

INFORMATION-DRIVEN VISUAL NAVIGATION FOR UAVS USING DEEP REINFORCEMENT LEARNING

Bc. Vojtěch Bešťák

Master's thesis
Faculty of Information Technology
Czech Technical University in Prague
Department of Applied Mathematics
Study program: Informatics
Specialisation: Knowledge Engineering 2020
Supervisor: Ing. Václav Truhlařík
May 7, 2026

FACULTY OF INFORMATION TECHNOLOGY CTU IN PRAGUE

Assignment of master's thesis

Title:	Information-Driven Visual Navigation for UAVs Using Deep Reinforcement Learning
Student:	Bc. Vojtěch Bešťák
Supervisor:	Ing. Václav Truhlařík
Study program:	Informatics
Branch / specialization:	Knowledge Engineering
Department:	Department of Applied Mathematics
Validity:	until the end of summer semester 2027/2028

Instructions

Design, implement, and evaluate a reinforcement-learning (RL)-based visual navigation system for an Unmanned Aerial Vehicle (UAV) equipped with a single nadir RGB camera. The agent must navigate toward a given global goal while actively preferring trajectories that maximize visual information density.

1. Familiarize yourself with reinforcement learning methods for image-based navigation and define the Markov Decision Process (MDP). Specify how the goal is represented in the agent's input.
2. Implement selected method(s) and train the agent on available map data with simplified grid-based motion. Set up the network and preprocessing pipeline and tune the main training hyperparameters.
3. Define a computable proxy of visual information density (e.g., edges, keypoints, learned features). Build a reward that balances goal progress and information density and justify the weighting.
4. Transfer the system to a more realistic motion model (potentially using high-fidelity simulator). Adapt the agent's action space and training setup.
5. Evaluate in both environments using suitable metrics and compare against a goal-only baseline. Present quantitative results and qualitative trajectory examples with failure-case analysis.

Electronically approved by Ing. Magda Friedjungová, Ph.D. on 13 March 2026 in Prague.

Czech Technical University in Prague
Faculty of Information Technology
© 2026 Bc. Vojtěch Bešťák. All rights reserved.
This thesis is school work as defined by Copyright Act of the Czech Republic. It has been submitted at Czech Technical University in Prague, Faculty of Information Technology. The thesis is protected by the Copyright Act and its use, with the exception of free statutory licenses and beyond the scope of the authorizations specified in the Declaration, requires the consent of the author.

Citation of this thesis: Bešťák Vojtěch. *Information-Driven Visual Navigation for UAVs Using Deep Reinforcement Learning.* Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2026.

I would like to express my sincere gratitude to my supervisor, Ing. Václav Truhlařík, for his guidance, patience, and the valuable technical insights he provided throughout this project. I am also grateful to the Intelligent and Mobile Robotics group for providing the environment and resources necessary for my research. I would also like to thank Terezie Hrubanová for facilitating the initial introduction to the team at CIIRC. Furthermore, I would like to thank RNDr. Milan Straka, Ph.D., for his expert consultation regarding the reinforcement learning paradigms used in this work. Finally, I would like to thank my family and friends for their continuous support and encouragement during my studies.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as a school work under the provisions of Article 60 (1) of the Act.

I declare that I have used AI tools during the preparation and writing of my thesis. I have verified the generated content. I confirm that I am aware that I am fully responsible for the content of the thesis.

In Prague on May 7, 2026

Abstract

Autonomous navigation of Unmanned Aerial Vehicles (UAVs) in GPS-denied environments heavily relies on Visual Place Recognition (VPR). However, VPR frequently fails in textureless or repetitive terrain due to severe perceptual aliasing. This thesis proposes an active perception strategy formulated as a Partially Observable Markov Decision Process (POMDP) to enhance localization reliability. The objective is to train a reactive flight controller that balances path efficiency with the active acquisition of visually unique landmarks. The methodology employs a two-phase framework, advancing from a discrete semantic grid-world to a continuous aerial simulator featuring authentic orthophotos and non-holonomic unicycle kinematics. To resolve perceptual aliasing without relying on raw pixel heuristics, the system integrates a SelaVPR++ Vision Transformer to generate a deep semantic saliency prior. A rigorous comparative analysis evaluates deep reinforcement learning and imitation learning paradigms: value-based, policy-gradient, and behavioural cloning approaches. The empirical results demonstrate that while imitation learning (DAgger) collapses under continuous dynamics due to the compounding of informational asymmetry and non-holonomic kinematic drift, Proximal Policy Optimization (PPO) successfully internalizes the navigation-perception trade-off. Within the training topographies, the final PPO model closes 23.4% of the localization performance gap between a naive shortest-path baseline and a globally informed Dijkstra-Pure-Pursuit Oracle; in zero-shot evaluation on held-out regions, the corresponding ratio is 31.2%, confirming that the policy generalizes to unseen environments without retraining. These findings confirm that reinforcement learning agents can effectively internalize global path-cost heuristics for localized trajectory refinement in visually ambiguous domains.

Keywords Unmanned Aerial Vehicles (UAV), Deep Reinforcement Learning, Visual Place Recognition (VPR), Active Perception, POMDP, Proximal Policy Optimization (PPO), Perceptual Aliasing, SelaVPR++

Abstrakt

Autonomní navigace bezpilotních letounů (UAV) v prostředích bez dostupnosti signálu GPS v zásadní míře spoléhá na vizuální rozpoznávání míst (VPR). Tyto systémy však často selhávají v monotónním či texturově chudém terénu v důsledku výrazného percepčního aliasingu. Tato práce navrhuje přístup založený na aktivním vnímání, který je formulován jako částečně pozorovatelný Markovův rozhodovací proces (POMDP), za účelem zvýšení spolehlivosti lokalizace. Cílem je vyvinout reaktivní algoritmus řízení letu, který vyvažuje efektivitu cesty k cíli s aktivním vyhledáváním vizuálně unikátních orientačních bodů. Vývoj probíhá ve dvou fázích, od prostředí s diskrétní mřížkou a daty z OpenStreetMap až po spojitý letecký simulátor s reálnými ortofotomapami a neholonomním kinematickým modelem. K vyřešení problému vizuální nejednoznačnosti systém namísto heuristik na úrovni pixelů integruje model SelaVPR++ založený na architektuře Vision Transformer, která poskytuje sémantickou informaci o významnosti prvků v obraze. Systematické srovnání vyhodnocuje metody posilovaného a imitačního učení. Empirické výsledky ukazují, že zatímco imitační učení (DAgger) ve spojitém prostředí selhává kvůli součinnému působení informační asymetrie a neholonomního kinematického driftu, algoritmus Proximal Policy Optimization (PPO) úspěšně nalézá kompromis mezi navigací a vnímáním. V rámci trénovacích topografií výsledný model snižuje o 23,4 % rozdíl ve výkonnosti mezi naivním přístupem nejkratší cesty a privilegovaným globálním algoritmem Dijkstra-Pure-Pursuit. Při vyhodnocení na zcela nových oblastech odpovídající hodnota činí 31,2 %, což potvrzuje schopnost algoritmu generalizovat na neviděná prostředí bez přetrénování. Tato zjištění potvrzují, že lokální reaktivní kontroléry mohou efektivně využívat globální charakteristiky terénu pro zpřesnění trajektorie ve vizuálně náročných oblastech.

Klíčová slova Bezpilotní letouny (UAV), hluboké posilované učení, vizuální rozpoznávání míst (VPR), aktivní vnímání, POMDP, Proximal Policy Optimization (PPO), percepční aliasing, SelaVPR++

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Problem Statement . . . 2
- 1.3 Contributions . . . 2
- 1.4 Thesis Outline . . . 3

I Theoretical Foundation **4**

2 Reinforcement Learning **5**
- 2.1 Markov Decision Process (MDP) . . . 5
- 2.2 Policies, Value Functions, and Bellman Equations . . . 7
- 2.3 Partially Observable MDPs (POMDP) . . . 9
- 2.4 Deep Reinforcement Learning Algorithms . . . 12
  - 2.4.1 Value-Based Learning (DQN) . . . 12
  - 2.4.2 Policy-Gradient Methods (PPO) . . . 14
- 2.5 Recurrent RL and Memory . . . 17
  - 2.5.1 Long Short-Term Memory (LSTM) . . . 18
  - 2.5.2 Recurrent PPO with LSTM . . . 19

3 Imitation Learning **20**
- 3.1 Behavioural Cloning (BC) . . . 21
- 3.2 Dataset Aggregation (DAgger) . . . 21
- 3.3 Asymmetric Information . . . 22

4 Computer Vision and Perception **24**
- 4.1 Neural Architectures for Visual Perception . . . 24
  - 4.1.1 Convolutional Neural Networks (CNN) . . . 24
  - 4.1.2 Visual Transformers (ViT) . . . 26
- 4.2 Fundamentals of Visual Place Recognition . . . 29
  - 4.2.1 Problem Formulation and Metrics . . . 29
  - 4.2.2 Perceptual Aliasing and Appearance Change . . . 30
- 4.3 Semantic Feature Extraction and Adaptation . . . 31
  - 4.3.1 Local and Learned Feature Descriptors . . . 31

4.3.2 Seamless Adaptation (SelaVPR) 32

II Realization and Performance Analysis 34

5 Phase I – Conceptual Validation in Semantic Environments 35
5.1 Environment and POMDP Formulation 35
5.1.1 Spatial Scale and Data Preprocessing 36
5.1.2 State and Action Spaces 36
5.1.3 Reward Design 37
5.2 Algorithmic Baseline (Dijkstra Oracle) 40
5.3 Evaluated Architectures 41
5.3.1 Shared Multi-Modal Backbone 42
5.3.2 Temporal Integration 43
5.3.3 Algorithm-Specific Heads 43
5.4 Learning Paradigms 43
5.4.1 Value-Based (DQN) 44
5.4.2 Policy-Based (PPO) 44
5.4.3 Imitation Learning (DAgger) 45
5.4.4 Hybrid Approach: IL Pre-training with RL Fine-tuning 45
5.5 Experimental Results 46
5.5.1 Experimental Setup and Baselines 46
5.5.2 Evaluation Framework 47
5.5.3 Objective Balancing 49
5.5.4 Ablation Study of the Curriculum Strategy 53
5.5.5 Comparative Analysis of Learning Paradigms 56
5.6 Analysis and Preliminary Conclusions 60
5.6.1 Summary of Experimental Findings 60
5.6.2 Limitations and the Transition to Phase II 61

6 Phase II – Active Perception in Aerial Imagery 63
6.1 Geospatial Data and Offline Perception 64
6.1.1 Aerial Imagery and Spatial Preprocessing 64
6.1.2 Offline VPR Database 65
6.2 The Continuous 2D Simulation Engine 66
6.2.1 Hierarchical Coordinate Mapping 66
6.2.2 Arena Selection and Rejection Sampling 67
6.2.3 Dynamic Saliency Potential Map Generation 68
6.2.4 UAV Kinematics and Actuation 69
6.2.5 Virtual Electronic Image Stabilization (EIS) 70
6.3 Continuous POMDP Formulation 72
6.3.1 State, Action, and Observation Spaces 72

6.3.2 Continuous Reward Design 73
6.4 Algorithmic Baseline (Continuous Cost-Graph Oracle) 76
6.4.1 VPR-Aware Cost Landscape 76
6.4.2 Trajectory Generation via Pure Pursuit 76
6.5 Evaluated Architectures and Paradigms 78
6.5.1 Architectural Scaling and Continuous Heads 79
6.5.2 Optimization Paradigms 80
6.6 Experimental Results and VPR Evaluation 81
6.6.1 Experimental Setup and Baselines 82
6.6.2 Evaluation Framework and VPR Metrics 83
6.6.3 Baseline Perceptual Performance 84
6.6.4 Objective Balancing and Reward Dynamics 85
6.6.5 Structural Ablation Studies 88
6.6.6 Comparative Analysis of Learning Paradigms 93
6.6.7 Zero-Shot Generalization to Unseen Topography 96
6.7 Analysis and Preliminary Conclusions 98
6.7.1 Summary of Experimental Findings 98

7 Discussion 100
7.1 Asymmetry and Policy Stability 100
7.2 Kinematics and Domain Transfer 101
7.3 Perceptual Priors and Horizons 102

8 Conclusion 104
8.1 Future Work 105

A Experimental Tracking and Reproducibility 106
A.1 Software Implementation 106
A.2 Pre-trained Model Parameters 106
A.3 Weights & Biases Training Histories 107

Bibliography 108

Enclosed Media Contents 114

List of Figures

2.1 The agent-environment interaction in a MDP 7
2.2 The agent-environment interaction in a POMDP 10

5.1 Visual pipeline for the feature-aware Oracle cost landscape . . . 41
5.2 Architectural overview of the Phase I. model. 42
5.3 Reward hyperparameter sensitivity analysis 50
5.4 Navigational efficiency and feature acquisition trade-off 51
5.5 Ablation study of the curriculum learning strategy 54
5.6 Qualitative trajectory analysis of the curriculum domain shift . 55
5.7 Learning dynamics across optimization paradigms 57
5.8 Decision-making biases across RL and IL paradigms 59
5.9 Comparative visualization of learned Phase I value landscapes . 60

6.1 Kinematic model and coordinate frames for Phase II. 70
6.2 SelaVPR-based saliency and Oracle cost landscape 77
6.3 Geometric derivation of the Pure Pursuit tracking law. 78
6.4 Architectural adaptations for the Phase II navigation model. . 80
6.5 Sensitivity analysis of the continuous navigation policy 86
6.6 Navigational overhead and localization precision trade-off . . . 87
6.7 Ablation analysis of the saliency prior and reward dynamics . . 90
6.8 Qualitative trajectory comparison of structural ablations 92
6.9 Learning dynamics across continuous optimization paradigms . 94
6.10 Decision-making biases in the continuous domain 95
6.11 Zero-shot generalization on held-out topographies 98

List of Tables

5.1 Phase I hyperparameter configurations 47
5.2 Performance of the selected Phase I sweep policy 52
5.3 Evaluation of curriculum strategies on the OSM domain 54
5.4 Comparison of learning paradigms in the OSM domain 58

6.1 Topographical characteristics of the training dataset 65
6.2 Phase II hyperparameter configurations 83
6.3 Comparative perceptual performance of deterministic baselines 85
6.4 Performance of the selected Phase II sweep policy 88
6.5 Structural ablation study hyperparameters 90
6.6 Quantitative comparison of Phase II architectural variants . . . 91
6.7 Comparison of learning paradigms in the continuous domain . . 94
6.8 Zero-shot generalization performance on held-out maps 97

A.1 Pre-trained agents available on the Hugging Face Hub 106
A.2 WandB logging summary for Phase I experiments. 107
A.3 WandB logging summary for Phase II experiments. 107

List of Abbreviations

AIL	Asymmetric Imitation Learning
ALE	Average Localization Error
BC	Behavioural Cloning
BPTT	Back-Propagation Through Time
CNN	Convolutional Neural Network
ČÚZK	Czech State Administration of Land Surveying and Cadastre
DAgger	Dataset Aggregation
DQN	Deep Q-Network
EIS	Electronic Image Stabilization
EPSG	European Petroleum Survey Group
FOV	Field of View
GAE	Generalized Advantage Estimation
GELU	Gaussian Error Linear Unit
GeM	Generalized Mean Pooling
GPS	Global Positioning System
GSD	Ground Sampling Distance
HASP	Heuristic-Augmented Static Policy
ISP	Implicit Static Policy
IL	Imitation Learning
IMU	Inertial Measurement Unit
KL	Kullback-Leibler
LN	Layer Normalization
LSTM	Long Short-Term Memory
MDP	Markov Decision Process
MLP	Multi-Layer Perceptron
MSA	Multi-Head Self-Attention
MSE	Mean Square Error
NN	Neural Network
NLP	Natural Language Processing
OSM	OpenStreetMap
POMDP	Partially Observable Markov Decision Process
PPO	Proximal Policy Optimization
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network

RQ	Research Question
SASP	Saliency-Augmented Static Policy
SADP	Saliency-Augmented Dynamic Policy
SelaVPR	Seamless Adaptation for Visual Place Recognition
S-JTSK	System of the Unified Trigonometric Cadastral Network
TD	Temporal Difference
TPE	Tree-structured Parzen Estimator
TRPO	Trust Region Policy Optimization
UAV	Unmanned Aerial Vehicle
ViT	Vision Transformer
VPR	Visual Place Recognition
6-DOF	Six Degrees of Freedom

List of Notation

General Mathematical Notation

$\doteq$	equality relationship true by definition
$\mathbb{E}[X]$	expectation of a random variable X
$\Pr\{X = x\}$	probability that a random variable X takes value x
$\mathbb{R}$	set of real numbers
$\|\cdot\|_2$	Euclidean norm (L_2 norm)
$\mathbb{I}$[predicate]	indicator function; 1 if true, else 0

The Agent-Environment Interface (MDP/POMDP)

$\mathcal{S}$	state space describing all possible configurations
$\mathcal{S}_{\text{goal}}, \mathcal{S}_{\text{safe}}$	sets of goal states and safe operational states
$\mathcal{A}, \mathcal{A}(s)$	action space; actions available in state s
Ω	observation space of the environment (POMDPs)
$\mathcal{R}$	set of possible rewards; $\mathcal{R} \subset \mathbb{R}$
$t, T_{\max}$	discrete time step and maximum episode horizon
S_t, A_t, R_t	state, action, and reward at time t
O_t	observation received at time t
$P(s' \mid s, a)$	transition probability distribution over states
$r(s, a, s')$	expected reward on transition from s to s' via a
γ, G_t	discount factor and expected return from time t
H_t	history of actions, observations, and rewards
$b_t(s), \mathcal{B}$	belief state and the set of all belief distributions
τ	discrete sequence of states and actions (trajectory)

Policies and Value Functions

π	a policy (agent's decision-making strategy)
$\pi(a \mid s)$	probability of selecting action a in state s
$\pi_E, \hat{\pi}$	expert policy and implicit policy (Imitation Learning)
v_π, q_π	state-value and action-value functions for policy π
v_*, q_*	optimal state-value and action-value functions
V, Q	neural approximations of value functions
$\rho_\pi(s, a)$	state-action occupancy measure under policy π
$d^\pi(s, b)$	discounted state-belief occupancy measure
$A^\pi(s, a)$	advantage function measuring relative action value
$\mathcal{H}[\pi]$	entropy of the policy distribution

Optimization and Deep Reinforcement Learning

$d_{\boldsymbol{\theta}}$	dimensionality of parameter vector $\boldsymbol{\theta}$
$\boldsymbol{\theta}, \boldsymbol{\theta}_t, \boldsymbol{\theta}^-$	neural parameters (online, current, and target)
$\boldsymbol{\theta}_\mu, \boldsymbol{\theta}_\sigma$	parameters for policy mean and variance
$J(\boldsymbol{\theta}), \mathcal{L}(\boldsymbol{\theta})$	theoretical objective and surrogate loss functions
α	learning rate parameter
$\epsilon_{\text{greedy}}, \epsilon_{\text{clip}}$	exploration probability and PPO clipping range
$r_t(\boldsymbol{\theta})$	probability ratio between current and previous policies
$\delta_t, \hat{A}_t$	TD error and generalized advantage estimate (GAE)
λ	smoothing parameter for GAE trade-off
$\mathcal{M}$	experience replay buffer used in DQN
$\mathcal{D}, \mathcal{D}_n$	datasets of expert demonstrations
β_n	DAgger expert query probability parameter

Neural Architectures and Observations

$\mathbf{h}_t, \mathbf{c}_t$	hidden state and cell state vectors at time t
$\mathbf{x}_t$	input vector to the recurrent layer at time t
d_{in}, d_h	dimensionality of input and hidden state vectors
$\mathbf{g}_t^f, \mathbf{g}_t^i, \mathbf{g}_t^o$	forget, input, and output gate activations
$\mathbf{W}, \mathbf{U}, \mathbf{b}$	input weights, recurrent weights, and bias vectors
$\mathbf{e}_{\text{view}}, \mathbf{e}_{\text{goal}}$	visual and navigational latent embeddings
$\mathbf{e}_t, \mathbf{y}_t$	multi-modal embedding and temporal feature vector
$\mathbf{I}_t, \mathbf{I}_t^{\text{raw}}$	processed visual observation and unscaled physical buffer
$\mathbf{D}_t$	visitation density tensor in the observation
$\mathbf{M}_t$	saliency prior tensor in the observation
$\mathbf{g}_t, \tilde{\mathbf{g}}_t$	stacked and instantaneous telemetry observation vectors
N_f	number of stacked frames in the observation

Vision and Place Recognition (VPR)

$\mathcal{I}, \mathcal{V}$	image space and reference VPR database
$\mathcal{D}_{\text{arena}}$	VPR database subset within active arena
N_{db}	number of images in the reference database
$\mathbf{c}_i, \mathbf{z}_i$	coordinates and global descriptors of database entries
$\mathbf{M}_i, \hat{\mathbf{M}}_i$	database saliency matrix and its upscaled version
w_i	arena-relative uniqueness weight for database entries
$\psi(I), d(\cdot, \cdot)$	feature extraction function and VPR distance metric
$I * K$	discrete 2D cross-correlation of image and kernel
$\nabla I, M$	image gradient vector and structural magnitude
$\mathbf{x}_p, P_{\text{size}}$	flattened image patch and fixed patch size
N_{patches}	number of patches extracted from a single image
$\mathbf{E}, \mathbf{E}_{\text{pos}}$	patch projection matrix and positional embeddings
$\mathbf{z}_\ell$	latent embeddings at transformer layer ℓ
$\mathbf{Q}, \mathbf{K}, \mathbf{V}$	Query, Key, and Value matrices in self-attention
$d_{\text{emb}}, \mathbf{F}, \mathbf{f}$	latent dimensionality, feature tensor, and descriptor

$\mathbf{M}, \mathbf{M}_{\text{local}}$	saliency map derived from patch tokens
L_{total}	side length of aggregated global textures
S_m, S_{px}	physical and pixel strides for database generation
r_{vpr}	spatial radius threshold for a successful match
$\epsilon_{\text{vpr}}, \delta_{\text{pos}}$	feature and spatial thresholds for aliasing
ρ	ground sampling distance (GSD) of imagery

