively, exclude such episodes uniformly and report that choice. Do not silently clamp an inconsistent oracle to make results appear valid.

Algorithm 2 follows Dijkstra's invariant because every lifted edge retains a nonnegative cost. A projected walk and its accumulated service mask have the same length in both graphs; hence the

Algorithm 2 Category-service oracle on a declared static graph

Require: Graph $\mathcal{V}, \mathcal{E}$ with nonnegative lengths; start v_0; service predicate $C(v)$ returning a category bit mask

$A_0 \leftarrow C(v_0)$; $d(v_0, A_0) \leftarrow 0$; all other distances $\leftarrow \infty$.
Push $(0, v_0, A_0)$ into a minimum-priority queue.
while the queue is not empty **do**
    Pop (c, v, A) with smallest cost; skip if $c \neq d(v, A)$.
    if A contains every requested category **then**
        return c and the predecessor path.
    end if
    for each edge (v, w) of length D_{vw} **do**
        $A' \leftarrow A \cup C(w)$; $c' \leftarrow c + D_{vw}$.
        if $c' < d(w, A')$ **then**
            Update distance and predecessor; push (c', w, A').
        end if
    end for
end while
return unreachable (exclude by the common pre-evaluation rule).

first fully served state popped has minimum valid length. The worst-case state count is $|\mathcal{V}|2^N$, with $O(|\mathcal{E}|2^N \log(|\mathcal{V}|2^N))$ time under a binary-heap implementation. This is an offline evaluation procedure, not information made available to the policy.

E Evaluation Details and Physical Demonstrations

E.1 Physical demonstrations and evaluation conventions

The physical campaign contains 100 trials. Figure 4 contains six selected demonstrations, each showing four red target markers and a blue executed trajectory. Panel letters identify demonstrations rather than chronological trial numbers. The visualization does not encode a success threshold, time, velocity, confidence, or a calibrated clearance scale. Point-cloud colors visualize the spatial reconstruction and do not define the target-category labels.

The images allow inspection of route shape and target placement in a common type of layout. They cannot resolve whether a marker was confirmed under the task's visibility rule, how close the vehicle passed to an obstacle in 3D, or whether an operator intervened. Different route shapes also do not establish that allocation entropy caused a behavioral difference. Linking the trajectory to time-aligned scores, masks, and confirmation events would make that interpretation testable.

Trial outcomes. Mission completion, individual goal confirmations, collision events, and manual interventions describe different aspects of a physical trial. An intervention can prevent a collision without completing the task autonomously. Likewise, a collision-free trajectory can terminate before every goal is found. Failure and aborted-trial outcomes therefore belong in the evaluation denominator. The total campaign size does not imply 100 trials per method or 100 independent scenes or sessions.

Image-to-log alignment. Interpreting a projected trace requires its coordinate frame, pose source, and camera/vehicle extrinsics. In particular, a top-down projection can hide vertical motion and does not determine three-dimensional obstacle clearance. Its two-dimensional length cannot replace the executed 3D path length in aerial path-efficiency scoring. The six displayed missions are selected examples rather than a random sample for estimating aggregate performance.

Interpretation of the demonstrations. The six visualizations illustrate route geometry and target placement rather than an aggregate success rate. A rate computed over the campaign has a denominator of 100 eligible trials, while uncertainty also depends on repeated runs within sessions and layouts. We therefore keep the qualitative physical demonstrations separate from the quantitative simulation comparison.

E.2 Common simulation protocol and statistical units

An evaluation episode is defined by its scene, start pose, requested categories, admissible object instances, confirmation predicate, action interface, and budget. Reusing episodes across methods controls variation in task difficulty; scene-disjoint training and evaluation separate navigation learning from scene memorization. With multiple object instances, the evaluator and oracle must accept the same set of service poses. The 0.5 m radius is one component of the predicate, together with visibility and confirmation.

Matched controls share detector weights, observations, rewards, controller, optimizer budget, and evaluation episodes. A hard-allocation control changes b_t to the selected goal embedding and removes the pathwise soft-allocation derivative. A detached-imagination control preserves the policy-side soft allocation but detaches its weight tensor only on the imagined-value branch; the precise detach location must be reported. The full control should use the same predictor/critic fitting losses so that loss of a gradient pathway is separated from removal of model capacity. Fitting a larger PPO baseline requires measuring both parameter count and actual decision compute.

The seed is the training replicate: each seed contributes a metric mean over evaluation episodes, and the standard error summarizes variation among these means. Pairwise analysis instead starts from differences on matched episodes. Scene-level clustering matters when many episodes share a layout; treating those episodes as independent replicates understates shared environmental variation. A hierarchical bootstrap can resample scenes and episodes within scenes while retaining method pairing, with training runs resampled at a separate level. This is a statistical extension of the descriptive comparisons in the main text; no bootstrap significance test is claimed from the aggregate means and standard errors alone.

Computational complexity. At width d, dense goal attention costs $O(N^2d + Nd^2)$, while a shared scorer scales with N. The imagined objective requires H model steps and HN goal-value evaluations, which can be batched. A shared network does not acquire additional trainable parameters merely because more goals are present. Training and deployment have different costs because the deployed actor omits latent imagination. Device latency additionally depends on precision, batching, synchronization, sensing, and controller overhead; asymptotic complexity alone does not determine real-time performance.

E.3 Baseline roles and multi-goal interfaces

Table 1 compares navigation systems on the same category-set objective. Their base methods solve different subproblems, so the role of each component matters when interpreting the comparison.

SAM: sequence-agnostic multi-object navigation. Gireesh et al. (2023) introduce an actor–critic framework that selects navigation subgoals without a prescribed visitation order. It provides a direct comparison for target prioritization. The interface consists of the current observation, remaining categories, selected navigation subgoal, and motion policy.

PONI: potential-based semantic search. Ramakrishnan et al. (2022) learn semantic-map potential functions from passive maps and use them to select waypoints for a requested category. In a multi-goal system, a category-selection layer supplies the active category and updates it after confirmation, while the semantic map can persist across targets. The aerial comparison additionally requires waypoint tracking in the aerial action space. A two-dimensional semantic map alone does not define unrestricted three-dimensional motion.

EGO-Planner: local trajectory optimization. Zhou et al. (2021b) optimize collision-aware quadrotor trajectories to supplied local goals. Within an object-navigation system, semantic search and goal ordering supply waypoints, and the planner supplies a trajectory for the low-level controller. This row therefore evaluates the navigation stack rather than an isolated semantic allocation algorithm.

FUEL: hierarchical exploration. Zhou et al. (2021a) combine an incremental frontier structure with coverage planning and quadrotor trajectory generation. Object navigation couples this explo-

ration process to category detection and confirmation. Complete coverage and completion of a requested object set are different stopping criteria.

Other navigation systems. ESC uses commonsense constraints, MOON uses optimization-based planning, STRIVE combines structured representations with vision-language reasoning, and OctoNav uses a generalist navigation policy. Their multi-goal interfaces must track confirmed categories across successive searches. Native training procedures differ: PONI learns map potentials with supervised training, whereas EGO-Planner and FUEL use planning and optimization. The environment-step axis in Figure 2 refers to the evaluated multi-goal systems; it does not imply that every base component is trained with PPO.