Simulation, Geometry, and Kinematics

$\mathbf{p}_t$	spatial Cartesian position vector $[x_t, y_t]^\top$
$\boldsymbol{\zeta}_t$	kinematic state $[x_t, y_t, \varphi_t, v_t, \omega_t]^\top$
$\mathbf{u}_t$	normalized control action $[u_v, u_\omega]^\top$
$v_{\text{cmd}}, \omega_{\text{cmd}}$	target physical linear and angular velocities
$V_{\max}, \Omega_{\max}$	physical limits for longitudinal and angular velocity
$A_{\max}, \alpha_{\max}$	limits for linear and angular acceleration
$\hat{v}_t, \hat{\omega}_t$	current normalized linear and angular velocities
d_t, d_{init}	goal distance (current and episode start)
$\Delta x_t, \Delta y_t$	goal direction unit vector components
$\Delta x_{\text{ego}}, \Delta y_{\text{ego}}$	egocentric goal direction components
e_t	remaining battery life or fuel ratio
H_a, W_a	dimensions of the flight arena in grid cells
H_c, W_c	spatial dimensions of the physical camera footprint
$H_{\text{in}}, W_{\text{in}}$	spatial dimensions of the neural input tensor
$W_{\text{ground}}, \alpha_{\text{fov}}$	physical footprint width and camera FOV
s	discretization factor (pixels per grid cell)
N_{buf}	pixel dimension of oversized stabilization buffer
ϵ_{goal}	spatial radius threshold for goal success
e_φ, l, κ	heading error, lookahead distance, and curvature
k_ω	proportional gain for Pure Pursuit controller

Reward Formulation and Scheduling

$R_{\text{success}}, R_{\text{fail}}$	terminal rewards for success and failure
$R_t^{(\text{step})}, R_t^{(\text{dist})}$	composite navigational and dense distance rewards
$R_t^{(\text{view})}, R_{\text{time}}$	view-based feature reward and step time penalty
$R_t^{(\text{border})}, R_t^{(\text{out})}$	collision and out-of-bounds penalties
c_{black}	negative scaling constant for out-of-bounds penalty
$c(\mathbf{p}), \eta$	spatial cost landscape and sensitivity hyperparameter
$\nu(\mathbf{p})$	visual salience evaluated at position $\mathbf{p}$
$\mathcal{F}_t, \hat{\mathcal{F}}$	Feature Availability Map (current and raw)
λ_{dep}	per-step depletion rate of the Feature Availability Map
$E_{\text{global}}, E_t(\mathbf{u})$	global edge map and local visual edges
G_x, G_y	spatial gradient maps from Sobel filtering
$G(\mathbf{u}), G_\sigma$	centering Gaussian mask and smoothing kernel
$\text{FOV}(\mathbf{p}), \chi_t^{\text{FOV}}$	visible coordinates and footprint indicator

$C_t(\mathbf{u})$	visual reward harvested at coordinate $\mathbf{u}$
N_{ref}	per-step reward normalization constant
$C_{\max}$	maximum magnitude for Sobel reward normalization
$\delta, \delta_{\text{safe}}$	noise threshold and safe operational margin
Ψ_t	progress coefficient for dynamic reward scheduling
$w_{\text{dist}}, w_{\text{view}}$	objective weights for distance and visual features

Evaluation Metrics and Baselines

$\xi, \mathcal{W}^*$	trajectory and optimal waypoint sequence
$L(\xi)$	physical length of the trajectory
$K, \Delta l_i$	segment count and length of discrete segment i
$\mathcal{C}(\xi), \hat{C}(\xi)$	theoretical path cost and its discrete approximation
$\text{Opt}(\xi), \text{Ext}(\xi)$	optimality score and path extension ratio
$\text{Cap}(\xi), I(\xi)$	feature capture ratio and total visual interest
R@ k	recall at top-k candidates for VPR localization
$\text{ALE}(\xi)$	average localization error over the trajectory
$\text{Succ}(\pi)$	empirical success rate of a policy
$\hat{\mathbf{p}}_t$	coordinate of top-ranked retrieved reference image
$\mathcal{T}_{\text{sub}}, N_{\text{sub}}$	set of evaluation indices and its cardinality
$N_{\text{ev}}, N_{\text{sd}}$	count of evaluation episodes and random seeds
f_{eval}	VPR query frequency during evaluation (Hz)

Chapter 1
Introduction

1.1 Motivation

With the increasing deployment of Unmanned Aerial Vehicles (UAVs) in applications such as search-and-rescue, agriculture, and surveillance, there is a critical need for robust autonomous navigation. To operate effectively, a UAV must reliably localize itself within dynamic and unmapped environments. Traditionally, UAVs rely on multi-sensor fusion, combining Inertial Measurement Units (IMUs) for short-term state estimation with Global Positioning Systems (GPS) for absolute coordinates. However, relying on GPS introduces significant vulnerabilities. The signal can be jammed, spoofed, or naturally degraded in GPS-denied environments like deep valleys or urban canyons [1]. Consequently, modern robotic systems increasingly turn to onboard vision to maintain spatial awareness.

A prominent solution is Visual Place Recognition (VPR), where the current camera view is queried against a pre-computed reference database of geotagged images [2]. While foundation models have significantly improved retrieval accuracy, VPR systems remain fundamentally limited by the visual distinctiveness of the terrain. When a UAV traverses homogeneous environments such as dense forests, water bodies, or repetitive agricultural fields, it suffers from severe perceptual aliasing. In these regions, spatially distant locations produce nearly identical visual observations. To maintain localization confidence, the drone must actively avoid these featureless zones and prioritize trajectories that follow distinctive semantic boundaries, such as roads, village perimeters, or riverbanks. Executing this behaviour reactively using only an onboard nadir camera constitutes a highly complex, partially observable control problem.

1.2 Problem Statement

Standard path-planning algorithms typically compute the geometrically shortest route to a target destination. These approaches inherently assume that the vehicle will maintain perfect state estimation throughout the flight. They do not account for the degradation of VPR confidence over perceptually aliased terrain. To resolve this, the UAV requires an "Active Perception" strategy. It must learn to deliberately deviate from the optimal geometric path to acquire visually unique landmarks, thereby minimizing state uncertainty.

This thesis approaches the active perception challenge as a Partially Observable Markov Decision Process (POMDP). The objective is to design, implement, and evaluate a reinforcement learning system that operates as a local, reactive flight controller. The agent must balance a multi-objective reward signal: minimizing the physical distance to a global coordinate while maximizing the accumulation of distinct semantic features within its limited sensor footprint. Furthermore, this policy must remain stable under the physical constraints of non-holonomic unicycle kinematics, where trajectory deviations incur significant temporal and velocity penalties.

1.3 Contributions

This thesis provides a comprehensive framework for training and evaluation of feature-aware navigation policies. The development is structured to ensure a rigorous transition from theoretical abstractions to high-fidelity aerial application. The primary technical and scientific contributions are as follows:

1. **Formalization of the Navigation POMDP:** We provide a rigorous mathematical definition of the visual navigation task as a Partially Observable Markov Decision Process. Crucially, we specify a multi-modal state vector where the global goal is represented as an egocentric relative direction and distance, ensuring a clear goal-oriented input for the agent.

2. **Computable Proxy for Visual Saliency:** We introduce a method for quantifying visual information density that moves beyond local pixel heuristics. By utilizing saliency maps derived from the SelaVPR++ Vision Transformer, we define a computable proxy for landmark distinctiveness that serves as the foundation for the active perception reward.

3. **Two-Phase Developmental Methodology:** This work utilizes a two-stage training framework to decouple reward logic from kinematic complexity. The methodology transitions from a discrete semantic grid-world for initial validation to a continuous motion model governed by non-holonomic unicycle constraints.

4. **Empirical Justification of Multi-Objective Rewards:** We conduct an extensive evaluation of reward formulations to balance the trade-off

between goal progress and visual information density. Through systematic hyperparameter analysis, we provide an empirical justification for the weighting coefficients that stabilize the final flight policy.

5. **Comparative Analysis and Zero-Shot Generalization:** We benchmark the learned policies against a privileged Dijkstra-Pure-Pursuit Oracle. Our analysis identifies the failure modes of behavioural cloning and demonstrates that policy-gradient methods can achieve robust zero-shot generalization in unseen topographies.

6. **Open-Source Research Framework:** We provide a complete, modular software implementation encompassing the custom aerial simulator, the training pipelines, and the evaluation metrics. This framework is publicly hosted on the CIIRC GitLab repository[1] to ensure the absolute reproducibility of our empirical findings and to serve as a baseline for future active perception research.

1.4 Thesis Outline

The work follows a logical progression from mathematical formalization to high-fidelity implementation. Part I introduces the necessary theoretical scaffolding for active perception. Within this part, Chapter 2 formalizes the decision-making process through the lens of RL and POMDPs. This is followed by Chapter 3 which analyzes the constraints of behavioural cloning and the impact of covariate shift. Chapter 4 then bridges these algorithmic concepts with the vision architectures required for robust place recognition.

The subsequent realization in Part II describes the iterative development and evaluation of the navigation system. Chapter 5 presents a conceptual validation using discrete semantic environments to isolate the reward logic. This provides the empirical foundation for Chapter 6 where the system is deployed within a continuous simulation engine operating on authentic aerial imagery. The thesis concludes with a synthesis of the empirical findings in Chapter 7 and a final summary of contributions in Chapter 8.

[1] `https://gitlab.ciirc.cvut.cz/bestavoj/information-driven-uav-navigation`

Part I
Theoretical Foundation

Chapter 2

Reinforcement Learning

Reinforcement learning (RL), often regarded as the third branch of machine learning alongside *supervised* and *unsupervised* learning, is a paradigm that focuses on training an agent to maximize a numerical reward signal through continuous interaction with its environment [3].

To develop an autonomous navigation agent capable of active perception, we must first establish a mathematical framework that captures the interaction between the UAV and its environment. We treat the navigation task as a discrete-time stochastic control process. Let us first define the ideal, fully observable case before addressing the complexities of real-world aerial perception.

2.1 Markov Decision Process (MDP)

To formally model the drone's navigation task, we utilize the framework of a Markov Decision Process (MDP). While the fundamental concepts of MDPs were established by Bellman [4], we follow the modern notation standard *MDPNv1* proposed by Thomas & Okal [5] and the conceptual definitions provided by Sutton & Barto [3].

▶ **Definition 2.1** (Markov Decision Process)**.** *A Markov Decision Process is defined as the tuple* $(\mathcal{S}, \mathcal{A}, \mathcal{R}, P, R, d_0, \gamma)$*, where*

- $\mathcal{S}$ *is the state space, describing all the possible configurations of the environment and the agent.*
- $\mathcal{A}$ *is the action space, containing the set of all the possible actions the agent can perform. The subset of actions available in a specific state* $s \in \mathcal{S}$ *is denoted by* $\mathcal{A}(s) \subseteq \mathcal{A}$*.*
- $\mathcal{R} \subseteq \mathbb{R} \cup \{-\infty, \infty\}$ *is a set of possible rewards that the environment can give to the agent.*

- $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ *is the transition function. This function characterizes the dynamics of the environment by describing the distribution of states at time* $t + 1$ *given the preceding state and action. Formally, for any* $s, s' \in \mathcal{S}$ *and* $a \in \mathcal{A}$*, the value* $P(s, a, s')$ *represents the probability of transitioning to state* s' *given state* s *and action* a*:*

$$P(s, a, s') \doteq \Pr\{S_{t+1} = s' \mid S_t = s, A_t = a\}.$$

The notation $P(s' \mid s, a)$ *is also used interchangeably.*

- $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \times \mathcal{R} \to [0, 1]$ *is the reward function. It describes the probability distribution of rewards at time* $t + 1$ *given the current state* s*, the action* a*, and the subsequent state* s'*. Formally:*

$$R(s, a, s', r) \doteq \Pr\{R_{t+1} = r \mid S_t = s, A_t = a, S_{t+1} = s'\}.$$

Following the notation standard, let $R(r \mid s, a, s') = R(s, a, s', r)$*.*

- $d_0 : \mathcal{S} \to [0, 1]$ *is the initial state distribution.*

- $\gamma \in [0, 1]$ *is the discount factor, determining how the future rewards should be valued.*

The agent-environment interaction is modeled as a discrete-time process for $t = 0, 1, \dots$. At each step, the agent observes the state $S_t \in \mathcal{S}$ and selects an action $A_t \in \mathcal{A}(s)$ according to its policy. The environment responds by transitioning to a new state $S_{t+1} \in \mathcal{S}$ and providing a scalar reward $R_{t+1} \in \mathcal{R}$. This process generates a *trajectory* τ representing the history of the interaction:

$$\tau = (S_0, A_0, R_1, S_1, A_1, R_2, \dots). \tag{2.1}$$

While the reward distribution $R(s, a, s', r)$ from theorem 2.1 provides a complete stochastic description of the environment's feedback, it is often more practical to work with the expected value of the reward signal. We define the expected reward function $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ as the mean of the next reward given the previous state, action, and resulting state:

$$r(s, a, s') \doteq \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a, S_{t+1} = s']. \tag{2.2}$$

A fundamental assumption of the *Markov* Decision Process is the *Markov* property, i.e. the probability of each possible value of S_{t+1} and R_{t+1} must depend only on the preceding state S_t and action A_t. Therefore, the state must include all available information there is about the environment and the past interactions. Formally it must be true that:

$$\begin{aligned} \Pr(S_{t+1} = s' \mid S_t, A_t, S_{t-1}, A_{t-1}, \dots) &= \Pr(S_{t+1} = s' \mid S_t, A_t), \\ \Pr(R_{t+1} = r \mid S_t, A_t, S_{t-1}, A_{t-1}, \dots) &= \Pr(R_{t+1} = r \mid S_t, A_t). \end{aligned} \tag{2.3}$$

However, in real-world aerial navigation, the drone's true state is rarely fully observable through a single camera frame, leading to a violation of the Markov property. This necessitates the extension to Partially Observable MDPs, later discussed in section 2.3.

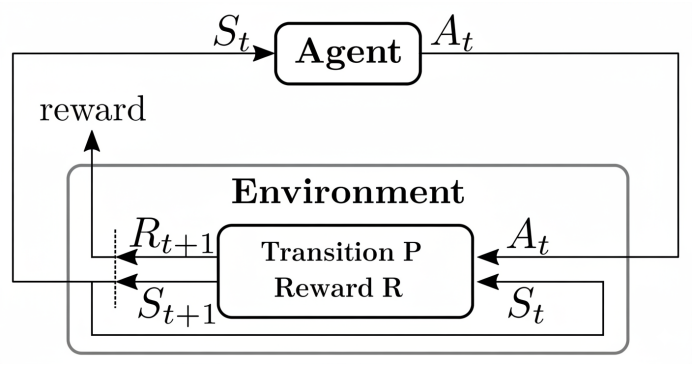


Figure 2.1 The agent-environment interaction in a MDP; adapted from [6]

2.2 Policies, Value Functions, and Bellman Equations

To solve the MDP, the agent requires a formal decision-making strategy to dictate its behaviour across the environment. This strategy is encapsulated by a policy.

▶ **Definition 2.2** (Policy [3]). *A policy π represents the agent's decision-making strategy by mapping states to a probability distribution over the action space. Formally, for any state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$, the policy defines the probability of selecting action a at time t, given that the environment is in state s:*

$$\pi(a \mid s) \doteq \Pr\{A_t = a \mid S_t = s\}. \tag{2.4}$$

The ultimate objective of the agent is not simply to maximize the immediate reward, such as finding a single visually distinctive patch of terrain, but to secure the highest possible cumulative reward over its entire flight trajectory. This cumulative objective is formalized as the expected return.

▶ **Definition 2.3** (Expected return [3]). *Let $\gamma \in [0, 1]$ and $T > t$, including the possibility of $T = \infty$. The general expected return for both episodic and continuous tasks is defined as:*

$$\begin{aligned} G_t &\doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots + \gamma^{T-k+1} R_T \\ &= \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k. \end{aligned} \tag{2.5}$$

To evaluate the quality of a specific policy π, we measure this expected return from a given starting position. This is captured by value functions, which can be defined either for a state or for a specific state-action pair.

▶ **Definition 2.4** (Value function [3]). *The value function, given a state $s \in \mathcal{S}$ and a policy π, is the expected return from an episode when starting in state s and then following the chosen policy π. It is defined as:*

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t \mid S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s\right], \tag{2.6}$$

and the function v_π is called state-value function for policy π.

▶ **Definition 2.5** (Action-Value function [3]). *The action-value function, given a state $s \in \mathcal{S}$, action $a \in \mathcal{A}$ and a policy π, is the expected return when starting from state s, taking action a and then following the chosen policy π. It is defined as:*

$$\begin{aligned} q_\pi(s,a) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] \\ &= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a\right], \end{aligned} \tag{2.7}$$

and the function q_π is called action-value function for policy π.

The core mathematical insight of reinforcement learning is that these value functions can be defined recursively. Known as the **Bellman equations**, they decompose the value of a state into the immediate reward plus the discounted value of the successor state.

▶ **Definition 2.6** (Bellman Equations [3]). *The Bellman equations express the recursive relationship between the value of a state (or state-action pair) and the values of its possible successors:*

$$\begin{aligned} v_\pi(s) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s] \\ &= \sum_{a \in \mathcal{A}} \pi(a \mid s) \sum_{s' \in \mathcal{S}} P(s' \mid s, a)\Big[r(s,a,s') + \gamma v_\pi(s')\Big], \end{aligned} \tag{2.8}$$

$$\begin{aligned} q_\pi(s,a) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] \\ &= \sum_{s' \in \mathcal{S}} P(s' \mid s, a)\Big[r(s,a,s') + \gamma \sum_{a' \in \mathcal{A}} \pi(a' \mid s') q_\pi(s', a')\Big]. \end{aligned} \tag{2.9}$$

Solving the RL problem ultimately means finding a behaviour strategy that outperforms all others. By comparing the state-value functions of different policies, we can define optimality.

▶ **Definition 2.7** (Optimal Policy). *A given policy π is better than or equal to another policy π', i.e. $\pi \geq \pi'$, if $v_\pi(s) \geq v_{\pi'}(s)$ for all $s \in \mathcal{S}$.*

While the existence of an optimal policy π^* is a proven theorem for finite MDPs [3], the transition to continuous state and action spaces, such as the environments used in this work, makes such guarantees non-trivial. In these high-dimensional or continuous cases, we follow the standard convention in Deep Reinforcement Learning: we assume the existence of a near-optimal policy that can be effectively represented by a neural network function approximator [7].

Under the optimal policy π^*, the value functions must satisfy a special form of the Bellman equations. Instead of taking the expected value over all possible actions, the agent strictly selects the action that yields the maximum expected return.

▶ **Definition 2.8** (Bellman Optimality Equations [3])**.**

$$\begin{aligned} v_*(s) &= \max_{a \in \mathcal{A}(s)} q_{\pi_*}(s,a) \\ &= \max_a \mathbb{E}_{\pi_*}[G_t \mid S_t = s, A_t = a] \\ &= \max_a \mathbb{E}_{\pi_*}[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a] \\ &= \max_a \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a] \\ &= \max_a \sum_{s' \in \mathcal{S}} P(s' \mid s,a) \left[r(s,a,s') + \gamma v_*(s') \right], \end{aligned} \tag{2.10}$$

$$\begin{aligned} q_*(s,a) &= \mathbb{E}\left[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \mid S_t = s, A_t = a \right] \\ &= \sum_{s' \in \mathcal{S}} P(s' \mid s,a) \left[r(s,a,s') + \gamma \max_{a'} q_*(s', a') \right]. \end{aligned} \tag{2.11}$$

By computing or approximating $v_*(s)$ or $q_*(s,a)$, the agent can navigate effectively by simply choosing the action that leads to the highest-valued subsequent state [3].

2.3 Partially Observable MDPs (POMDP)

The standard MDP relies heavily on the assumption that the agent has access to the true, fully observable state of the environment S_t. However, in real-world aerial navigation, this is rarely the case.

A UAV equipped with a downward-facing (nadir) camera only receives a localized projection of the environment. If the drone flies over a vast, textureless wheat field, a single camera frame (observation) does not contain enough information to uniquely identify the drone's true global coordinates. This phenomenon, known as *perceptual aliasing*, violates the Markov property, as the current observation alone is insufficient to predict future states [8, 9].

In this setting, the global state S_t becomes a *latent variable*. The agent no longer receives $s \in \mathcal{S}$ directly. Instead, it must infer its position and orientation from a sequence of noisy or ambiguous visual inputs. This transition

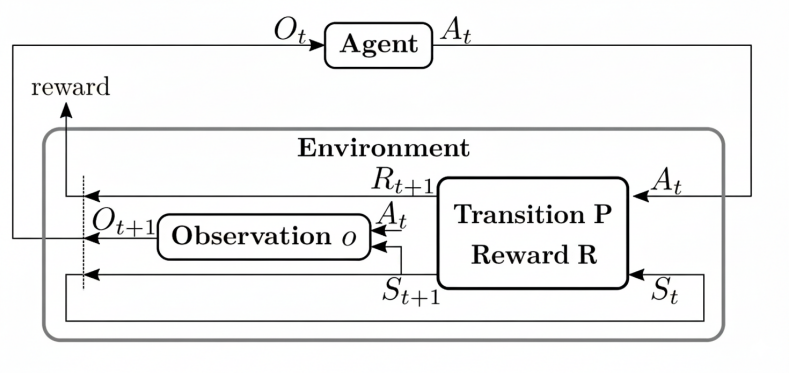


Figure 2.2 The agent-environment interaction in a POMDP; adapted from [6]

from direct state access to indirect inference necessitates the extension of the mathematical framework to a Partially Observable Markov Decision Process (POMDP) [10].

▶ **Definition 2.9** (Partially Observable Markov Decision Process)**.** *A Partially Observable Markov Decision Process is defined as the tuple*

$$(\mathcal{S}, \mathcal{A}, \mathcal{R}, P, R, \Omega, O, d_0, \gamma),$$

where:

- $\mathcal{S}$, $\mathcal{A}$, $\mathcal{R}$, P, R, d_0 *and* γ *are the states, actions, rewards, transition function, reward function, initial state distribution, and discount factor, identical to the MDP formulation from theorem 2.1.*
- Ω *is the observation space of the environment*
- $O : \mathcal{S} \times \mathcal{A} \times \Omega \to [0, 1]$ *is the observation function. It gives the probability of making an observation* $o \in \Omega$ *given a state* $s \in \mathcal{S}$ *and an action* $a \in \mathcal{A}$*.*

As shown in fig. 2.2, the environment emits an observation O_t instead of the true state. To resolve local ambiguities, the agent tracks its *history* H_t [3], which encapsulates all past actions, observations, and rewards:

$$H_t \doteq (A_0, R_1, O_1, \ldots, A_{t-1}, O_t, R_t). \tag{2.12}$$

The history H_t grows continuously with every time step, making it computationally intractable to use directly as an input for decision making. Instead of relying on the full sequence, the agent constructs an internal representation known as a *belief state*.

▶ **Definition 2.10** (Belief State [10])**.** *Because the true environment state* S_t *is hidden, the agent maintains a belief state* b_t*, defined as the probability distribution over all possible states* $s \in \mathcal{S}$ *given the current history* H_t*. We denote the random variable representing the belief state at time* t *as* B_t*. Formally:*

$$b_t(s) \doteq \Pr(S_t = s \mid H_t). \tag{2.13}$$

Since the belief state b_t is a *sufficient statistic* for the history H_t, any optimal policy can be reframed as a mapping from the belief space $\mathcal{B}$ (the continuous space of all possible belief distributions) to the action space: $\pi : \mathcal{B} \to \mathcal{A}$ [10]. This mathematical transformation effectively reduces the POMDP to a fully observable *Belief MDP*, conceptually restoring the Markov property. In high-dimensional visual environments, however, computing and updating the exact continuous probability distribution $b_t(s)$ is analytically intractable. This necessitates the use of practical approximations to manage state uncertainty.

Approximating the Belief State The challenge of recovering the Markov property in environments with limited visibility is typically addressed through two distinct paradigms for state representation. As evaluated by Hausknecht and Stone [11] in their comparison of fixed-history and recurrent architectures, these approaches serve as viable alternatives for approximating the underlying state of a POMDP:

1. **Recurrence:** This approach utilizes Recurrent Neural Networks (RNNs) to maintain a persistent hidden state $\boldsymbol{h}_t$, which acts as a compressed, continuous representation of the entire history H_t. The recursive update mechanism allows the agent to implicitly integrate information across multiple time steps, narrowing the discrepancy between the observation-based value estimate and the true state-based value.

2. **Information Augmentation:** This approach involves enhancing the instantaneous observation space Ω with auxiliary spatial or temporal metadata to form an augmented representation. By explicitly providing the hidden variables required for decision-making, the observation is rendered *pseudo-Markovian*. This transformation effectively reduces the complexity of the POMDP, allowing the decision process to be treated as a standard MDP.

Active Perception in UAV Navigation Within the scope of this work, partial observability is inherent to the aerial navigation task. A critical distinction is maintained between the hidden state $s \in \mathcal{S}$, which encapsulates the UAV's global spatial configuration (e.g., precise coordinates and orientation), and the observation $o \in \Omega$, which comprises only the information accessible via the drone's onboard sensors, e.g. localized view from a downward-facing (nadir) camera. Because such sensors provide only a partial, ego-centric projection of the environment rather than a global overview, the agent must make decisions under state uncertainty.

This creates a state-estimation challenge where a single observation does not contain sufficient information to uniquely determine the agent's position

on a global map. To navigate efficiently, the agent must reconcile this information gap by approximating a belief state – integrating the stream of incoming observations and past actions to maintain a persistent estimate of its position and progress.

2.4 Deep Reinforcement Learning Algorithms

The integration of deep neural networks as function approximators allows Reinforcement Learning to scale to high-dimensional sensory inputs, such as the aerial imagery used in VPR tasks. This section categorizes the algorithmic approaches based on their optimization objectives: value-function estimation, direct policy optimization, and the integration of temporal memory.

Before detailing specific algorithms, it is necessary to establish a notational distinction. Following standard reinforcement learning conventions [3], we denote the *true* theoretical value functions using lowercase letters (e.g., $v_\pi(s)$, $q_\pi(s,a)$). In contrast, we denote the agent's *learned estimates* of these value functions, whether stored in a tabular format or approximated via neural networks, using uppercase letters (e.g., $V(s)$, $Q(s,a)$).

To maintain consistency with standard reinforcement learning literature [12, 13], the foundational algorithms in section 2.4.1 and section 2.4.2 are introduced under the assumption of a fully observable Markov Decision Process, utilizing the true state S_t. In the practical implementation for our UAV navigation task, where only partial observations O_t are available, S_t is replaced by an informative proxy. As discussed in section 2.3, this is achieved either via a recurrent hidden state $\boldsymbol{h}_t$ or by augmenting the observation space to satisfy the Markov property as closely as possible.

2.4.1 Value-Based Learning (DQN)

The value-based paradigm is centered on the estimation of the optimal action-value function $q_*(s,a)$. Under this framework, the behaviour is derived through the systematic evaluation of expected returns for every state-action pair. By maintaining these estimates, the agent selects actions that are greedy with respect to the current learned values. This approach is fundamentally represented by Q-learning, which utilizes the iterative approximation of the Bellman optimality equations to converge toward the optimal action-value function. Such a formulation serves as the theoretical substrate for the more advanced deep reinforcement learning architectures used in complex environments [3].

Q-learning Q-learning is a foundational off-policy Temporal Difference (TD) algorithm. Unlike Monte Carlo methods that require a complete episode return to update values, TD methods estimate action-values incrementally using single-step transitions [3]. The algorithm aims to minimize the temporal difference error between the current estimate $Q(S_t, A_t)$ and the target value, which

represents the immediate reward plus the discounted value of the optimal next action. This results in the standard tabular update formula:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\Big[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)\Big], \quad (2.14)$$

where α is the learning rate, and the maximization term evaluates to zero if S_{t+1} is a terminal state.

Deep Q-learning (DQN) For continuous or high-dimensional state spaces, such as the aerial observations analyzed in this work, maintaining a finite, tabular $Q(S_t, A_t)$ representation is computationally intractable. Instead, the action-value function is approximated via a parameterized function $Q(s, a; \boldsymbol{\theta})$, where $\boldsymbol{\theta} \in \mathbb{R}^{d_{\boldsymbol{\theta}}}$ represents the vector of all learnable parameters, encompassing the weights and biases of a deep neural network [3, 12]. This paradigm was established by Mnih et al. [12], who utilized nonlinear function approximation to train agents on Atari games directly from high-dimensional sensory inputs.

To optimize the parameters $\boldsymbol{\theta}$, the Q-learning update is treated as a stochastic gradient descent problem. The objective is to minimize the Mean Square Error (MSE) using the following loss function, which is defined as the expectation over transitions uniformly sampled from the replay-buffer $\mathcal{M}$:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{M}}\left[\left(r + \gamma \max_{a'} \hat{Q}(s', a'; \boldsymbol{\theta}^-) - Q(s, a; \boldsymbol{\theta})\right)^2\right]. \quad (2.15)$$

While this formulation follows the standard Q-learning update, applying it to deep neural networks is notoriously unstable due to the "deadly triad" of bootstrapping, off-policy learning, and function approximation [3]. To ensure convergence, DQN introduces three stabilizing mechanisms that directly modify the components of eq. (2.15):

- **Experience replay-buffer** – Transitions (S_t, A_t, R_t, S_{t+1}) are stored in a cyclic buffer $\mathcal{M}$. Training batches are sampled uniformly from $\mathcal{M}$, which breaks the temporal correlation between consecutive observations and prevents the network from overfitting to recent trajectories [14].

- **Target network** – Bootstrapping from a continuously updating network creates a highly non-stationary target. DQN mitigates this by introducing a separate target network $\hat{Q}(s, a; \boldsymbol{\theta}^-)$. Its weights $\boldsymbol{\theta}^-$ are frozen and only synchronized with the online network weights $\boldsymbol{\theta}$ after a fixed number of steps [12].

- **ϵ-greedy strategy** – To manage the exploration-exploitation trade-off, the agent follows an ϵ-greedy policy. With probability ϵ_{greedy}, a random action is selected to explore the state space, while with probability $1 - \epsilon_{\text{greedy}}$, the greedy action $\arg\max_a Q(s, a; \boldsymbol{\theta})$ is exploited [3].

While DQN provides a robust framework for extracting policies directly from high-dimensional visual inputs, its reliance on a discrete action space max-operator $\max_a Q$ makes it unsuitable for environments requiring continuous control. As the UAV navigation task in chapter 6 of this thesis requires continuous unicycle kinematics (heading and velocity), a transition from value-based methods to policy-gradient methods is necessary.

2.4.2 Policy-Gradient Methods (PPO)

Unlike value-based methods such as DQN, which approximate action-value functions to derive a greedy strategy, *policy-gradient* methods directly optimize the policy $\pi(a \mid s, \boldsymbol{\theta})$. This approach allows the agent to learn stochastic policy, enabling action selection based on a probability distribution rather than a deterministic estimate of action values. The performance of the policy is defined by a scalar objective function $J(\boldsymbol{\theta})$. To maximize this objective, parameters are updated via gradient ascent:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha \cdot \widehat{\nabla J(\boldsymbol{\theta}_k)}, \tag{2.16}$$

where $\widehat{\nabla J(\boldsymbol{\theta}_k)} \in \mathbb{R}^{d_{\boldsymbol{\theta}}}$ is a stochastic estimate of the gradient of the performance measure with respect to the parameters $\boldsymbol{\theta}_k$ [3]. While $J(\boldsymbol{\theta})$ defines the theoretical performance measure over the entire state-action distribution, modern deep reinforcement learning algorithms typically optimize a *surrogate objective function*, denoted by $\mathcal{L}(\boldsymbol{\theta})$. This notation distinguishes the empirical objective, calculated from a finite batch of samples, from the true expected return [13].

While the policy gradient provides a clear mathematical direction for improvement, vanilla implementations often suffer from high variance in the gradient estimates. To mitigate this, many modern algorithms utilize an *Actor-Critic architecture*. In this framework, the model is split into two components [3]:

- **The Actor:** Represents the policy $\pi(a \mid s, \boldsymbol{\theta})$, responsible for selecting actions based on the current state.
- **The Critic:** Approximates the state-value function $V(s, \boldsymbol{\theta})$, which predicts the expected cumulative reward the agent will receive from state s onwards.

The critic's primary role is to compute the *advantage* function A^π, which serves as the weight for the policy gradient. Formally, the advantage is defined as the difference between the action-value and the state-value:

$$A^\pi(S_t, A_t) = q_\pi(S_t, A_t) - v_\pi(S_t). \tag{2.17}$$

This term measures whether a specific action is better or worse than the policy's average behaviour in a given state [15]. Since the true advantage is unknown at training time, it must be replaced by an estimator $\hat{A}_t$. As the

actor-critic architecture does not maintain a separate action-value network, the advantage is estimated using the one-step temporal difference (TD) residual:

$$\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t), \tag{2.18}$$

where V is the state-value function learned by the critic [3]. To mitigate the high variance associated with single-step transitions, we utilize Generalized Advantage Estimation (GAE) [15]. GAE aggregates these residuals over the finite rollout horizon of length T as:

$$\hat{A}_t = \sum_{l=0}^{T-t-1} (\gamma\lambda)^l \delta_{t+l}, \tag{2.19}$$

where $\lambda \in [0, 1]$ is a smoothing parameter that enables a trade-off between estimation bias and variance. This mechanism is critical for maintaining stable policy updates, especially when processing the high-dimensional visual inputs required for aerial navigation.

Proximal Policy Optimization To address the instability of vanilla policy gradients, *Trust Region Policy Optimization* (TRPO) was introduced, which ensures that policy updates do not deviate too far from the previous policy by enforcing a constraint on the KL-divergence [16]. While mathematically robust, TRPO is computationally expensive due to its reliance on second-order optimization. *Proximal Policy Optimization* (PPO) was subsequently proposed as a first-order alternative that maintains the stability of the "trust region" concept while being significantly easier to implement and more computationally efficient [13]. This efficiency is particularly beneficial for the multi-phase training regime used in this thesis, where the agent must process high-dimensional aerial imagery across thousands of iterations.

First, let us denote the probability ratio $r_t(\boldsymbol{\theta}) = \frac{\pi(A_t|S_t,\boldsymbol{\theta})}{\pi(A_t|S_t,\boldsymbol{\theta}_{old})}$. This ratio shows how much more probable an action is in a new policy compared to an older one. If $r_t(\boldsymbol{\theta}) > 1$, the action is now more likely than before [13]. The novel proposed objective by Schulman et al. is the following:

$$\mathcal{L}^{CLIP}(\boldsymbol{\theta}) = \hat{\mathbb{E}}_t \left[\min \left(r_t(\boldsymbol{\theta})\hat{A}_t,\ \text{clip}(r_t(\boldsymbol{\theta}), 1 - \epsilon_{\text{clip}}, 1 + \epsilon_{\text{clip}})\hat{A}_t \right) \right], \tag{2.20}$$

where ϵ_{clip} is the clip range hyperparameter, usually $\epsilon_{\text{clip}} = 0.2$. The expectation $\hat{\mathbb{E}}_t[\cdot]$ indicates the empirical average over a finite batch of samples [13]. The clipping mechanism restricts $r_t(\boldsymbol{\theta})$ to the interval $[1 - \epsilon_{\text{clip}}, 1 + \epsilon_{\text{clip}}]$, preventing excessively large policy updates. By taking the minimum between the unclipped and clipped objectives, PPO effectively establishes a pessimistic lower bound on policy performance, guarding against catastrophic drops in performance caused by over-correction.[13].

Since the actor and the critic usually share the same parameters between the policy and the value function, we maximize the following combined objective function [13]:

$$\mathcal{L}_t^{CLIP+VF+ENT}(\boldsymbol{\theta}) = \hat{\mathbb{E}}_t \left[\mathcal{L}_t^{CLIP}(\boldsymbol{\theta}) - c_1 \mathcal{L}_t^{VF}(\boldsymbol{\theta}) + c_2 \mathcal{H}[\pi_{\boldsymbol{\theta}}](S_t) \right], \tag{2.21}$$

where c_1 and c_2 are coefficients balancing the individual objectives and the other individual terms are as follows:

- Clipped Surrogate Objective $\mathcal{L}_t^{CLIP}$: As defined in eq. (2.20), it drives the policy update while restricting the change to a "trust region" to prevent catastrophic performance drops.
- Value Function Loss $\mathcal{L}_t^{VF}$: An MSE loss that ensures the critic accurately predicts the expected returns. It is defined as:

$$\mathcal{L}_t^{VF} = \hat{\mathbb{E}}_t \left[(V(S_t, \boldsymbol{\theta}) - V_t^{\text{targ}})^2 \right], \tag{2.22}$$

  where V_t^{targ} is the target return for time step t.
- Entropy Bonus $\mathcal{H}[\pi_{\boldsymbol{\theta}}](S_t)$: This term regularizes the policy to prevent premature convergence by penalizing overly deterministic behaviour. It was originally introduced by Williams and Peng [17] and later popularized in deep reinforcement learning by Mnih et al. [18] within the Actor-Critic framework. In policy gradient methods, the action-distribution variance often collapses before the state space is sufficiently explored, trapping the agent in suboptimal deterministic behaviours. By penalizing excessive certainty (low entropy), this term maintains the stochasticity required to discover high-reward trajectories.

In the context of active perception, this entropy bonus is crucial to prevent the UAV from prematurely converging to a sub-optimal policy, such as hovering endlessly over a single highly-textured building, by forcing it to explore varied terrain.

Continuous Action Space While the PPO objective is agnostic to the underlying action space, adapting the algorithm for continuous UAV kinematics requires redefining the policy $\pi(a \mid s; \boldsymbol{\theta})$ as a probability density function. Following the standard continuous formulation [13], the global parameter vector $\boldsymbol{\theta} = [\boldsymbol{\theta}_\mu, \boldsymbol{\theta}_\sigma]$ is partitioned to decouple the state-dependent mean from the exploration noise. We parameterize the policy as a multivariate Gaussian distribution:

$$\pi(\cdot \mid s; \boldsymbol{\theta}) = \mathcal{N}\left(\boldsymbol{\mu}(s; \boldsymbol{\theta}_\mu), \operatorname{diag}(\boldsymbol{\sigma}^2)\right), \tag{2.23}$$

where the actor network, parameterized by $\boldsymbol{\theta}_\mu$, outputs the state-dependent mean vector $\boldsymbol{\mu}(s; \boldsymbol{\theta}_\mu)$. Conversely, the exploration noise is governed by a

state-independent, learnable vector $\boldsymbol{\theta}_\sigma$ representing the log-standard deviations $\log \boldsymbol{\sigma}$. Optimizing the variance in the logarithmic space ensures that the resulting standard deviations remain strictly positive, preventing numerical instability and gradient explosion during optimization [16]. To guarantee that sampled actions A_t respect the physical constraints of the actuators, the values are clipped to the valid interval $[a_{low}, a_{high}]$ prior to environment interaction.

2.5 Recurrent RL and Memory

Standard reinforcement learning algorithms, including the foundational formulation of PPO rely on feedforward neural networks (e.g., Multi-Layer Perceptrons) that map a fully observable state S_t to an action A_t. However, as established in section 2.3, our UAV navigation task constitutes a POMDP. The agent receives only a localized observation O_t, which inherently violates the Markov property if processed in isolation.

Because the history sequence grows indefinitely with every time step, it is computationally intractable to feed it directly into a standard neural network. A robust solution to this problem is the integration of Recurrent Neural Networks (RNNs) [19]. RNNs are a specialized family of neural networks explicitly designed to process sequential data by maintaining an internal hidden state.

This recursive structure, introduced by Elman [20], allows information to persist across time steps by feeding the hidden state from the previous step back into the network as an additional input. Given a sequence of sequential inputs, an RNN updates a hidden state vector $\boldsymbol{h}_t$ at each time step t. Adapting the standard formulation from Goodfellow et al. [19] to our reinforcement learning framework, the recursive update is defined as:

$$\boldsymbol{h}_t = \phi(\boldsymbol{h}_{t-1}, O_t, \boldsymbol{\theta}), \tag{2.24}$$

where $\boldsymbol{h}_t$ is the hidden state at time step t, O_t is the current observation, and ϕ is a non-linear function parameterized by the network weights $\boldsymbol{\theta}$ that maps a previous hidden state and the current input into a new hidden state.

A critical architectural feature of RNNs is that the parameters $\boldsymbol{\theta}$ are shared across all time steps. This parameter sharing is vital; without it, the network would not be able to generalize across input sequences of arbitrary and variable lengths [19].

In the context of deep reinforcement learning, the hidden state $\boldsymbol{h}_t$ serves as a learned, continuous proxy for the history H_t. While classical POMDP theory relies on a probabilistic belief state $b_t(s)$ to manage uncertainty [10], recurrent neural networks approximate this process through high-dimensional latent representations. As demonstrated by Hausknecht and Stone [11], this mechanism effectively narrows the gap between the observation-based value estimate $Q(O_t, A_t; \boldsymbol{\theta})$ and the true state-based value $Q(S_t, A_t; \boldsymbol{\theta})$. This internal representation effectively creates a *pseudo-Markovian* state, where the

latent vector $\boldsymbol{h}_t$ provides sufficient context to satisfy the Markov property. Consequently, the agent can recover much of the information lost to partial observability and approximate the optimal policy as if the hidden state S_t were directly accessible.

2.5.1 Long Short-Term Memory (LSTM)

Recurrent neural networks are typically trained via the Back-Propagation Through Time (BPTT) [21] algorithm. In this process, the network is unfolded across discrete time steps and standard backpropagation is applied. However, vanilla RNNs struggle to learn long-term dependencies due to the well-documented *vanishing and exploding gradient* problems [22]. As the error gradient is propagated backward through many time steps, repeated matrix multiplications cause it to either shrink exponentially to zero or grow uncontrollably. Consequently, the network either fails to associate events separated by large time gaps or suffers from highly unstable, oscillating weight updates. In the context of aerial navigation, this means a vanilla RNN would quickly "forget" the last distinctive spatial landmark if the UAV spends too many time steps traversing a homogenous, featureless area.

To address these fundamental limitations, Hochreiter and Schmidhuber [22] proposed the Long Short-Term Memory (LSTM) architecture. Instead of relying entirely on a simple recurrent unit with a single hidden state $\boldsymbol{h}_t$, an LSTM introduces a more sophisticated memory block that maintains an internal, long-term cell state $\boldsymbol{c}_t$. This dual-state mechanism enables the network to actively manage its own memory, learning precisely when to capture, retain, or discard information over extended sequences.

To formally define the dimensions of the recurrent layer, let d_{in} represent the dimensionality of the input vector $\boldsymbol{x}_t$ at time step t and d_h denote the dimensionality of the hidden state $\boldsymbol{h}_t$. In the context of our navigation task, $\boldsymbol{x}_t \in \mathbb{R}^{d_{\text{in}}}$ corresponds to the visual observation O_t. The learnable parameters of the LSTM layer consist of the following weight matrices and bias vectors [23]:

- Input Weights: $\boldsymbol{W}^f, \boldsymbol{W}^i, \boldsymbol{W}^o, \boldsymbol{W}^c \in \mathbb{R}^{d_h \times d_{\text{in}}}$, which transform the current observation.
- Recurrent Weights: $\boldsymbol{U}^f, \boldsymbol{U}^i, \boldsymbol{U}^o, \boldsymbol{U}^c \in \mathbb{R}^{d_h \times d_h}$, which transform the previous hidden state.
- Bias Vectors: $\boldsymbol{b}^f, \boldsymbol{b}^i, \boldsymbol{b}^o, \boldsymbol{b}^c \in \mathbb{R}^{d_h}$.

Using these parameters, the update logic for the LSTM is defined by the following system of equations [23]:

$$\begin{aligned}
\mathbf{g}_t^f &= \sigma(\mathbf{W}^f \mathbf{x}_t + \mathbf{U}^f \mathbf{h}_{t-1} + \mathbf{b}^f), \\
\mathbf{g}_t^i &= \sigma(\mathbf{W}^i \mathbf{x}_t + \mathbf{U}^i \mathbf{h}_{t-1} + \mathbf{b}^i), \\
\mathbf{g}_t^o &= \sigma(\mathbf{W}^o \mathbf{x}_t + \mathbf{U}^o \mathbf{h}_{t-1} + \mathbf{b}^o), \\
\tilde{\mathbf{c}}_t &= \tanh(\mathbf{W}^c \mathbf{x}_t + \mathbf{U}^c \mathbf{h}_{t-1} + \mathbf{b}^c), \\
\mathbf{c}_t &= \mathbf{g}_t^f \odot \mathbf{c}_{t-1} + \mathbf{g}_t^i \odot \tilde{\mathbf{c}}_t, \\
\mathbf{h}_t &= \mathbf{g}_t^o \odot \tanh(\mathbf{c}_t),
\end{aligned} \tag{2.25}$$

where $\sigma(x) = (1+e^{-x})^{-1}$ denotes the logistic sigmoid function and $\odot$ represents the Hadamard (element-wise) product. Both σ and tanh are applied element-wise to their respective vectors.

Crucially, the cell state $\boldsymbol{c}_t$ is updated via addition rather than purely multiplication, allowing gradients to flow backward through time unimpeded. The flow of information is regulated by three distinct gates: the *forget gate* ($\boldsymbol{g}_t^f$) determines which outdated memory to discard, the *input gate* ($\boldsymbol{g}_t^i$) controls what new perceptual information from O_t is worth storing, and the *output gate* ($\boldsymbol{g}_t^o$) filters the internal cell state to produce the final hidden state $\boldsymbol{h}_t$ used for the agent's immediate action selection [22, 23].

2.5.2 Recurrent PPO with LSTM

To address the partial observability of the navigation task, the PPO algorithm is integrated with the LSTM architecture. In this recurrent formulation, the policy (Actor) and value function (Critic) are no longer conditioned solely on the current observation O_t, but rather on the internal hidden state of the recurrent layer. This hidden state $\boldsymbol{h}_t$ serves as a compressed representation of the history H_t, effectively providing the necessary context for decision-making [24]. The modified functions are defined as follows:

- **The Actor:** $\pi(A_t \mid O_t, \boldsymbol{h}_{t-1}, \boldsymbol{\theta})$,
- **The Critic:** $V(O_t, \boldsymbol{h}_{t-1}, \boldsymbol{\theta})$.

Crucially, the theoretical objective functions and the clipping mechanism of PPO defined in eq. (2.21) remain invariant to this architectural change [13]. In practice, the LSTM layer acts as a temporal aggregator, situated between the high-dimensional feature extraction (e.g., from a CNN or ViT) and the fully connected heads of the Actor-Critic network [24].

Chapter 3

Imitation Learning

Defining a reward function R that accurately captures desired behaviour without inducing unintended side effects, a challenge known as *reward hacking* [25], is often complex in high-dimensional environments. Imitation learning offers an alternative paradigm where instead of maximizing a scalar reward signal, the agent learns to perform a task by mimicking an expert policy $\pi_E : \mathcal{S} \to \mathcal{A}$ [26].

In the context of this thesis, this expert policy π_E represents a source of optimal or near-optimal behaviour, such as a human pilot or, as utilized in our Phase I and Phase II environments (section 5.2 and section 6.4), an algorithmic pathfinding oracle. The expert provides a set of demonstrations $\mathcal{D}$ consisting of multiple trajectories. While a trajectory was defined in section 2.1 as a sequence of triplets (S_t, A_t, R_{t+1}), the imitation learning framework typically focuses on the state-action mappings that characterise the expert's behaviour. Because the objective is to mimic the expert's decisions rather than maximizing the environment's rewards, the reward signals are omitted from the demonstrations, although they could be included to provide additional information about the expert's performance [26, 27]. Consequently, each trajectory $\tau \in \mathcal{D}$ is simplified into a sequence of state-action pairs

$$\tau = \{(S_0, A_0), (S_1, A_1), \ldots, (S_T, A_T)\}. \tag{3.1}$$

Using the collected dataset $\mathcal{D}$, the imitation learning task is framed as an optimization problem where we seek to find a policy $\pi_{\boldsymbol{\theta}}$ that mimics the behaviour of an expert policy π_E. In the general optimization-based framework for imitation [26], this is defined as the minimization of a divergence between the distributions of state-action pairs encountered by the expert and the agent.

To formally characterize these distributions, we utilize the state-action occupancy measure ρ_π, which represents the discounted probability of visiting a state-action pair (s, a) when following policy π [26, 28].

▶ **Definition 3.1** (State-Action Occupancy Measure)**.** *Given a policy π and an initial state distribution d_0, the occupancy measure $\rho_\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is defined as:*

$$\rho_\pi(s, a) = \pi(a \mid s) \sum_{t=0}^{\infty} \gamma^t \Pr(S_t = s \mid \pi). \tag{3.2}$$

Using this representation, we can map the conceptual goal of matching expert features to a formal optimization problem. The optimal parameters $\boldsymbol{\theta}^*$ for the agent's policy are those that minimize a similarity measure D between the occupancy measures [26]:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} D\left(\rho_{\pi_E} \parallel \rho_{\pi_{\boldsymbol{\theta}}}\right). \tag{3.3}$$

3.1 Behavioural Cloning (BC)

The most fundamental approach to IL is Behavioural Cloning (BC), which treats the problem as a standard supervised learning task. BC learns a direct mapping from states to actions without attempting to recover the underlying reward function or environment dynamics [26].

Given a dataset of expert demonstrations $\mathcal{D}$ containing N_{data} state-action pairs $\{(S_t^{(i)}, A_t^{(i)})\}_{i=1}^{N_{\text{data}}}$, the learner's policy $\pi_{\boldsymbol{\theta}}$ is optimized by selecting a loss function $\mathcal{L}$ appropriate for the action space. Common choices include ℓ_1, ℓ_2 loss, or Kullback-Leibler (KL) divergence. State-of-the-art BC algorithms utilize the Negative Log-Likelihood of the expert's actions [27]:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) = -\frac{1}{N_{\text{data}}} \sum_{i=1}^{N_{\text{data}}} \log \pi_{\boldsymbol{\theta}}(A_t^{(i)} \mid S_t^{(i)}; \boldsymbol{\theta}). \tag{3.4}$$

While this objective function provides a simple and computationally efficient way to train a policy, it relies on the assumption that the state distribution encountered by the agent during deployment will identically match the expert's training distribution. Specifically, it suffers from scenarios in which the agent observes states that it has not seen before [26].

3.2 Dataset Aggregation (DAgger)

To combat the distributional shift inherent in Behavioural Cloning, the Dataset Aggregation (DAGGER) algorithm was proposed [29]. DAGGER is an iterative framework that ensures the agent encounters the states resulting from its own execution errors while still receiving corrective guidance from the expert.

The process begins by training an initial policy $\pi_{\boldsymbol{\theta},1}$ via standard BC on the initial expert dataset $\mathcal{D}$. In each subsequent iteration n, the agent performs the following steps [29]:

1. The agent collects new trajectories by rolling out a mixed policy:

$$\pi_n^{\text{rollout}} = \beta_n \pi_E + (1 - \beta_n)\pi_{\boldsymbol{\theta},n}, \tag{3.5}$$

   where $\beta_n \in [0, 1]$ is a decay parameter controlling the probability of querying the expert during data collection. In early iterations, a high β_n ensures the drone stays within relevant regions of the state space. As training progresses and $\beta_n \to 0$, the agent relies more on its own learned policy, thereby encountering states S_t that may deviate significantly from the original expert distribution.

2. For every state S_t visited during the rollout, the expert π_E is queried to provide the optimal action $A_t = \pi_E(S_t)$.

3. These new state-action pairs are aggregated into the dataset: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_n$, where $\mathcal{D}_n = \{(S_t, \pi_E(S_t))\}$.

4. A new policy $\pi_{\boldsymbol{\theta},n+1}$ is trained on the entire augmented dataset $\mathcal{D}$.

By training on a dataset that is iteratively updated with the expert's reactions to the agent's own errors, DAGGER effectively teaches the agent how to recover and return to the optimal path.

Despite the robustness of DAGGER against distributional shift, its standard formulation rests upon the symmetry of information between the expert and the agent. Specifically, it assumes both policies are conditioned on the same state space $\mathcal{S}$. In complex robotic systems, this symmetry is frequently broken, necessitating a more nuanced approach to imitation.

3.3 Asymmetric Information

To address the informational discrepancy between a simulator-based expert and a real-world agent, we look toward the framework of *Asymmetric Imitation Learning* (AIL) [30]. This setting directly corresponds to the POMDP environment established in section 2.3.

In this framework, the expert policy $\pi_E : \mathcal{S} \to \mathcal{A}$ operates on the privileged global state S_t. Conversely, the agent policy $\pi_{\boldsymbol{\theta}}$ is restricted to the available history of observations H_t. To bridge the gap created by partial observability, the agent relies on a practical representation of H_t to approximate the belief state b_t. Whether utilizing recurrent temporal integration or state space augmentation (as established in section 2.3), the fundamental theoretical challenge remains: the mapping from the true state to the agent's representation is frequently non-bijective.

This loss of information results in *perceptual aliasing* [8] previously discussed in section 2.3. When the agent processes an ambiguous representation that corresponds to multiple possible world states $s \in \mathcal{S}$, it cannot uniquely identify the expert's state-dependent action. Under these conditions of state

uncertainty, Warrington et al. [30] demonstrate that the optimization of the agent converges to an *implicit policy*.

▶ **Definition 3.2** (Implicit policy)**.** *Given a state-conditional expert policy π_E and a history-conditional agent policy $\pi_{\boldsymbol{\theta}}$, the implicit policy $\hat{\pi}$ is defined as the posterior predictive density of the expert's actions, marginalized over the agent's state uncertainty. Formally, for any history H_t and action $a \in \mathcal{A}$:*

$$\hat{\pi}(a \mid H_t) \doteq \mathbb{E}_{s \sim b_t}\left[\pi_E(a \mid s)\right], \tag{3.6}$$

where $b_t(s)$ is the belief state as defined in theorem 2.10. Under the AIL objective, the agent $\pi_{\boldsymbol{\theta}}$ minimizes the divergence from this implicit policy.

Mathematically, the implicit policy calculates the expected action of the expert across all states that the agent currently deems probable. If the expert commands contradictory actions in spatially distinct but perceptually aliased regions, this expectation yields an averaged, sub-optimal control signal.

To analyze the long-term behaviour the agent under a specific policy, a measure that describes which parts of the environment the agent actually visits is needed. We define the *discounted state-belief occupancy* $d^\pi(s, b)$, which represents the probability distribution over state-belief pairs (s, b) weighted by the discount factor γ:

$$d^\pi(s, b) \doteq (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t \Pr(S_t = s, B_t = b \mid \pi). \tag{3.7}$$

The constant $(1 - \gamma)$ provides normalization so that $\sum_{s,b} d^\pi(s, b) = 1$. Using this measure, we can formally define the condition under which an agent can successfully recover the expert's behaviour despite the informational gap. Following Warrington et al. [30], we introduce the concept of *identifiable policy pairs*.

▶ **Definition 3.3** (Identifiable Policy Pair)**.** *Consider an MDP-POMDP pair $\{\mathcal{M}_\mathcal{S}, \mathcal{M}_\Omega\}$ with an expert policy $\pi_E(a \mid s)$ and an agent policy $\pi_{\boldsymbol{\theta}}(a \mid b)$. The pair $\{\pi_E, \pi_{\boldsymbol{\theta}}\}$ is considered **identifiable** if and only if the expected Kullback-Leibler divergence between the expert and agent is zero over the occupancy distribution induced by the agent:*

$$\mathbb{E}_{(s,b) \sim d^{\pi_{\boldsymbol{\theta}}}}\left[D_{KL}\big(\pi_E(a \mid s) \parallel \pi_{\boldsymbol{\theta}}(a \mid b)\big)\right] = 0. \tag{3.8}$$

This thesis addresses the loss of identifiability through *Active Perception* [31] rather than altering the expert policy as in Adaptive Asymmetric DAgger [30]. By evaluating the distinctiveness of observations, the agent can steer its trajectory toward identifiable regions of the state space. This requires extracting robust features from raw image data to resolve perceptual aliasing, shifting the focus from control theory to computer vision and the analysis of neural architectures for scene context.

Chapter 4

Computer Vision and Perception

The theoretical frameworks of reinforcement and imitation learning, detailed in the preceding chapters, rely on the assumption that the agent receives a sufficiently informative representation of its environment. However, in vision-based aerial navigation, the raw observation consists of high-dimensional pixel arrays that do not explicitly contain spatial or semantic meaning. To effectively navigate and address the problem of perceptual aliasing defined in section 2.3, these raw inputs must be processed to extract robust, discriminative features. This chapter outlines the foundations of computer vision utilized to bridge this sensory gap, detailing the underlying neural architectures and their specific application to the task of visual place recognition.

4.1 Neural Architectures for Visual Perception

The transformation of raw image data into compact latent representations is typically achieved through specialized deep learning function approximators. Standard fully connected networks scale poorly with high image resolutions and inherently discard the 2D spatial organization of the input [19]. This section examines the two primary paradigms utilized for visual feature extraction: models based on local spatial hierarchies and models driven by global self-attention mechanisms.

4.1.1 Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNNs) represent a specialized class of deep neural architectures specifically optimized for processing data with a known grid-like topology, such as discrete temporal sequences or images. While the foundational principles of training such networks via backpropagation were

popularized by LeCun et al. [32] for the task of handwritten character recognition, the formal distinction of a CNN lies in the replacement of general matrix multiplication in at least one of its layers with a discrete *convolution* operation [19].

Mathematical Formulation The core of the convolutional layer is a discrete linear operation that transforms an input tensor I using a learnable kernel $K \in \mathbb{R}^{m \times n}$. While the formal definition of a two-dimensional convolution involves a kernel reflection (i.e., indexing $I(i-m, j-n)$), standard deep learning libraries, including those used in this work, implement the operation as a discrete *cross-correlation*:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n) K(m,n), \tag{4.1}$$

where (i, j) are the spatial coordinates of the output feature map S. In the context of function approximation via backpropagation, the distinction between convolution and cross-correlation is purely representational. Since the coefficients of K are optimized from data, the network can implicitly learn a reflected kernel if required by the objective function. Consequently, the omission of the reflection step reduces computational overhead without restricting the model's representational capacity [19]. For the remainder of this thesis, we follow standard nomenclature and refer to the operation in eq. (4.1) as "convolution."

Sparse connectivity In a traditional fully connected layer, every output unit interacts with every input unit, necessitating a weight matrix of size $m \times n$. In contrast, convolutional layers enforce sparse connectivity (or sparse interactions) by utilizing a kernel K that is significantly smaller than the input I. Consequently, each output feature only represents a local neighbourhood of pixels. This drastically reduces the number of operations required to compute the feature maps.

While sparse connectivity limits the number of inputs for each output, parameter sharing (or weight tying) ensures that the same set of kernel parameters is applied across all spatial locations of the input. In a standard neural network, each element of the weight matrix is used exactly once. In a CNN, every member of the kernel is used at every position of the input. This structural prior not only minimizes the memory footprint of the model but also ensures that the layer is *translation equivariant*. Formally, a function f is equivariant to a transformation $\mathcal{T}$ if $f(\mathcal{T}(I)) = \mathcal{T}(f(I))$ [19]. Specifically, if the input I is shifted by a vector Δ, the resulting feature map S is shifted by the same vector Δ. In the context of UAV perception, this means that if a landmark shifts within the camera frame, its corresponding representation in the feature map shifts by the same amount, preserving spatial relationships.

However, CNNs are not equivariant to other transformations, such as rotation or scaling.

Spatial Hierarchies and the Receptive Field A standard convolutional block comprises three sequential stages: a linear convolution, a non-linear activation (e.g., ReLU), and a pooling function [19]. While pooling introduces a degree of local translation invariance, its primary structural role is to downsample spatial resolution, which effectively increases the *receptive field* (the input region that a specific neuron "sees") in subsequent layers [33].

This architecture enables a hierarchical feature extraction process inspired by orientation-selective neurons in biological vision. Early layers extract elementary primitives like oriented edges and corners. Subsequent layers aggregate these local features to represent increasingly complex semantic objects. To understand the global context of a landscape, such as the spatial relationship between a road and a distant forest, a CNN must stack many layers to aggregate information from across the image [33].

However, because these features are built strictly from local neighbourhoods, CNNs exhibit a strong *local bias*. While deep layers have larger receptive fields, they often struggle to capture long-range dependencies across the entire frame. For a UAV traversing repetitive terrain, this reliance on local hierarchies leads to *perceptual aliasing*: if two spatially distinct regions share the same local edge and texture distributions, the CNN's hierarchical representation may be insufficient to disambiguate them.

4.1.2 Visual Transformers (ViT)

The Transformer architecture, originally proposed by Vaswani et al. [34], has become the foundational model in natural language processing (NLP). Dosovitskiy et al. [35] demonstrated that this architecture can be directly applied to visual tasks with minimal modifications by treating image patches identically to text tokens, creating the *Vision Transformer* (ViT).

▶ Note 4.1. To distinguish from the scalar value functions $V(s)$ and $v_\pi(s)$ defined in the Reinforcement Learning chapters, we adhere to the standard Transformer notation utilizing bold uppercase letters to denote matrices.

Image Tokenization and Linear Projection Let $\mathcal{I} \subseteq \mathbb{R}^{H\times W\times C}$ denote the space of all possible images of resolution $H \times W$ with C color channels. To process an input image $\mathbf{I} \in \mathcal{I}$, the ViT first reshapes it into a sequence of flattened 2D patches $\mathbf{x}_p \in \mathbb{R}^{N_{\text{patches}}\times(P^2_{\text{size}}\cdot C)}$, where P_{size} is the fixed patch size and the resulting sequence length is $N_{\text{patches}} = HW/P^2_{\text{size}}$. These high-dimensional pixel arrays are then mapped into a latent space of dimension d_{emb} using a shared linear projection $\mathbf{E} \in \mathbb{R}^{(P^2_{\text{size}}\cdot C)\times d_{\text{emb}}}$ [35]. By applying the same projection matrix identically to every patch, the model learns a spatially

invariant feature extractor while significantly reducing the number of learnable parameters compared to a location-specific mapping.

To maintain the 2D topology of the original image, a set of positional embeddings $\mathbf{E}_{\text{pos}}$ is added to the projected tokens. In classification tasks, a dedicated `[class]` token $\mathbf{x}_{\text{class}} \in \mathbb{R}^{D_{\text{emb}}}$ is prepended to the sequence to aggregate global information. However, for the localization and active perception tasks described in this thesis, the remaining patch tokens are of greater importance. Architectures such as DINOv2 [36] and SelaVPR++ [2] demonstrate that these tokens retain high-resolution spatial features which are lost when only the class token is utilized. The construction of the initial input sequence $\mathbf{z}_0$ is formulated as:

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}}; \mathbf{x}_p^1\mathbf{E}; \mathbf{x}_p^2\mathbf{E}; \dots; \mathbf{x}_p^{N_{\text{patches}}}\mathbf{E}] + \mathbf{E}_{\text{pos}}. \tag{4.2}$$

The Transformer Encoder Block The fundamental building block of the ViT is the Transformer Encoder, which consists of a stack of L identical layers. To facilitate the training of deep architectures, the model deviates from the original *Post-Layer Normalization* (Post-LN) scheme used in [34]. Instead, it adopts a *Pre-Layer Normalization* (Pre-LN) configuration, where the normalization is applied to the input of each sub-layer rather than the output of the residual sum. This ensures that the identity shortcut remains unaffected, allowing for a more stable gradient flow during the early stages of optimization [37].

Each block $\ell \in \{1, \dots, L\}$ consists of a Multi-Head Self-Attention (MSA) module and a Multi-Layer Perceptron (MLP) as follows:

$$\mathbf{z}'_\ell = \text{MSA}(\text{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1}, \tag{4.3}$$

$$\mathbf{z}_\ell = \text{MLP}(\text{LN}(\mathbf{z}'_\ell)) + \mathbf{z}'_\ell, \tag{4.4}$$

where the MLP typically consists of two linear layers with a GELU non-linearity [35]. By interleaving normalization and residual connections, the architecture can scale to a high number of layers without the signal degradation typically observed in purely feed-forward deep networks.

Self-Attention Mechanism To compute the dependencies between patches within the MSA module defined in eq. (4.3), the ViT models the relationships between all patches simultaneously. The input sequence to the sub-layer, $\mathbf{z}_{\ell-1}$, is projected into three distinct matrices: Queries ($\mathbf{Q} = \mathbf{z}_{\ell-1}\mathbf{W}_Q$), Keys ($\mathbf{K} = \mathbf{z}_{\ell-1}\mathbf{W}_K$), and Values ($\mathbf{V} = \mathbf{z}_{\ell-1}\mathbf{W}_V$). Conceptually, the interaction between $\mathbf{Q}$ and $\mathbf{K}$ determines the *relevance* of every patch to every other patch, while $\mathbf{V}$ represents the actual latent information to be propagated. The core attention mechanism is then defined as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}, \tag{4.5}$$

where d_k is the dimension of the keys and queries [34]. The dot product $\mathbf{QK}^\top$ represents a pairwise similarity metric between all tokens. The row-wise softmax converts these similarities into a weight distribution, and the scaling by $1/\sqrt{d_k}$ counteracts the growth of dot-product magnitudes in high-dimensional spaces. Finally, by multiplying this weight matrix by $\mathbf{V}$, the mechanism computes a weighted sum of the value vectors. This ensures that the updated representation for each patch is an aggregation of features from across the entire image, where the contribution of each distant patch is prioritized by its semantic relevance to the current location.

In practice, this approach scales to *Multi-Head Self-Attention* (MSA) to jointly attend to information from different representation subspaces. The input sequence is linearly projected h times, and for each head $i \in \{1, \ldots, h\}$ within layer ℓ, the attention is computed as:

$$\text{head}_{\ell,i} = \text{Attention}(\mathbf{z}_{\ell-1}\mathbf{W}^Q_{\ell,i}, \mathbf{z}_{\ell-1}\mathbf{W}^K_{\ell,i}, \mathbf{z}_{\ell-1}\mathbf{W}^V_{\ell,i}), \tag{4.6}$$

where the dimensions are typically set such that $d_k = d_v = d_{\text{emb}}/h$. The outputs of all heads are concatenated and transformed via a final projection matrix $\mathbf{W}^O_\ell \in \mathbb{R}^{hd_v \times d_{\text{emb}}}$ to form the complete MSA module utilized in eq. (4.3):

$$\text{MSA}(\mathbf{z}_{\ell-1}) = \text{Concat}(\text{head}_{\ell,1}, \ldots, \text{head}_{\ell,h})\mathbf{W}^O_\ell. \tag{4.7}$$

This multi-head structure prevents the attention mechanism from averaging out diverse spatial features, allowing different heads to specialize in distinct semantic attributes, such as local edges or global geometric configurations.

Output Sequence and Global Representation After L layers of transformation, the encoder produces a final latent sequence $\mathbf{z}_L = [\mathbf{z}^0_L; \mathbf{z}^1_L; \ldots; \mathbf{z}^{N_{\text{patches}}}_L]$, where each $\mathbf{z}^i_L \in \mathbb{R}^{d_{\text{emb}}}$. In the original ViT formulation for image classification, the model isolates the state of the `[class]` token to derive a global representation:

$$\mathbf{y} = \text{LN}(\mathbf{z}^0_L), \tag{4.8}$$

where $\mathbf{y}$ is then passed to a linear head for label prediction. This approach assumes that the `[class]` token has successfully aggregated the most discriminative information from all other patches [35]. However, for tasks requiring spatial reasoning or feature extraction, the remaining N_{patches} patch tokens $\{\mathbf{z}^1_L, \ldots, \mathbf{z}^{N_{\text{patches}}}_L\}$ are of equal importance. These tokens retain a one-to-one correspondence with the original spatial grid of the image. As will be discussed in section 4.3.1, this spatial preservation allows for the derivation of localized saliency measures, effectively repurposing the Vision Transformer as a high-resolution feature extractor.

Inductive Bias and Complexity Constraints This sequence-based representation is what allows the ViT to operate with minimal *inductive bias*

compared to other architectures. In machine learning, inductive bias refers to the set of architectural assumptions a model uses to generalize to new data [38].

As established in section 4.1.1, CNNs have a strong spatial inductive bias. They are architecturally hard-coded to assume that local pixels are related (locality) and that patterns are the same regardless of their position (translation equivariance). In contrast, the ViT is designed with minimal image-specific priors [35]. Because the image is processed as a sequence of flattened, independent patches, the model initially has no "knowledge" of the 2D grid or how one patch relates to its neighbour. Instead of being restricted by a local kernel, the architecture is capable of learning correlations between any two patches regardless of their distance. For the task of UAV navigation, this means the model can learn to prioritize unique distant landmarks over local, repetitive textures – a key requirement for solving the problem of perceptual aliasing.

However, this lack of structural bias introduces a significant computational constraint. The self-attention matrix requires computing pairwise similarities between all N_{patches} tokens, resulting in a time and memory complexity of $\mathcal{O}(N_{\text{patches}}^2 \cdot d_{\text{emb}})$ [34]. Because N_{patches} is proportional to the image resolution ($N_{\text{patches}} = HW/P_{\text{size}}^2$), increasing sensor resolution leads to a quadratic increase in processing requirements, presenting a primary bottleneck for real-time UAV applications.

4.2 Fundamentals of Visual Place Recognition

Visual Place Recognition (VPR) is the process of determining whether a current observation corresponds to a previously visited geographic location [39]. In the context of autonomous UAV navigation, this task is typically formulated as an image retrieval problem. Given a query image $I_q \in \mathcal{I}$ captured by the onboard sensor, the objective is to identify the most likely match within a reference database $\mathcal{V} = \{I_1, I_2, \ldots, I_{N_{\text{db}}}\} \subset \mathcal{I}$ consisting of N_{db} geo-tagged images representing the environment's map.

4.2.1 Problem Formulation and Metrics

To enable efficient search in high-dimensional image spaces, we define a feature extraction function $\psi : \mathcal{I} \to \mathbb{R}^{D_{\text{out}}}$ that maps an input image I to a compact, global descriptor of fixed size D_{out}. The retrieval task then reduces to finding the index i^* of the database entry that minimizes a distance metric $d(\cdot, \cdot)$ in the feature space:

$$i^* = \underset{i \in \{1,\ldots,N_{\text{db}}\}}{\arg\min} \; d(\psi(I_q), \psi(I_i)), \tag{4.9}$$

where d is commonly implemented as the Euclidean distance (ℓ_2 norm) or cosine similarity [40]. For a VPR system to be robust, the mapping ψ must

be invariant to conditional changes, such as varying illumination, weather, or seasonal cycles, while remaining highly discriminative to viewpoint changes.

While the global descriptor $\psi(I)$ enables high-speed retrieval, the aggregation process (e.g., Global Average Pooling or NetVLAD) typically discards the relative spatial arrangement of visual features. To mitigate the loss of geometric context, VPR systems often adopt a hierarchical two-stage pipeline. In this paradigm, a set of top-k candidates is first retrieved using global features to narrow the search space. Subsequently, a computationally intensive geometric verification step is performed using local feature correspondences (e.g., SIFT, SURF, or DELF) to confirm the spatial consistency of the match [41].

Performance Metrics The performance of VPR is typically evaluated using the Recall@k (R@k) metric. This value represents the probability that the ground-truth location is captured within the first k candidates retrieved from the database $\mathcal{V}$. Let $\mathcal{Q}$ be a set of query images, and for each $q \in \mathcal{Q}$, let $\{r_1, r_2, \ldots, r_k\} \subseteq \mathcal{V}$ denote the k most similar images according to the distance metric d in the feature space. A query q is successfully localized if at least one retrieved candidate lies within a spatial radius r_{vpr} of the query's true coordinates. Formally, the metric is defined as:

$$\text{R@k} = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \mathbb{I}\left[\min_{i \in \{1,\ldots,k\}} \|\text{pos}(q) - \text{pos}(r_i)\|_2 \leq r_{\text{vpr}} \right], \tag{4.10}$$

where $\mathbb{I}[\cdot]$ denotes the indicator function and $\text{pos}(\cdot) : \mathcal{I} \to \mathbb{R}^2$ represents the mapping to geographic coordinates. In standard benchmarks, the radius r_{vpr} is typically fixed at 25 m [40].

4.2.2 Perceptual Aliasing and Appearance Change

While the objective of VPR is to maximize R@k, the retrieval performance is fundamentally limited by the discriminative power of the feature extraction function ψ. Two primary phenomena degrade this performance: *Appearance Change* and *Perceptual Aliasing* [39].

Perceptual aliasing occurs when the feature extraction function ψ maps observations from spatially distant states to proximal points in the feature space. Formally, for two images I_i, I_j, aliasing exists if:

$$d(\psi(I_i), \psi(I_j)) < \epsilon_{\text{vpr}} \quad \text{while} \quad \|\text{pos}(I_i) - \text{pos}(I_j)\|_2 > \delta_{\text{pos}}, \tag{4.11}$$

where ϵ_{vpr} is the matching threshold and δ_{pos} represents the minimum distance required for a match to be considered a false positive.

Conversely, the system must remain robust to *Appearance Change*, where the environment's visual state evolves while the geographic coordinates remain constant ($\|\text{pos}(I_i) - \text{pos}(I_j)\|_2 \approx 0$). In this scenario, the descriptor must ensure $d(\psi(I_i), \psi(I_j)) < \epsilon_{\text{vpr}}$ despite significant variations in illumination or weather.

These challenges necessitate moving beyond raw pixel-level heuristics towards representations that isolate stable, semantically unique landmarks.

4.3 Semantic Feature Extraction and Adaptation

The performance of an active perception agent is fundamentally limited by its ability to quantify the information content of its surroundings. While the VPR pipeline identifies the most likely location in a database, the navigation policy requires a continuous signal that identifies regions of high localizability. This section outlines the methodologies used to transform raw pixel arrays into discriminative feature representations. We progress from classical geometric heuristics to modern transformer-based architectures that capture the semantic context required for GPS-denied navigation.

4.3.1 Local and Learned Feature Descriptors

Feature descriptors can be broadly categorized into hand-crafted local heuristics and data-driven learned representations. Local heuristics focus on the geometric properties of the image signal, such as intensity changes or corner distributions. In contrast, learned descriptors utilize deep neural networks to extract high-dimensional latent vectors that encode the semantic meaning of the scene.

Local Heuristics (Sobel) A traditional approach to identifying informative regions in an image is through the analysis of first-order derivatives to detect local intensity changes (edges) [42]. For an image $I(x, y)$, the gradient ∇I and its magnitude $M(x, y)$ are defined as:

$$\nabla I = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix}, \quad M(x, y) = \sqrt{g_x^2 + g_y^2}. \tag{4.12}$$

In discrete image processing, these partial derivatives are approximated via convolution with specialized manually-designed kernels. The *Sobel operator* [42] utilizes 3×3 masks to compute the gradient components:

$$K_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}, \quad K_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}. \tag{4.13}$$

While $M(x, y)$ provides a measure of local structural density, it is a purely heuristic proxy for saliency. It lacks semantic context and cannot distinguish between unique landmarks and repetitive textures (e.g., fields or forests), making it susceptible to the perceptual aliasing problem defined in section 4.2.1.

Learned Semantic Features (DINOv2) To address the constraints of manually-designed kernels, modern VPR approaches utilize foundation models such as DINOv2 [36]. By leveraging the ViT architecture and self-attention mechanism discussed in section 4.1.2, DINOv2 facilitates a transition from local textural analysis to global semantic reasoning. Technically, the model is trained through a self-supervised distillation process where a student network is optimized to match the output of a teacher network under varying image augmentations. This is further refined by a masked image modeling objective, which requires the model to infer the content of hidden patches based on their surrounding context. Effectively, this training regime encourages the emergence of object-aware features and a robust understanding of scene geometry without the need for explicit human-annotated labels [36].

The theoretical utility of DINOv2 for the specific demands of UAV navigation is supported by the findings of Keetha et al. [43]. Despite not being trained for the VPR task, DINOv2 encodes visual features that serve as a robust substrate for localization due to their zero-shot transferability. Specifically, their results on aerial datasets, such as Nardo-Air, indicate that DINOv2 features significantly outperform supervised VPR models when deployed in unstructured environments. This suggests that the model effectively bridges the domain gap between ground-level training data and nadir aerial perspectives. Such inherent transferability justifies the use of DINOv2 as the primary feature extractor for aerial navigation, as the resulting latent space is robust to the conditional variances, including illumination and seasonal changes, that typically degrade heuristic-based VPR methods.

4.3.2 Seamless Adaptation (SelaVPR)

While foundation models such as DINOv2 provide robust general-purpose features, they are not inherently optimized for the geometric and conditional variances specific to VPR. The *Seamless Adaptation* (SelaVPR) framework [44] addresses this by refining the pre-trained ViT representations for high-precision localization.

Two-Stage Retrieval As established in section 4.2.1, conventional VPR pipelines are often bottlenecked by the computational cost of geometric verification. The SelaVPR++ iteration [2] mitigates this by replacing traditional local feature matching with a hierarchical re-ranking scheme based entirely on global representations.

The architecture generates two distinct types of descriptors. Initially, a set of candidates is retrieved using compact, low-dimensional binary features. This allows for highly efficient nearest-neighbor search over massive reference databases using Hamming distance. Subsequently, the top-k candidates are re-ranked using high-dimensional floating-point features. These high-dimensional embeddings preserve a richer semantic context and structural geometry, which

is critical for resolving challenging cases of perceptual aliasing during the final verification step.

Saliency Maps Beyond global descriptors, the SelaVPR++ architecture explicitly extracts and utilizes the spatial information preserved within the transformer blocks. In standard Vision Transformer architectures, image-level representations are typically derived from a dedicated classification token (`[class]`). However, because spatial configuration is critical for VPR, SelaVPR++ discards the `[class]` token. Instead, the architecture relies entirely on the output of the final transformer block, which consists of N_{patches} patch tokens. To generate the global descriptors discussed in the previous section, the authors [2] apply Generalized Mean (GeM) pooling [45] over these tokens. GeM pooling uses a parameter p to compute a generalized mean across the feature map, functioning as a tunable middle ground between average and max pooling. Formally, the GeM pooling transforms feature maps in the following way:

$$\mathbf{f} = \left(\frac{1}{N} \sum_{i=1}^{N_{\text{patches}}} \mathbf{z}_i^p \right)^{\frac{1}{p}}, \tag{4.14}$$

where $\mathbf{z}_i$ are the patch tokens.

Localized structural information is recovered by organizing the N_{patches} patch tokens into a spatial feature tensor $\mathbf{F} \in \mathbb{R}^{H' \times W' \times d_{\text{emb}}}$. This representation is analogous to the feature maps found in convolutional architectures. The spatial dimensions satisfy $H' \times W' = N_{\text{patches}}$, and d_{emb} denotes the latent embedding dimensionality. To isolate features relevant to localization, a semantic saliency matrix $\mathbf{M} \in \mathbb{R}^{H' \times W'}$ is derived by computing the mean intensity across the embedding dimension d_{emb} of the adapted feature map $\mathbf{F}$. We define the discrete components of this map as:

$$\mathbf{M}_{u,v} = \frac{1}{d_{\text{emb}}} \sum_{c=1}^{d_{\text{emb}}} \mathbf{F}_{u,v,c}, \tag{4.15}$$

where (u, v) denote the discrete spatial coordinates of the patch tokens. Because the adaptation process explicitly optimizes the network to distinguish static, reliable landmarks from dynamic foreground elements, the resulting matrix $\mathbf{M}$ serves as a localized indicator of spatial utility. Regions with high activation in $\mathbf{M}$ correspond to visually distinct structures, whereas uniform areas yield low activations.

Part II

Realization and Performance Analysis

Chapter 5

Phase I – Conceptual Validation in Semantic Environments

This chapter details the first developmental phase of the proposed system, focusing on the conceptual validation of feature-aware navigation. Before addressing the complexities of continuous UAV kinematics and high-dimensional latent space of Vision Transformers, it was necessary to determine whether a reinforcement learning agent could successfully internalize a multi-objective policy that balances path efficiency with visual feature acquisition.

To achieve this, the navigation problem is abstracted into a discrete grid-world environment. This simplification allows for the isolation of the reward-shaping logic and the POMDP formulation from the stochastic noise of a high-fidelity simulator. By utilizing *OpenStreetMap LandUse* data as a semantic proxy for aerial imagery and spatial gradients (Sobel filters) as a proxy for visual landmarks, we create a tractable benchmark to evaluate the stability of various optimization paradigms.

The experiments presented here demonstrate the learnability of the active perception trade-off and provide the empirical justification for the transition to the continuous aerial engine and SelaVPR++-based descriptors detailed in chapter 6.

5.1 Environment and POMDP Formulation

The environment for Phase I is constructed as a discrete two-dimensional grid world where the agent navigates via 4-way movement. The simulated drone is equipped with a nadir-facing grayscale camera capturing local image crops derived from the *OpenStreetMap LandUse* map tiles dataset [46]. While the underlying data represents categorical land boundaries (such as forests, roads,

and water bodies), the tiles are processed as continuous grayscale rasters to simulate raw visual sensor inputs. To ground this abstract grid into a physically representative aerial environment, the metric scaling is first established. This mapping provides the spatial basis for the formalization of the navigation task within the POMDP framework defined in section 2.3.

5.1.1 Spatial Scale and Data Preprocessing

To ensure the discrete grid-world provides a physically grounded proxy for aerial navigation, the raw *OpenStreetMap LandUse* GeoTIFF rasters are metrically scaled. We assume a simulated flight altitude of $h = 75\,\text{m}$ and a camera field of view of $\alpha_{\text{fov}} = 90°$. Following a pinhole camera projection, the physical ground footprint width W_{ground} is calculated as:

$$W_{ground} = 2h \tan\left(\frac{\alpha_{\text{fov}}}{2}\right). \tag{5.1}$$

Given the discrete neural network input resolution of $W_{in} = 11$ pixels, we define the Ground Sampling Distance (GSD) as $\rho = W_{ground}/W_{in}$. The cadastral maps are resampled via anti-aliased bilinear interpolation to match this target GSD, ensuring that each discrete transition in the state space $\mathcal{S}$ corresponds to a uniform physical displacement.

To prevent the agent from initializing in featureless regions, we employ a variance-based rejection sampling strategy. A candidate $H_a \times W_a$ arena is accepted only if the peak-to-peak normalized pixel intensity strictly exceeds 0.2. This guarantees the presence of at least one semantic boundary, providing the necessary visual stimuli for the active perception task.

5.1.2 State and Action Spaces

The global state space $\mathcal{S}$ comprises the complete environmental grid. The true state $S_t \in \mathcal{S}$ at time t is fully characterized by the global map and the agent's current coordinates. Let $\mathbf{p} : \mathcal{S} \to \mathbb{Z}^2$ be a projection function that maps a state s to its absolute position vector $[x, y]^\top$ within the arena boundaries. The agent's position at time t is denoted by the vector $\mathbf{p}_t = \mathbf{p}(S_t) = [x_t, y_t]^\top$, where x_t and y_t represent discrete Cartesian coordinates.

Because the environment is partially observable, the agent does not have access to the global state. Instead, it receives a local observation $O_t \in \Omega$. To approximate the Markov property and capture temporal dynamics, the observation is constructed as a tuple aggregating visual history, spatial memory, and relative goal information.

The observation space is defined by the physical camera footprint $H_c \times W_c$ and the neural input resolution $H_{in} \times W_{in}$. As established in section 5.1.1, this discrete grid-world allows these resolutions to be equivalent, where $H_{in} = H_c = 11$ and $W_{in} = W_c = 11$. Formally, the observation provided to the

agent is defined as $O_t = (\mathbf{I}_t, \mathbf{D}_t, \mathbf{g}_t)$. To mitigate partial observability, all components utilize a temporal stack of the $N_f = 4$ most recent steps to create a short-term delay embedding. The observation O_t integrates three specialized input modalities:

- $\mathbf{I}_t \in \mathbb{R}^{N_f \times H_{in} \times W_{in}}$: A visual tensor stacking grayscale frames. The pixel data is a crop centered at $\mathbf{p}_t$, processed via Sobel filters to extract spatial gradients.
- $\mathbf{D}_t \in \mathbb{R}^{N_f \times H_{in} \times W_{in}}$: A visitation density tensor stacking local crops of the visitation grid. To embed trajectory history into the observation, a Gaussian distribution centered at the agent's position is incrementally added to a global density grid at each time step. Stacking these crops allows the model to internalize the direction and velocity of previous exploration.
- $\mathbf{g}_t \in \mathbb{R}^{6N_f}$: The low-dimensional telemetry vector. We define the instantaneous state as $\tilde{\mathbf{g}}_t = [d_t, \Delta x_t, \Delta y_t, x_t, y_t, e_t]^\top$. The final input $\mathbf{g}_t$ is the flattened concatenation $[\tilde{\mathbf{g}}_t, \tilde{\mathbf{g}}_{t-1}, \ldots, \tilde{\mathbf{g}}_{t-N_f+1}]^\top$. This vector includes the normalized distance to the goal d_t, the relative direction $(\Delta x_t, \Delta y_t)$ unit vector, the agent's normalized absolute coordinates $\mathbf{p}_t = [x_t, y_t]^\top$ (serving as a global positional embedding for boundary awareness), and the remaining battery life/fuel $e_t \in [0, 1]$.

Action Space For this phase, the action space $\mathcal{A}$ is a discrete set of four primitive movements. We map the agent's output indices $\{0, 1, 2, 3\}$ to the cardinal directions:

$$\mathcal{A} = \{\text{East, North, West, South}\}. \tag{5.2}$$

Transitions $P(s' \mid s, a)$ are assumed to be deterministic, where each action results in a one-cell shift in the corresponding direction. To ensure the agent remains within the defined arena, movement is constrained by the grid dimensions $H_a \times W_a$. If a selected action would result in a transition outside these bounds (i.e., $x_{t+1} < 0$, $x_{t+1} \geq W_a$, $y_{t+1} < 0$, or $y_{t+1} \geq H_a$), the agent's position remains unchanged such that $\mathbf{p}_{t+1} = \mathbf{p}_t$.

5.1.3 Reward Design

The reward function R_t is designed to solve a multi-objective optimization problem: minimizing the path length to the goal while maximizing the density of observed visual features. We define R_t as a piecewise function to distinguish

between the dense rewards provided during navigation and the sparse rewards triggered by terminal transitions:

$$R_t = \begin{cases} R_{\text{success}}, & \text{if } S_t \in \mathcal{S}_{\text{goal}}, \\ R_{\text{fail}}, & \text{if } t \geq T_{\max}, \\ R_t^{(\text{step})}, & \text{otherwise}, \end{cases} \tag{5.3}$$

where the composite navigational reward $R_t^{(\text{step})}$ is defined using static weighting coefficients:

$$R_t^{(\text{step})} = w_{\text{dist}} \cdot R_t^{(\text{dist})} + w_{\text{view}} \cdot R_t^{(\text{view})} + R_{\text{time}} + R_t^{(\text{border})}, \tag{5.4}$$

and where:

- $\mathcal{S}_{\text{goal}} = \{s \in \mathcal{S} \mid \|\mathbf{p}(s) - \mathbf{p}_{\text{goal}}\|_2 < \epsilon_{\text{goal}}\}$ is the set of terminal goal states.
- $T_{\max} = 300$ is the episode horizon. Reaching this limit triggers a truncation, preventing the agent from accumulating infinite negative time penalties.
- R_{success} and R_{fail} are terminal sparse rewards.
- $R_{\text{time}} = -0.01$ is a constant step penalty applied to encourage the agent to find the most efficient path between visual clusters.
- w_{dist} and w_{view} are hyperparameters balancing the trade-off between goal-seeking and feature-following behaviour.

Distance and Boundary Penalties The distance reward $R_t^{(\text{dist})}$ represents the change in Euclidean distance to the goal between successive steps. Unlike a sparse goal reward, this provides a dense signal for the agent:

$$R_t^{(\text{dist})} = d_{t-1} - d_t, \tag{5.5}$$

where d_t is the current distance. Note that $R_t^{(\text{dist})} < 0$ if the agent moves away from the target, forcing the policy to weigh the negative cost of deviation against the positive gain from visual features.

Furthermore, to maintain boundary awareness within the $H_a \times W_a$ arena, we define a border penalty $R_t^{(\text{border})}$. This penalty is activated if the agent attempts an action that would result in a collision with the arena limits, effectively leaving the state S_t unchanged:

$$R_t^{(\text{border})} = -\mathbb{I}[\mathbf{p}_t = \mathbf{p}_{t-1}]. \tag{5.6}$$

Consumable View Reward To formalize the concept of active perception in this discrete phase, we modeled high-frequency terrain features as a consumable resource. Let $\mathcal{F}_0 \in [0,1]^{H_a \times W_a}$ be the global *Feature Availability Map*. This map defines the finite visual resources available for collection:

$$\mathcal{F}_0(\mathbf{p}) = \begin{cases} 1, & \text{if } E_{global}(\mathbf{p}) > \delta, \\ 0, & \text{otherwise,} \end{cases} \tag{5.7}$$

where E_{global} is the edge magnitude computed over the entire arena and δ is a noise filtering threshold.

The locally observed visual edges $E_t(\mathbf{u}) \in [0,1]^{H_c \times W_c}$ at a relative pixel location $\mathbf{u}$ are defined using spatial image gradients:

$$E_t(\mathbf{u}) = \text{clip}\left(\frac{\sqrt{G_x(\mathbf{u})^2 + G_y(\mathbf{u})^2}}{C_{\max}},\ 0,\ 1 \right), \tag{5.8}$$

where G_x and G_y represent the gradient maps obtained by convolving the input image with 3×3 Sobel kernels. The normalization constant $C_{\max} = 4.0$ represents the theoretical maximum gradient magnitude of the filter.

At each time step t, the agent collects rewards based on the detected edges and the remaining uncollected resources. We define the reward collected at a specific coordinate $\mathbf{u}$ within the arena as the product of the feature availability, the detected edge magnitude, and the centering Gaussian mask G to incentivize the agent to center the features within its field of view:

$$C_t(\mathbf{u}) = \mathcal{F}_t(\mathbf{u}) \cdot E_t(\mathbf{u}) \cdot G(\mathbf{u}). \tag{5.9}$$

Consequently, the total view-based reward $R_t^{(\text{view})}$ at time t is calculated by summing the rewards over the agent's current field of view:

$$R_t^{(\text{view})} = \frac{1}{N_{\text{ref}}} \sum_{\mathbf{u} \in \text{FOV}(\mathbf{p}_t)} C_t(\mathbf{u}), \tag{5.10}$$

where N_{ref} is a normalization constant representing the reward obtained from a perfectly centered step-edge. The operator $\text{FOV}(\mathbf{p})$ defines the set of grid coordinates visible to the nadir camera when centered at position $\mathbf{p} = [x, y]^\top$:

$$\text{FOV}(\mathbf{p}) = \left\{ [u, v]^\top \in \mathbb{Z}^2 : \begin{matrix} x - \lfloor W_c/2 \rfloor \le u \le x + \lfloor W_c/2 \rfloor, \\ y - \lfloor H_c/2 \rfloor \le v \le y + \lfloor H_c/2 \rfloor \end{matrix} \right\}. \tag{5.11}$$

To prevent infinite cyclic harvesting (reward hacking), the availability map is updated at each step:

$$\mathcal{F}_{t+1}(\mathbf{p}) = \max(0,\ \mathcal{F}_t(\mathbf{p}) - C_t(\mathbf{p})), \tag{5.12}$$

where $C_t(\mathbf{p})$ is the reward harvested at coordinate $\mathbf{p}$ at time t.

5.2 Algorithmic Baseline (Dijkstra Oracle)

In standard reinforcement learning, agents are often evaluated by their total accumulated reward. However, because the reward function design in this environment underwent continuous evolution (changing weights between distance and visual features), raw reward is not a stable metric for comparing different agent architectures. Furthermore, unlike standard supervised learning, there is no straightforward loss metric such as RMSE or cross-entropy.

Instead, it is necessary to establish a theoretical baseline – an Oracle – against which the agent's trajectory will be compared. To find this optimum baseline, the environment is treated as a fully observable graph, and the navigation task is formulated as a cost minimization problem.

Visual Density Estimation To represent the environment as a searchable structure, we define a directed graph $\mathcal{G} = (\mathcal{S}, \mathcal{E}_{\text{grid}})$, where the set of vertices corresponds to the state space $\mathcal{S}$ (the grid cells), and $\mathcal{E}_{\text{grid}}$ denotes the set of valid 4-way transitions between adjacent cells. We define the visual salience $\nu(\mathbf{p}) \in [0, 1]$ as the normalized density of significant Sobel edges within the agent's field of view when positioned at coordinates $\mathbf{p}$. This is computed by applying a uniform box filter to the global edge map E_{global}:

$$\nu(\mathbf{p}) = \text{clip}\left(\frac{1}{\rho \cdot A_{\text{cam}}} \sum_{\mathbf{u} \in \text{FOV}(\mathbf{p})} \mathbb{I}[E_{\text{global}}(\mathbf{u}) > \delta],\ 0,\ 1\right), \tag{5.13}$$

where:

- $\mathbb{I}[\cdot]$ is the indicator function, identifying pixels where the spatial gradient magnitude exceeds the noise threshold $\delta = 0.1$.
- $A_{cam} = H_c \times W_c$ is the total area of the camera footprint.
- $\rho = 0.75$ is a target saturation constant, ensuring $\nu(s)$ reaches 1.0 even if the field of view is not entirely filled with edges.

Cost Minimization and the Oracle Baseline To establish a theoretical optimum against which the agents can be compared, we formulate the navigation task as a spatial cost minimization problem. We define a continuous spatial cost landscape $c(\mathbf{p})$ across the arena. The cost at any position $\mathbf{p}$ balances the base distance penalty against the lack of visual features to account for perceptual aliasing:

$$c(\mathbf{p}) = 1.0 + \exp\left(\eta \cdot (1 - \nu(\mathbf{p}))\right), \tag{5.14}$$

where $\nu(\mathbf{p})$ denotes the visual salience at position $\mathbf{p}$ and the hyperparameter η dictates the trade-off between path efficiency and feature acquisition. This

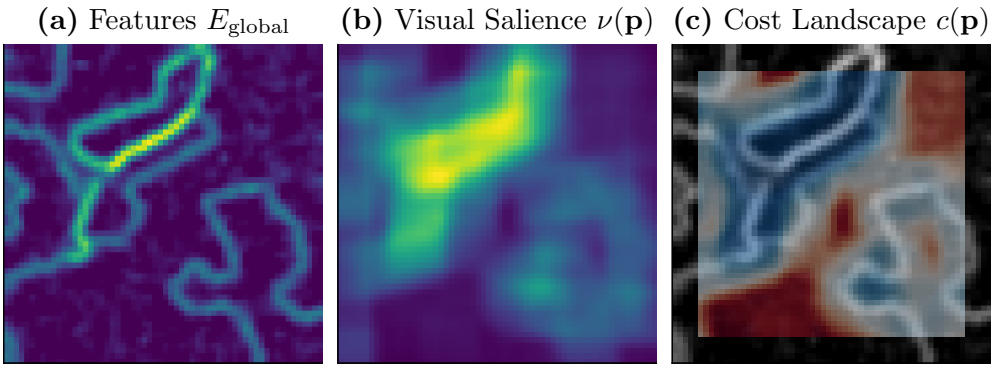


Figure 5.1 Generation of the feature-aware cost landscape for the Dijkstra Oracle over a 60×60 grid. **(a)** The global feature map E_{global} extracted via Sobel filtering. **(b)** The visual salience $\nu(s)$ representing the local density of significant edges. **(c)** The resulting cost landscape $c(\mathbf{p})$ overlaid on the original features. The dark boundary represents the infinite-cost constraint ($\lfloor W_c/2 \rfloor$); this ensures the Oracle plans trajectories only where a valid observation O_t can be fully sampled within $\mathcal{S}$.

sensitivity parameter was empirically fixed at $\eta = 10.0$ to ensure that the Oracle baseline demonstrates a perceptible bias toward feature-rich corridors while avoiding pathologically long detours driven by isolated, high-salience pixels. For a continuous trajectory ξ, the path cost $\mathcal{C}(\xi)$ is the line integral over this scalar field:

$$\mathcal{C}(\xi) = \int_{\xi} c(\mathbf{p})\, dl, \tag{5.15}$$

where dl represents the differential arc length along the path ξ.

The Feature-Aware Oracle is defined as the agent that finds the optimal trajectory $\xi^* = \arg\min_{\xi} \mathcal{C}(\xi)$. To solve this computationally, the environment is treated as a fully observable grid graph $\mathcal{G} = (\mathcal{S}, \mathcal{E}_{grid})$ where edge weights approximate eq. (5.15) between adjacent cells. This discrete minimization problem is solved using Dijkstra's algorithm to yield the baseline ξ^* for every evaluation episode. As visualized in fig. 5.1c, this formulation warps the shortest-path geometry to follow high-salience corridors and provides a deterministic ground truth for feature-aware navigation.

5.3 Evaluated Architectures

As outlined in section 2.4, we employ a deep neural network to approximate either the action-value function $Q(O_t, a; \boldsymbol{\theta})$ or a stochastic policy $\pi(a \mid O_t; \boldsymbol{\theta})$. To ensure a fair comparison across different learning paradigms (DQN, PPO, and IL), a shared feature extraction backbone is utilized, while the terminal layers are adapted to the specific requirements of each algorithm. The overall architecture can be seen in the fig. 5.2.

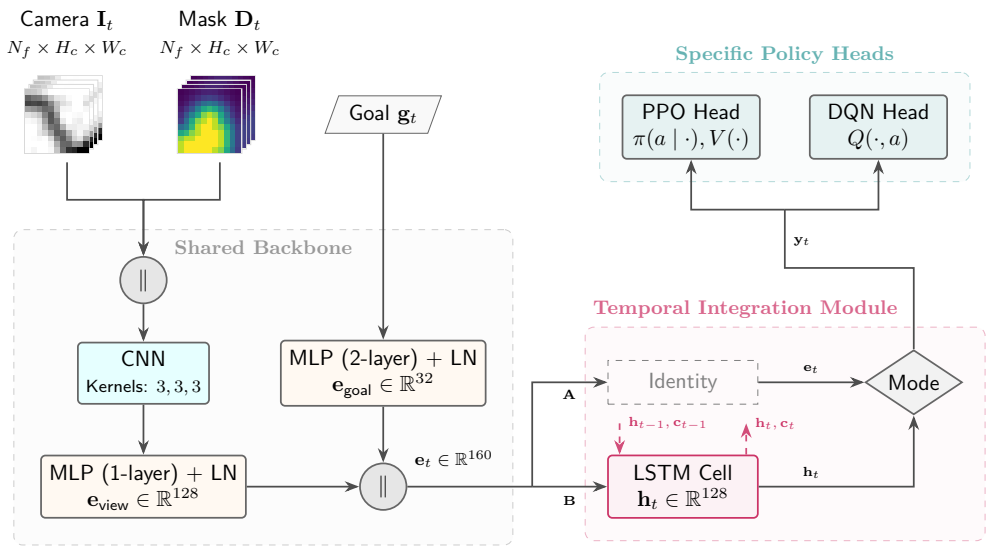


Figure 5.2 Architectural overview of the Phase I. model. The shared backbone $\boldsymbol{\theta}_{\text{backbone}}$ fuses visual observations $(\mathbf{I}_t, \mathbf{D}_t)$ and navigational guidance $(\mathbf{g}_t)$ into a multi-modal embedding $\mathbf{e}_t \in \mathbb{R}^{160}$. This representation is processed by the *Temporal Integration Module* in one of two evaluated configurations: (A) an Identity pass-through for feed-forward baselines, or (B) an LSTM recurrent bottleneck that outputs a hidden state $\mathbf{h}_t \in \mathbb{R}^{128}$ to capture long-term temporal context. The resulting feature vector $\mathbf{y}_t$ is interfaced with algorithm-specific heads for policy (π, V) or value-based (Q) optimization.

5.3.1 Shared Multi-Modal Backbone

The backbone is designed as a multi-modal encoder that fuses high-dimensional visual observations with low-dimensional navigational telemetry. The observation $O_t = (\mathbf{I}_t, \mathbf{D}_t, \mathbf{g}_t)$ is processed through two specialized branches:

1. **Visual Branch (CNN):** The input consists of a concatenated tensor $\in \mathbb{R}^{2N_f \times H_{in} \times W_{in}}$ representing the temporal stack of N_f grayscale camera frames and their corresponding visitation masks. The encoder utilizes a three-layer convolutional neural network (CNN) with a stride of 2 in the final layer to downsample the spatial features. This is followed by a linear projection layer and *Layer Normalization* to produce a latent visual embedding $\mathbf{e}_{\text{view}} \in \mathbb{R}^{128}$.

2. **Navigational Branch (MLP):** The goal-oriented vector $\mathbf{g}_t$ (containing distance, direction, and state information) is processed by a two-layer MLP. This branch ensures that critical navigational cues are not lost when fused with the higher-dimensional visual features. It outputs a guidance embedding $\mathbf{e}_{\text{goal}} \in \mathbb{R}^{32}$.

The final feature vector $\mathbf{e}_t = [\mathbf{e}_{\text{view}}; \mathbf{e}_{\text{goal}}]$ serves as a 160-dimensional joint representation of the agent's current observation.

5.3.2 Temporal Integration

To address the partial observability inherent in the local camera view, we evaluate two configurations for temporal processing. We define $\mathbf{y}_t$ as the final latent feature vector passed to the decision-making heads:

- **Identity:** The embedding is passed directly to the heads ($\mathbf{y}_t = \mathbf{e}_t$). In this setup, the model relies strictly on the N_f frames stacked in the input to capture short-term motion cues.
- **Recurrent (LSTM):** An LSTM layer with $d_h = 128$ units is inserted as a temporal bottleneck. The embedding $\mathbf{e}_t$ serves as the input $\mathbf{x}_t$, while the LSTM maintains a hidden state $\mathbf{h}_t$ and cell state $\mathbf{c}_t$. The heads receive $\mathbf{y}_t = \mathbf{h}_t$, which acts as a learned proxy for the belief state b_t. This enables the policy to internalize long-term dependencies, such as the overall trajectory, which is critical when traversing textureless regions where current observations O_t are perceptually aliased.

5.3.3 Algorithm-Specific Heads

The final layers map the feature vector $\mathbf{y}_t$ to the action or value space. The architecture of these heads depends on the optimization objective:

- **DQN Head:** A 2-layer MLP that maps $\mathbf{y}_t$ to the action-value set $\mathbb{R}^{|\mathcal{A}|}$. Each output represents $Q(O_t, a_i; \boldsymbol{\theta})$, the expected discounted return for selecting action a_i.
- **PPO Actor-Critic Head:** This module employs two parallel branches. The *actor* branch maps $\mathbf{y}_t$ to a 4-dimensional logit vector, defining a categorical distribution $\pi(a \mid O_t)$ over the discrete actions. Simultaneously, the *critic* branch reduces $\mathbf{y}_t$ to a scalar value $V(O_t)$ for advantage estimation.

The Imitation Learning variants do not introduce a distinct head. During behavioural cloning, the actor head of the PPO configuration is retained and optimised via cross-entropy loss against the expert action labels $\pi_E(O_t)$. The architectural distinction between IL and PPO therefore lies solely in the training objective and data source, both of which are detailed in section 5.4. This shared parameterization also enables the hybrid IL pre-training and RL fine-tuning approach, where the actor head's weights are carried over directly into the subsequent PPO stage.

5.4 Learning Paradigms

With the architectural backbone established, we now define the specific optimization strategies used to train the agent's policy. The objective of this

comparative analysis is to identify which learning paradigm best handles the multi-objective reward structure within the discrete grid-world environment. We evaluate three distinct approaches: value-based learning (DQN) as a foundational baseline, policy-based optimization (PPO) for improved training stability, and imitation learning (DAGGER) to address the exploration challenges inherent in this task. By utilizing the shared backbone across all paradigms, we ensure that performance differences stem strictly from the algorithmic dynamics and their ability to internalize the active perception trade-off between path efficiency and visual feature acquisition.

5.4.1 Value-Based (DQN)

As established in section 2.4.1, the Deep Q-Network was evaluated as the foundational baseline for the discrete action space $\mathcal{A}$. The primary challenge in the expansive arena is the sparsity of terminal goal rewards. Under a standard ϵ-greedy exploration strategy, random walks are highly unlikely to traverse the environment successfully. Consequently, the agent relies heavily on the dense distance reward $R_t^{(\text{dist})}$ to maintain directional progress.

To stabilize the approximation of the Q-function, two specific mechanisms were critical. First, the Experience Replay Buffer $\mathcal{M}$ was sized to hold 10^5 transitions. This was necessary because adjacent grid observations O_t and O_{t+1} are highly correlated. Uniform sampling from $\mathcal{M}$ ensured that the mini-batches used for gradient descent contained a diverse distribution of terrain types, preventing the CNN from overfitting to local semantic textures. Second, the Target Network $\boldsymbol{\theta}^-$ was updated every 300 steps to mitigate the moving target problem.

5.4.2 Policy-Based (PPO)

While DQN provided initial conceptual validation, preliminary empirical observations during the hyperparameter search indicated that value-based methods exhibited high sensitivity to the ratio of reward weights w_{dist} and w_{view}. In configurations where visual exploration was heavily prioritized, the greedy nature of the Q-learning update occasionally led to the agent becoming trapped in local reward maxima, such as infinite loops around clusters of high-saliency features.

To investigate whether a policy-gradient approach could offer improved training stability under these multi-objective constraints, we introduced Proximal Policy Optimization (section 2.4.2) as an alternative paradigm. The actor head was configured to output a categorical distribution over the discrete action space $\mathcal{A}$. The use of a clipped surrogate objective is intended to prevent large, destabilizing policy updates when the agent encounters novel regions of the *OSM LandUse* dataset. Furthermore, an entropy bonus coefficient c_2 was integrated into the objective function. This design choice aims to maintain

a degree of stochasticity during training, encouraging the drone to continue exploring visually unique patches and preventing premature convergence to a deterministic shortest path. By evaluating both value-based and policy-based methods, we seek to determine which optimization objective better internalizes the navigation-perception trade-off in the discrete domain.

5.4.3 Imitation Learning (DAgger)

Even with dense reward shaping, RL agents waste immense computational resources discovering basic navigational competence through random exploration. To bypass this sample inefficiency, we applied the DAgger algorithm (section 3.2).

This approach relies on an asymmetric information structure. While the agent's policy $\pi_{\boldsymbol{\theta},n}$ only has access to the localized, partially observable crop $O_t \in \Omega$, we utilize the discrete Dijkstra Oracle as the privileged expert π_E. The Oracle has access to the fully observable graph $\mathcal{G}$ and the global true state $S_t \in \mathcal{S}$ to provide optimal action labels. During training, the agent executes its current imperfect policy $\pi_{\boldsymbol{\theta},n}$ to generate trajectories. For every state visited, π_E provides the ground-truth discrete action A_t, and the resulting pairs are aggregated into a growing dataset $\mathcal{D}_n$. This explicitly addresses the covariate shift problem that plagues standard behavioural cloning by teaching the agent how to recover from its own navigational errors within the discrete grid.

5.4.4 Hybrid Approach: IL Pre-training with RL Fine-tuning

While the Dijkstra Oracle π_E provides optimal trajectories within the discrete graph formulation, pure imitation learning often demonstrates brittle performance in long-horizon navigation tasks. Because the expert defines optimality solely through a static cost field, an agent trained via Behavioural Cloning or DAgger lacks the intrinsic goal-seeking robustness needed to recover when execution errors push it into unexplored regions of the state space. This often manifests as a failure to converge on the final goal coordinates in the expansive arena, even when the agent successfully follows visual features for the majority of the episode.

To address this, we implemented a hybrid training pipeline that leverages the sample efficiency of IL to bypass the RL "cold-start" problem, followed by policy-gradient optimization to refine the agent's behaviour. This transition is made possible by the shared architectural design of the feature extractor (fig. 5.2). Since the PPO and IL variants utilize an identical multi-modal backbone and categorical actor head, the weights $\boldsymbol{\theta}$ converged during the final DAgger iteration serve as a high-competence initialization for the PPO agent.

By initializing the policy in this high-performing region of the parameter space, we avoid the unstable early exploration phase of RL. The subsequent

PPO fine-tuning then allows the agent to break away from the Oracle's heuristic constraints. In this stage, the agent optimizes the true multi-objective reward signal R_t, discovering localized feature-harvesting behaviours that utilize the spatial memory $\mathbf{D}_t$, a dynamic context that the global graph-search heuristic, being a stationary cost-map solver, cannot represent.

5.5 Experimental Results

This section presents the empirical evaluation of the Phase I active perception system, focusing on the agent's capability to navigate discrete semantic environments while balancing spatial efficiency with visual feature acquisition. To ensure a rigorous and reproducible assessment, we first establish a standardized experimental configuration and define a set of quantitative metrics that benchmark the learned policies against deterministic spatial baselines. Utilizing this framework, the subsequent analysis is designed to address three core research questions (RQs):

1. **Objective Balancing (RQ 1):** How does the tunable trade-off between path efficiency and active feature acquisition manifest in the agent's trajectory, and at what point do we observe diminishing returns?
2. **Environmental Complexity (RQ 2):** To what extent does a structured curriculum of increasing task difficulty mitigate policy collapse when transitioning from synthetic topological curves to noisy semantic boundaries?
3. **Learning Paradigm Comparison (RQ 3):** Among the evaluated architectures (value-based DQN, policy-gradient PPO, and imitation learning via DAgger), which paradigm demonstrates the highest stability and competence in internalizing this multi-objective POMDP?

5.5.1 Experimental Setup and Baselines

All experiments were conducted utilizing the CIIRCluster and the ClusterFIT computational infrastructures. The development, hyperparameter tuning, and final policy evaluations for the entire research project cumulatively required over 2100 independent training and evaluation jobs. This represents a total computational footprint of approximately 5,800 GPU-hours and over 94,000 CPU-hours.

Environment Constraints During all Phase I experiments, the UAV operated at a simulated altitude of 75 m over an arena of $W_a = H_a = 60$ grid cells. The downward-facing camera captured a constrained local observation window of $W_c = H_c = 11$ pixels. This limited field of view was a deliberate design choice to enforce strict partial observability, necessitating active visual search

Table 5.1 Hyperparameter configurations utilized across the Phase I optimization paradigms.

Hyperparameter	DQN	PPO / LSTM-PPO	DAgger
Learning Rate (α)	1×10^{-4}	3×10^{-4}	3×10^{-4}
Batch Size	64	256	128
Discount Factor (γ)	0.99	0.99	-
Optimization Epochs	-	10	10
Buffer / Rollout Size	100 000	8 192*	-
Target Update Freq.	1 000 steps	-	-
Exploration Strategy	$\epsilon \to 0.05$ (linear)	Entropy Coef: 0.01	-
Clip Range (ϵ_{clip})	-	0.2	-
DAgger Iterations	-	-	10 (at 50k steps)
Total Training Steps	3×10^6	3×10^6	600×10^3

* Represents the total rollout buffer (16 envs $\times$ 512 steps) per gradient update.

to locate landmarks. To prevent infinite loitering behaviours, each episode was strictly bounded to a maximum horizon of $T_{\max} = 300$ steps.

Training Configuration and Hyperparameters To ensure a rigorous and unbiased comparison across the evaluated optimization paradigms, all models were implemented using the *Stable Baselines3* framework and shared an identical multi-modal feature extraction backbone (fig. 5.2). The decision heads for both the policy and value networks consisted of a two-layer MLP with 128 hidden units each, initialized using orthogonal weights.

Given the maximum episode horizon of 300 steps, a discount factor of $\gamma = 0.99$ was universally applied. This provided the agents with an effective planning horizon of approximately 100 steps ($\frac{1}{1-\gamma}$), which was empirically sufficient to navigate the 60×60 grid while prioritizing distal goal progress over immediate, localized visual rewards.

The Adam optimizer was utilized across all configurations. To ensure convergence and to account for variance in both environment initialization and network weight distribution, the pure RL baselines (DQN, PPO, LSTM-PPO) were trained for 3×10^6 environment interactions across five independent random seeds. The distinguishing, algorithm-specific hyperparameters are summarized in table 5.1.

5.5.2 Evaluation Framework

To rigorously assess the performance of the trained agents, we establish a standardized evaluation framework. Let an evaluated trajectory ξ be defined as a sequence of spatial coordinates $(\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_K)$. The segment length

between consecutive points is denoted as $\Delta l_i = \|\mathbf{p}_i - \mathbf{p}_{i-1}\|_2$. In the context of the Phase I discrete grid, this simplifies to $\Delta l_i \equiv 1$ for all valid 4-way actions. Using this mathematical groundwork, the agents are compared against two deterministic baselines using a suite of metrics designed to quantify the trade-off between navigational efficiency and active perception.

Baselines for Comparison The agents are benchmarked against the following reference trajectories:

1. **Naive shortest path** ξ_{naive} – The geometrically shortest route across the discrete grid, determined via Dijkstra's algorithm using a uniform cost map ($c(\mathbf{p}) = 1.0$). To ensure a deterministic and visually straight trajectory, a minor penalty based on the perpendicular distance from the start-to-goal vector resolves any path ties. The search terminates at the boundary of the goal radius ϵ_{goal}.
2. **Feature-aware optimum** ξ^* – The path minimized using Dijkstra's algorithm from Section 5.2.

Success Rate An episode is considered successful if the agent terminates within the goal radius ϵ_{goal} before the horizon $T_{\max}$. We define the empirical success rate of a given policy π, evaluated over a set of N_{ev} evaluation episodes, as:

$$\text{Succ}(\pi) \doteq \frac{1}{N_{\text{ev}}} \sum_{j=1}^{N_{\text{ev}}} \mathbb{I}\left[\|\mathbf{p}_K^{(j)} - \mathbf{p}_{\text{goal}}^{(j)}\|_2 < \epsilon_{\text{goal}}\right], \tag{5.16}$$

where $\mathbb{I}[\cdot]$ is the indicator function and $\mathbf{p}_K^{(n)}$ is the final position of the n-th trajectory. In the following analysis, this metric is reported as a percentage.

In our implementation, we discretize the trajectory into K linear segments. We approximate the differential dl using the Euclidean segment lengths Δl_i. By assuming the cost field varies linearly between sampled points, the total cost is estimated as the sum of the average endpoint costs weighted by the segment length:

$$\hat{C}(\xi) = \sum_{i=1}^{K} \frac{c(\mathbf{p}_{i-1}) + c(\mathbf{p}_i)}{2} \Delta l_i. \tag{5.17}$$

Using this estimator, we define the **Optimality Score**, denoted as $\text{Opt}(\xi)$, as the ratio of the Oracle's optimal cost to the agent's incurred cost:

$$\text{Opt}(\xi) = \frac{\hat{C}(\xi^*)}{\hat{C}(\xi_{\text{agent}})}. \tag{5.18}$$

An ideal agent perfectly matching the Oracle's spatial efficiency achieves a score of $\text{Opt}(\xi) = 1.0$.

Path Extension This metric quantifies the navigational detour taken to acquire visual features. We define the **Path Extension**, denoted as $\mathrm{Ext}(\xi)$, as the ratio of the agent's trajectory length $L(\xi)$ to the length of the grid-optimal naive path $L(\xi_{\text{naive}})$:

$$\mathrm{Ext}(\xi) = \frac{L(\xi)}{L(\xi_{\text{naive}})} = \frac{\sum_{i=1}^{K_{\text{agent}}} \Delta l_i}{\sum_{j=1}^{K_{\text{naive}}} \Delta l_j}, \tag{5.19}$$

where K_{agent} and K_{naive} denote the total number of discrete transitions in the agent's trajectory and the naive baseline trajectory, respectively. By utilizing the grid-optimal baseline $L(\xi_{\text{naive}})$ rather than the continuous Euclidean distance, we account for grid discretization artifacts and the goal termination radius. This formulation guarantees that $\mathrm{Ext}(\xi) \geq 1.0$ for any successful episode. A value of 1.0 indicates perfect spatial efficiency, while values > 1.0 represent exploratory detours driven by the active perception objective. Episodes where the agent fails to reach the goal are excluded from this metric.

Feature Capture To quantify the efficacy of the active perception strategy, we define the total visual interest $I(\xi)$ accumulated along a trajectory. This is calculated as the distance-weighted sum of the visual salience $\nu(\mathbf{p})$ encountered by the agent:

$$I(\xi) = \sum_{i=1}^{K} \nu(\mathbf{p}_{i-1}) \Delta l_i. \tag{5.20}$$

The **Feature Capture** metric, denoted as $\mathrm{Cap}(\xi)$, is then expressed as the ratio of the interest captured by the agent to that captured by the feature-aware Oracle (ξ^*):

$$\mathrm{Cap}(\xi) = \frac{I(\xi_{\text{agent}})}{I(\xi^*)} \quad \text{for } I(\xi^*) > 0. \tag{5.21}$$

In degenerate episodes where the environment is entirely devoid of significant visual features (i.e., $I(\xi^*) = 0$), the capture ratio becomes mathematically undefined. To prevent artificial inflation of the evaluation metrics, such episodes are excluded from the aggregated mean calculations. A value of $\mathrm{Cap}(\xi) = 1.0$ indicates that the agent's trajectory is as visually informative as the Dijkstra-optimal path, whereas values > 1.0 indicate the agent actively prioritized feature-dense terrain beyond the Oracle's baseline.

5.5.3 Objective Balancing

To assess how this balance is internalized by the policy, we perform a sensitivity analysis on the reward coefficients w_{dist}, w_{view}, and R_{success}. This investigation allows us to map the behavioural transition from a baseline shortest-path policy to an "Active Perception" strategy that actively prioritizes salient landmarks.

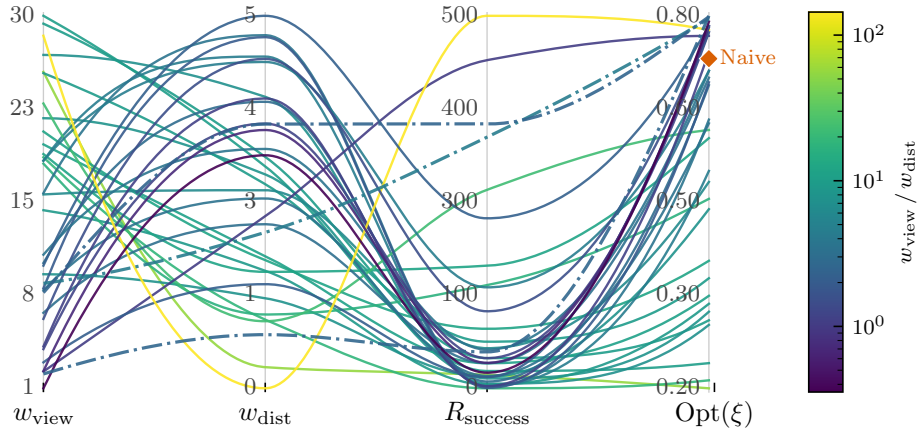


Figure 5.3 Parallel coordinate plot illustrating the global sensitivity of the navigation policy to the reward hyperparameters. Trajectories are color-coded by the $w_{\text{view}}/w_{\text{dist}}$ ratio. The final axis Opt(ξ) highlights the performance relative to the naive baseline ($\diamond$). Dashed traces represent the top 3 configurations by Opt(ξ).

By systematically varying these weights, we establish the empirical bounds of the navigation-perception trade-off. This analysis is critical for identifying the "competence corridors" where the agent successfully reaches the goal while maintaining high visual confidence.

Global Sensitivity and Performance Corridors To map the sensitivity of the policy to the reward structure without an exhaustive grid search, we employed Bayesian hyperparameter optimization utilizing PPO as the underlying learning backbone. Specifically, the Tree-structured Parzen Estimator (TPE) algorithm, implemented via the *Optuna* framework [47], was used to sample the continuous parameter space efficiently. The objective was to maximize the trajectory optimality score Opt(ξ).

A total of 35 unique reward configurations were evaluated using the TPE sampler with 10 initial startup trials. The search space was bounded to $w_{\text{view}} \in [0.5, 30.0]$ and $w_{\text{dist}} \in [0.1, 5.0]$ using linear sampling, while the terminal reward was sampled logarithmically within $R_{\text{success}} \in [10.0, 500.0]$. Each configuration was trained for 1.5×10^6 steps to ensure convergence of the multi-modal backbone.

The resulting mapping between these reward coefficients and the policy performance is visualized in fig. 5.3. A clear "competence corridor" emerges for configurations with moderate weight ratios where $w_{\text{view}}/w_{\text{dist}} \in [1, 5]$ (indicated by the dark-purple and dark-blue traces). In this regime, the agent successfully balances the dense distance reward with localized visual incentives. However, as the ratio increases into the intermediate range $w_{\text{view}}/w_{\text{dist}} \in$

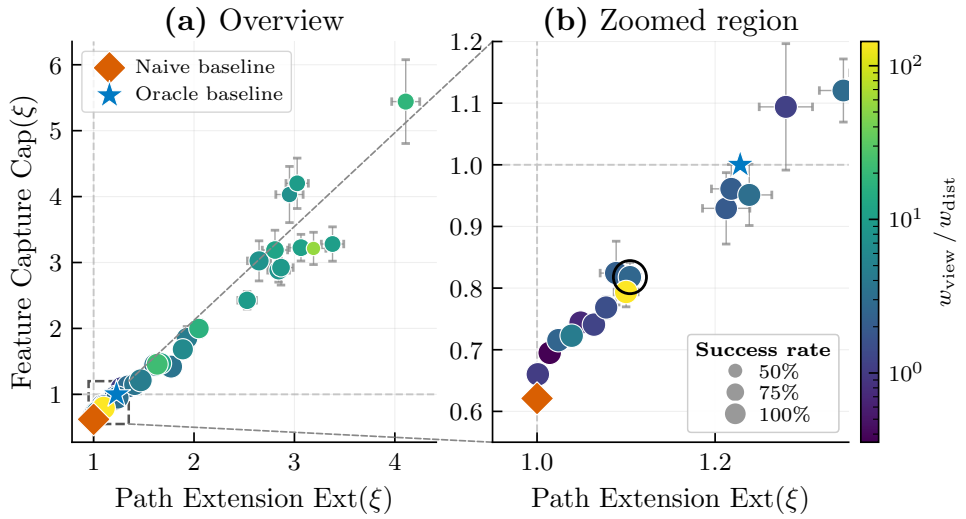


Figure 5.4 Sensitivity analysis of the navigation policy to the $w_{\text{view}}/w_{\text{dist}}$ weight ratio. **(a) Overview** of the behavioural space, illustrating the divergence toward "reward hacking" (excessive loitering) at high visual weights. **(b) Detail of the competence region** highlighting agents that achieve superior feature capture relative to both the Dijkstra Oracle ($\star$) and the naive straight-line baseline ($\diamond$). Marker diameters denote the empirical success rate $\text{Succ}(\pi)$, illustrating the increased risk of episode truncation ($T_{\max}$) as the path extension $\text{Ext}(\xi)$ grows. Error bars represent ± 1 standard error across $N_{\text{ev}} = 500$ evaluation episodes per configuration. The selected configuration identified via the filtering protocol in section 5.5.3 is highlighted with a ring.

$[10, 30]$ (teal and light green traces), we observe a significant degradation in optimality. These configurations fall well below the naive baseline ($\diamond$), providing empirical evidence of the "loitering" problem where the agent becomes trapped by localized visual features. Finally, a single run with an extreme ratio $w_{\text{view}}/w_{\text{dist}} \approx 100$ achieves high optimality by prioritizing the global objective. This is explained by its maximum terminal reward $R_{\text{success}} = 500$, which acts as an overpowering goal signal that suppresses loitering despite the high visual incentive.

Navigational Detours and Feature Acquisition To further quantify the efficiency of the learned policies, we evaluated the trade-off between the incurred navigational overhead and the acquired visual interest. Figure 5.4 illustrates the relationship between the path extension $\text{Ext}(\xi)$ and the feature capture ratio $\text{Cap}(\xi)$.

As observed in fig. 5.4a, configurations biased heavily toward visual exploration (teal and green markers) result in degenerate, reward-hacking policies. The agent exploits the consumable nature of the visual rewards, executing localized cyclic patterns to artificially inflate $\text{Cap}(\xi)$ well beyond the Oracle's

theoretical limit. This comes at the cost of navigational competence, evidenced by trajectory extensions up to 4.0× the naive path and higher trajectory variance. Conversely, fig. 5.4b isolates the competence region where the agent operates as intended. The dark purple and blue configurations bridge the behavioural gap between the naive shortest path ($\diamond$) and the Dijkstra Oracle ($\star$). While the extreme-ratio configuration (yellow marker) also reaches this high-optimality region, its behaviour is distinct. Due to the high terminal reward, this policy adopts a predominantly goal-seeking strategy. It successfully avoids loitering and surpasses the naive baseline as well.

Policy Selection and Results To identify a baseline for the Phase I comparative analysis, we systematically filtered the results of the hyperparameter sweep. The goal was to isolate a policy that maximizes perceptual awareness without compromising the primary navigation task. We designate the configuration identified via this selection process as the **Heuristic-Augmented Static Policy (HASP)**, as it utilizes fixed reward weights to optimize for heuristic visual edges. The selection followed a three-stage filtering process:

1. **Mission Reliability:** We first discarded all configurations where the success rate fell below 95%. This eliminated agents in the high-w_{view} regime that prioritize infinite feature accumulation (loitering) over goal-seeking.
2. **Perceptual Awareness:** We removed "naive" policies where the Feature Capture Cap(ξ) was lower than 80% of the Oracle baseline, ensuring the agent actually internalizes the active perception objective.
3. **Optimality Maximization:** From the remaining candidates, we selected the configuration that maximized the Optimality Score Opt(ξ) to ensure detours strictly followed the spatial cost-minimization logic rather than random exploration.

Based on these criteria, the configuration with $w_{\text{view}} \approx 2.20$, $w_{\text{dist}} \approx 0.86$, and $R_{\text{success}} \approx 57.35$ (ratio $\approx 2.6 : 1$) was selected. The performance of the resulting HASP agent against the established baselines is summarized in table 5.2.

Table 5.2 Performance comparison of the best-performing PPO policy identified during the hyperparameter sweep. Values represent the mean ± one standard deviation computed across $N_{\text{ev}} = 500$ evaluation episodes.

Method	Success [%]	Opt(ξ) (↑)	Ext(ξ) (↓)	Cap(ξ) (↑)
Naive Shortest Path	100.0	0.728 ± 0.213*	1.000 ± 0.00	0.621 ± 0.293*
HASP (sweep)	98.8	**0.775** ± 0.20	**1.104** ± 0.26	**0.817** ± 0.53
Oracle Baseline	100.0	1.000 ± 0.00	1.228 ± 0.423*	1.000 ± 0.00

* Baselines vary across episodes due to randomized start/goal positions and map layouts.

The empirical results validate the core hypothesis of this phase. By accepting a 10.4% average increase in path extension, the policy achieves a superior optimality score compared to the naive baseline (0.775 vs. 0.728). This indicates that the agent's detours are not random loitering but mathematically efficient maneuvers within the cost landscape. Furthermore, the substantial increase in feature capture over the naive path (0.817 vs 0.621) proves the agent's ability to actively prioritize salient terrain.

5.5.4 Ablation Study of the Curriculum Strategy

Initially, we hypothesized that the high-dimensional visual noise and complex junction topologies of the OSM LandUse dataset would prevent the RL agent from discovering the active perception trade-off from scratch. Early exploratory runs suggested a risk of policy degeneration, where agents either ignored visual features entirely to minimize path length or became trapped in localized high-reward loops. To address this, we developed a granular four-stage curriculum designed to gradually bridge the gap between local line-following and autonomous feature-seeking [48]:

1. **Stage I: Single-line Tracking**: A randomized B-spline curve defines the path that the agent must follow between the start and goal points.
2. **Stage II: Multi-line Discrimination** – The environment was populated with intersecting curves, teaching the agent to maintain tracking across junctions.
3. **Stage III: Off-line Navigation** – Start and goal points were placed outside the generated curves, forcing the agent to transition from pure tracking to active visual search.
4. **Stage IV: Real-World Generalization (OSM)** – The agent was transferred to the raw OSM LandUse dataset, navigating genuine semantic boundaries.

To evaluate the efficacy of this approach, we trained three variants of the PPO agent: the full 4-stage curriculum, a 2-stage curriculum (skipping directly from Stage I to Stage IV), and a direct training baseline initialized exclusively on the target OSM domain. The networks were trained continuously, retaining their weights across stage transitions. Each configuration was evaluated across five independent random seeds.

The empirical results reveal a significant disparity between training convergence and evaluation robustness. As visualized in fig. 5.5, the stage boundaries triggered severe domain shift, evidenced by the sharp drops in the Optimality Score at 0.75×10^6 and 1.5×10^6 timesteps. While the curriculum agents' training metrics eventually recovered to match the direct training baseline by the

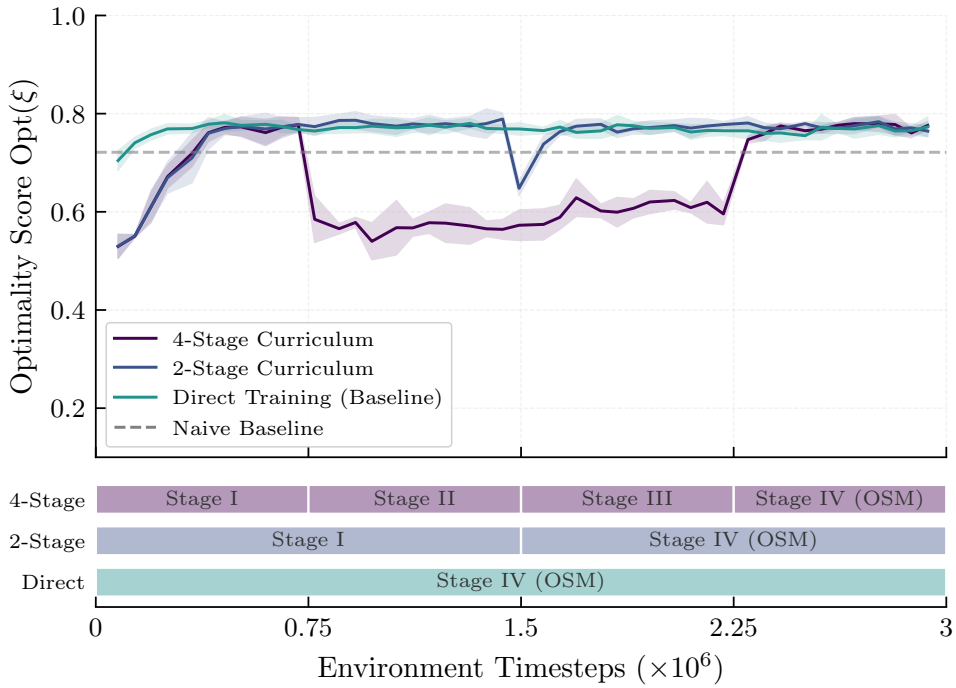


Figure 5.5 Comparative evaluation of the curriculum learning strategy against direct domain optimization ($N_{\text{sd}} = 5$ seeds). The **upper panel** tracks the Optimality Score Opt(ξ) over training timesteps, with shaded regions representing the standard deviation. The **lower panel** synchronizes the training timeline across the synthetic (Stages I–III) and real-world (Stage IV) environments. Performance drops at stage boundaries indicate the impact of domain shift on the policy's stability compared to the direct training baseline.

end of Stage IV, rigorous deterministic evaluation (summarized in table 5.3) exposes severe overfitting.

The data indicates that the curriculum-based agents acquired an overly strong prior for continuous edge-following during the synthetic stages. When transferred to the noisy semantic blocks of the OSM LandUse tiles, this prior manifested as pathologically exploratory behaviour. The 4-Stage and 2-Stage agents achieved artificially high Feature Capture rates (1.019 and 0.966, re-

Table 5.3 Quantitative evaluation of curriculum strategies on the OSM target domain. Values represent the mean ± one standard deviation computed across $N_{\text{sd}} = 5$ independent training seeds ($N_{\text{ev}} = 500$ evaluation episodes per seed).

Strategy	Success [%] (↑)	Opt(ξ) (↑)	Ext(ξ) (↓)	Cap(ξ) (↑)
4-Stage Curriculum	83.8 ± 10.7	0.646 ± 0.09	1.225 ± 0.09	**1.019** ± 0.16
2-Stage Curriculum	76.6 ± 2.8	0.589 ± 0.02	1.233 ± 0.02	**0.966** ± 0.02
Direct Training	**98.9** ± 1.0	**0.779** ± 0.01	**1.057** ± 0.02	0.750 ± 0.03

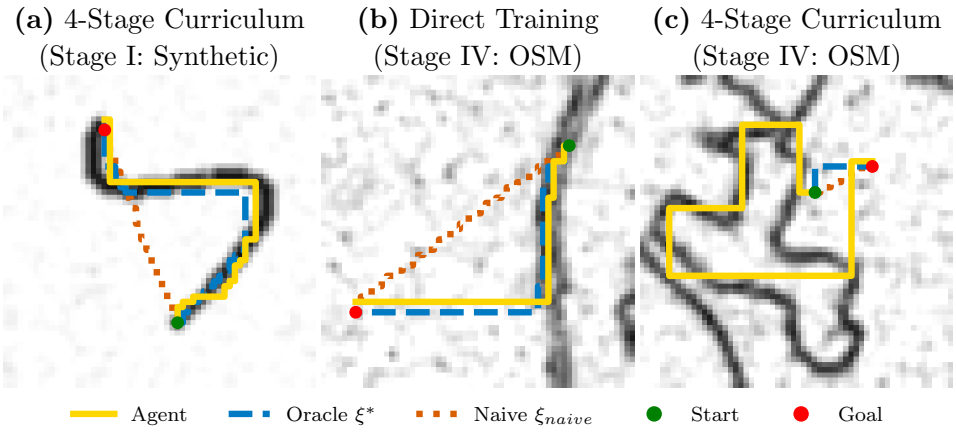


Figure 5.6 Qualitative trajectory analysis demonstrating the failure of curriculum transfer due to domain shift. The background represents the inverted Sobel magnitude of the terrain. **(a)** The 4-stage curriculum agent, evaluated at the conclusion of Stage I, successfully tracks synthetic B-splines during Stage I. **(b)** The direct training baseline successfully navigates noisy OSM data, deviating from the naive path to approximate the interest-optimal Oracle. **(c)** The curriculum-trained agent exhibits pathological behaviour when transferred to the target domain, executing suboptimal, localized detours that fail to reach the goal, which prevents the feature extractor from generalizing to the noisy semantic boundaries of the OSM dataset.

spectively) but suffered from substantial Path Extension (≈ 1.22). By prioritizing localized feature acquisition over global spatial progress, these agents frequently exhausted the episode time horizon, causing their mission success rates to plummet to 83.8% and 76.6%.

In contrast, the Direct Training baseline successfully balanced the multi-objective reward. While its feature capture was lower (0.750), it achieved the highest mission reliability (98.9%) and the best Optimality Score (0.779), demonstrating a much more robust internalization of the global spatial cost landscape.

To qualitatively illustrate this pathological behaviour and the resulting domain shift, fig. 5.6 visualizes the agents' navigational paths across different training environments. It should be noted that the trajectory in fig. 5.6a represents the policy at the conclusion of Stage I, prior to any domain transfer. While the agent successfully internalizes localized tracking in this sterile environment, the filters co-adapted to these clean gradients fail to interpret the irregular semantic boundaries of the target domain. This results in erratic, localized loitering and an inability to reach the goal (fig. 5.6c). Conversely, the direct training baseline (fig. 5.6b) processes the semantic noise effectively, maintaining global goal-seeking behaviour while successfully detouring to acquire visual features.

Consequently, the curriculum learning strategy was rejected. The results demonstrate that direct optimization on the target domain is strictly superior. It forces the policy to learn the active perception trade-off natively, avoiding the brittle priors induced by synthetic pre-training.

5.5.5 Comparative Analysis of Learning Paradigms

To identify the most suitable optimization strategy for the active perception POMDP, a comparative analysis was conducted across the evaluated learning paradigms: value-based (DQN), policy-gradient (PPO, LSTM-PPO), imitation learning (DAgger), and a hybrid initialization approach (DAgger + PPO). To ensure an unbiased comparison, all models utilized the shared multi-modal backbone detailed in fig. 5.2 and were evaluated under the best-performing reward configuration identified in section 5.5.3.

Learning Dynamics and Training Stability The learning dynamics are depicted in fig. 5.7. The pure reinforcement learning approaches (DQN, PPO, and LSTM-PPO) were evaluated over 3×10^6 environment interactions. They exhibited standard convergence behaviours, gradually overcoming the sparse terminal rewards to surpass the naive shortest-path baseline. PPO and its recurrent variant demonstrated smoother, monotonic improvements due to the clipped surrogate objective, whereas the value-based DQN exhibited higher variance during early exploration before stabilizing.

Conversely, the pure imitation baseline (DAgger) was trained over 600×10^3 interactions (an initial dataset of 100×10^3 steps followed by 10 interactive rounds of 50×10^3 steps). To evaluate if imitation could bypass the RL cold-start problem, the hybrid DAgger + PPO agent was initialized using the weights from the highest-performing policy iteration identified during the DAgger evaluation. To prevent the imitation weights from being ruined by a randomly initialized critic, we included a stabilization stage of 500×10^3 interactions. During this time, the actor was frozen while the agent moved through the environment, allowing the critic network to learn the value distribution of the expert policy. This provided a reliable starting point for the subsequent 3×10^6 PPO steps.

Quantitative Navigational Performance To assess the final robustness of these converged policies, they were evaluated over $N_{\text{ev}} = 500$ deterministic episodes without exploratory noise. The aggregated metrics in table 5.4 serve as the definitive verdict on their navigational competence.

Among the pure RL methods, the value-based DQN unexpectedly demonstrated the highest overall competence, achieving an Optimality Score of 0.820 and a near-perfect success rate. This superiority is attributed to the deterministic, discrete transitions of the grid-world environment. Coupled with a large experience replay buffer, DQN effectively decorrelated the semantic textures

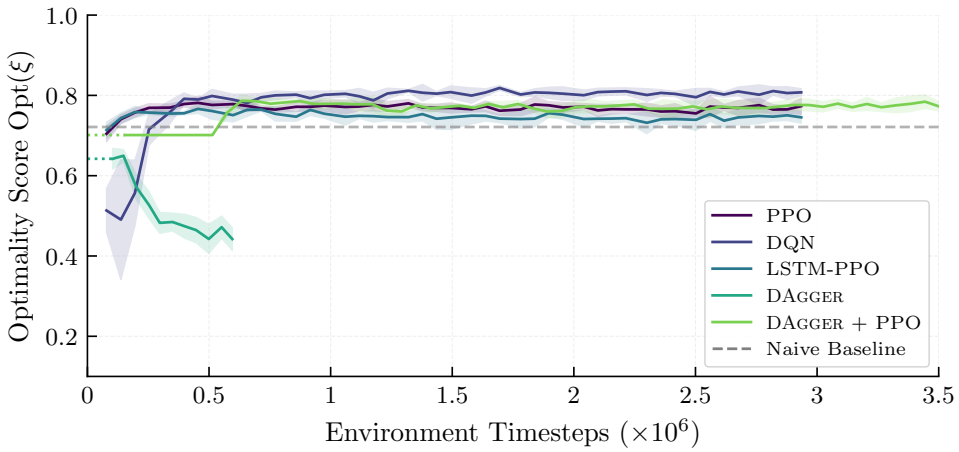


Figure 5.7 Learning dynamics across the evaluated optimization paradigms. The pure RL baselines were trained for 3×10^6 environment interactions, while the pure DAgger agent was trained for 0.6×10^6 interactions. The DAgger + PPO agent was initialized from an early imitation checkpoint (60×10^3 interactions), followed by a 500×10^3 interaction stabilization phase with a frozen actor and 3×10^6 RL fine-tuning steps. Solid lines denote the mean Optimality Score $\text{Opt}(\xi)$ across $N_{\text{sd}} = 5$ seeds, with shaded regions representing ± 1 standard deviation. The horizontal dashed line indicates the performance of the naive shortest-path baseline.

of the OSM maps. PPO's slightly lower optimality (0.779) stems from its on-policy entropy bonus, which naturally induces minor spatial inefficiencies to maintain exploration. Furthermore, the addition of a recurrent temporal bottleneck (LSTM-PPO) did not yield performance gains. This confirms that the explicitly constructed observation vector O_t, aggregating $N_f = 4$ frames, spatial memory $\mathbf{D}_t$, and the goal vector $\mathbf{g}_t$, sufficiently resolves the partial observability of the grid-world, making recurrent hidden states an unnecessary optimization overhead here.

The evaluation also exposes the brittleness of pure imitation in long-horizon navigation tasks. While the DAgger agent successfully replicated the Oracle's localized behaviour to achieve the highest visual Feature Capture (0.906), it suffered from compounding execution errors that led to pathological loitering. Lacking an intrinsic reward signal to recover from unmapped states, it recorded the lowest Success Rate (90.2%) and an Optimality Score (0.698) below the naive baseline.

The hybrid DAgger + PPO approach successfully "rescued" this imitation policy. By reintroducing the spatial RL reward, it resolved the compounding errors, restoring the Success Rate to 98.7% and lifting the Optimality Score to 0.784. However, when compared directly to the PPO agent trained from scratch, the final asymptotic performance of the hybrid model is statistically overlapping. This indicates that while the asymmetric pipeline functions cor-

Table 5.4 Quantitative comparison of learning paradigms in the OSM target domain. Values represent the mean ± one standard deviation computed across $N_{\text{sd}} = 5$ independent training seeds ($N_{\text{ev}} = 500$ evaluation episodes per seed). Deterministic baselines are reported without variance.

Learning Paradigm	Success [%] (↑)	Opt(ξ) (↑)	Ext(ξ) (↓)	Cap(ξ) (↑)
Naive Shortest Path	100.0	0.728	1.000	0.621
Oracle Baseline	100.0	1.000	1.228	1.000
Value-Based (DQN)	**99.7** ± 0.2	**0.820** ± 0.01	1.095 ± 0.01	0.821 ± 0.03
Policy-Based (PPO)	98.9 ± 1.0	0.779 ± 0.01	**1.057** ± 0.02	0.750 ± 0.03
Recurrent (LSTM-PPO)	98.8 ± 0.5	0.767 ± 0.01	1.092 ± 0.03	0.781 ± 0.03
Imitation (DAgger)	90.2 ± 3.2	0.698 ± 0.03	1.170 ± 0.05	**0.906** ± 0.08
Hybrid (DAgger + PPO)	98.7 ± 0.9	0.784 ± 0.00	1.066 ± 0.02	0.771 ± 0.02

rectly, the exploration landscape of this discrete environment is not severe enough to induce a true RL cold-start problem. Consequently, standard RL from scratch is fully capable of discovering the active perception trade-off, rendering the substantial computational overhead of the Oracle-guided pipeline unnecessary for this specific domain.

Qualitative Behavioural Analysis To investigate the quantitative disparities from table 5.4, we visualize the learned policies on a shared evaluation map (fig. 5.8). Here, the feature-aware Oracle dictates a significant detour to exploit a dense visual corridor. The DQN agent (fig. 5.8a) successfully internalizes this trade-off, mimicking the detour before decisively turning to the goal. In contrast, the PPO agent (fig. 5.8b) exhibits a risk-averse, efficiency-biased policy, cutting the corner early to prioritize dense spatial rewards. Finally, the DAgger agent (fig. 5.8c) illustrates the fundamental robustness limit of behavioural cloning. While it initially tracks the visual features, a small accumulation of execution errors eventually pushes the agent into a featureless region of the state space. Lacking an intrinsic value-based "pull" toward the goal, the agent's policy becomes stochastic in these unmapped regions, resulting in the observed loitering loop and a failure to recover the global trajectory.

Value Function and Policy Visualization The performance disparity between the value-based and policy-gradient paradigms is further justified by visualizing the underlying value landscapes. Figure 5.9 presents a comparison between the DQN action-value surface and the PPO state-value estimate for a shared environment topography. To ensure a valid camera footprint of 11×11 pixels can be sampled at every coordinate without exceeding the arena limits, the playable range is restricted to a 50×50 central sub-grid.

The exhaustive evaluation of the value function across the spatial domain is feasible in this phase because the observation O_t is strictly tied to the grid coordinate. This is a direct consequence of the discrete action space $\mathcal{A}$ and

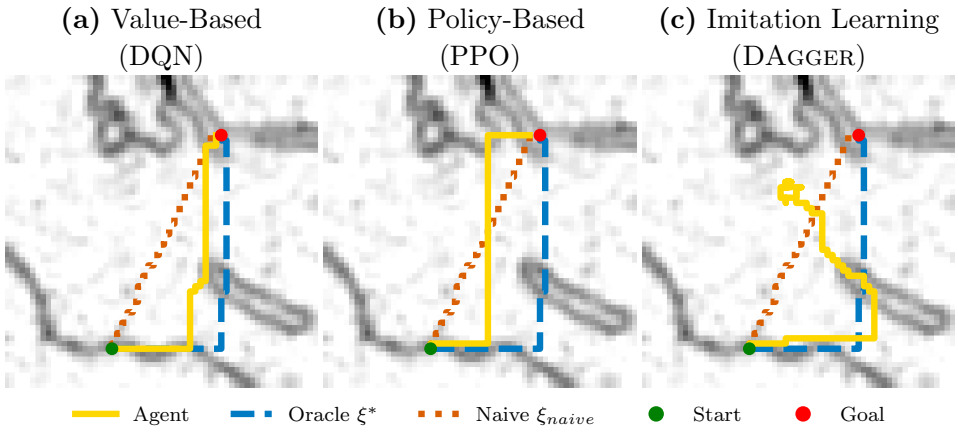


Figure 5.8 Qualitative trajectory comparison on a shared environment. The value-based DQN effectively approximates the Oracle's feature-aware detour **(a)**, whereas PPO exhibits a conservative, path-efficient bias **(b)**. The DAgger agent **(c)** illustrates the recovery failure inherent in pure imitation; once small errors push the agent into featureless terrain, the absence of an intrinsic navigational objective to prioritize the goal coordinates results in pathological loitering and mission failure.

the fixed orientation of the camera. The data was generated by forwarding all possible spatial observations O_t through the trained neural backbones while holding the navigational context constant. The target goal coordinates are fixed, the battery ratio is set to $e_t = 1.0$, and the temporal frame stack is populated with stationary observations. Consequently, each pixel in the resulting heatmaps corresponds to the expected return when an episode originates at that specific coordinate. In fig. 5.9b, the DQN surface represents the maximum action-value $\max_a Q(O_t, a)$, while fig. 5.9c shows the PPO state-value estimate $V(O_t)$.

Both models successfully identified the high-saliency regions as local maxima. These regions correspond to the semantic boundaries between different LandUse categories where the Sobel filter generates the strongest response. To facilitate qualitative comparison, both maps utilize a shared color scale representing the expected discounted return. The DQN surface in fig. 5.9b exhibits higher topological contrast near these semantic edges. The overlaid greedy policy demonstrates a decisive preference for following these boundaries. In contrast, the PPO value surface in fig. 5.9c appears more diffuse. This lack of gradient definition likely contributes to the slightly lower optimality scores reported in table 5.4. This visualization confirms that while both paradigms internalized the active perception objective, the explicit Q-value estimation of DQN provided a more stable gradient for navigating the discrete cost landscape.

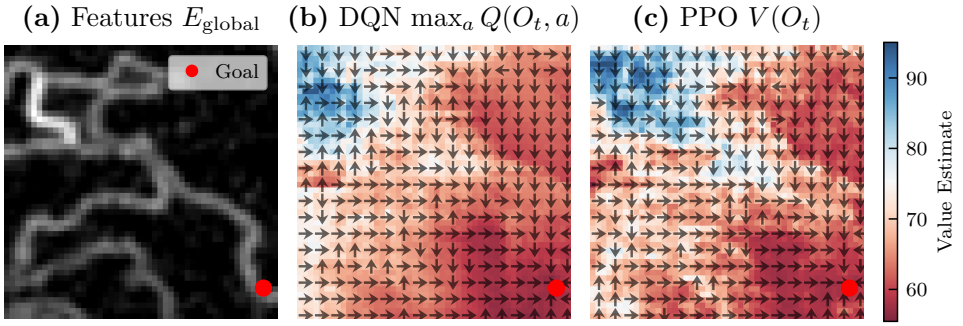


Figure 5.9 Comparison of learned value landscapes and greedy policies for the Phase I agents. **(a)** Input feature map E_{global} with the fixed target goal. **(b)** DQN value surface $\max_a Q(O_t, a)$ and corresponding greedy vector field. **(c)** PPO critic surface $V(O_t)$ and deterministic policy flow. Arrow density is subsampled every 3 cells for clarity. Each coordinate in the heatmaps represents the expected return if the agent were to start an episode at that location given the fixed navigational context.

5.6 Analysis and Preliminary Conclusions

The experimental evaluation in the discrete semantic environment successfully confirms the feasibility of learning a multi-objective policy for feature-aware navigation. By isolating the reward formulation from the complexities of continuous kinematics, this phase provides the empirical foundation for the active perception hypothesis.

5.6.1 Summary of Experimental Findings

The outcomes of this developmental cycle directly address the research questions formulated in section 5.5:

- **Learned Trade-off Competence (RQ 1):** The hyperparameter analysis confirms that the agent successfully internalizes the trade-off between path efficiency and feature acquisition. The high-performing policies consistently chose to extend the total trajectory length by approximately 10% to achieve a 20% increase in visual feature capture. This result is significant as it demonstrates that a localized reward signal, coupled with a goal-oriented observation vector, is sufficient to drive exploratory detours that approximate the behaviour of the privileged Dijkstra Oracle. Essentially, the agent learns to react to local visual stimuli in a way that optimizes the global path cost, despite lacking access to the full environmental cost-landscape during inference.
- **Failure of Synthetic Curricula (RQ 2):** The rejection of the multi-stage curriculum highlights the danger of rigid priors. Pre-training on

clean, synthetic B-splines induced a policy that could not generalize to the irregular semantic boundaries of the *OSM LandUse* dataset. Direct training on the target domain, while initially less stable, forced the agent to learn a more robust representation of real-world visual noise.

- **Paradigm Performance and Selection (RQ 3):** The comparison of learning paradigms yielded two complementary findings. First, the relative failure of pure imitation learning (DAgger) provided the most significant technical insight of this phase. The DAgger agent suffered from a fundamental *informational asymmetry*: the Dijkstra Oracle π_E planned trajectories using the global state S_t, whereas the agent was restricted to localized, partially observable crops O_t. Due to perceptual aliasing, identical local observations often corresponded to different optimal actions in the global frame. Without the intrinsic "pull" of a distance-to-goal reward $R_t^{(\text{dist})}$ to anchor the policy, the imitation learner frequently fell into pathological loitering cycles, emphasizing that active perception is more robustly formulated as a reinforcement learning problem than a supervised behavioural cloning task. Second, among the value-based and policy-gradient alternatives, the **Direct-Training DQN** emerged as the most capable architecture for this phase. It achieved the highest Optimality Score (0.820) by effectively utilizing the experience replay buffer to decorrelate semantic textures. Consequently, this architecture is designated as the representative configuration for Phase I, serving as the proof-of-concept for the multi-objective reward structure.

5.6.2 Limitations and the Transition to Phase II

While the experimental results in the discrete semantic environment successfully validated the core hypothesis of active perception, the Phase I framework relied on several domain abstractions. To transition this system to realistic UAV deployment, the underlying POMDP formulation must be expanded to address the following limitations:

Perceptual Aliasing and the Heuristic Flaw The use of spatial image gradients (∇I) to detect visual interest was effective in Phase I primarily due to the nature of the *OSM LandUse* dataset. Because this dataset represents terrain as clean semantic polygons, the Sobel filters perfectly correlated with the discrete class boundaries (e.g., the edge between a forest and a road). However, real-world high-resolution aerial imagery is significantly more complex. A well-documented limitation of heuristic gradient descriptors in the VPR literature [49] is that local edge detectors respond indiscriminately to high-frequency texture. When applied to authentic orthophotos, a Sobel filter would trigger on every individual tree branch in a dense forest or on the repetitive row structure of agricultural fields, which constitutes terrain that

is visually rich yet geographically non-unique. Because many geographically distinct locations produce identical gradient observations, maximizing for raw "edges" does not guarantee the agent is observing a localizable landmark. The VPR community's shift from hand-crafted local features toward global semantic descriptors (e.g., NetVLAD [40] and its successors) was precisely motivated by this failure mode.

Consequently, Phase II replaces the local gradient proxy with deep semantic descriptors that evaluate true visual distinctiveness rather than raw geometric contrast. Critically, the transition to continuous kinematics and the transition to a semantic perception signal are treated as a *coupled* design step, not as independent variables. This is a deliberate methodological choice: the overarching goal of Phase II is to instantiate a system that operates in a realistic aerial simulation regime, where both constraints hold simultaneously. Constructing a hybrid environment (continuous kinematics with Sobel-based rewards) would produce an internally inconsistent configuration that is neither representative of the validated Phase I abstraction nor of the target deployment scenario, where non-holonomic kinematics and semantic perception co-occur by definition.

The primary research question of Phase II is therefore not "which component drives performance?", but "does the complete active perception system remain effective under realistic constraints?". The contribution of the perceptual component within this coupled setup is nonetheless verified by a dedicated structural ablation in Phase II, which isolates the role of the saliency prior under identical kinematic and environmental conditions.

Kinematic Constraints and Continuous Action Spaces The Phase I model assumed the UAV moves as a discrete point on a grid using four cardinal actions. This implicitly assumes a model where the agent can execute instantaneous lateral translations in any direction without altering its global orientation. In reality, fixed-wing and forward-flying multi-rotor UAVs follow non-holonomic unicycle kinematics. Their state is defined continuously by position, heading, and velocity components $(x_t, y_t, \varphi_t, v_t, \omega_t)$, meaning the vehicle must physically rotate to change its trajectory over time. This continuous motion renders the discrete 4-way action space $\mathcal{A}$ and the value-based DQN algorithm insufficient for a real flight controller. The final system requires a transition to a continuous metric simulation where the agent controls normalized turning rates and forward velocities, necessitating the use of policy-gradient methods (PPO) optimized for continuous bounds.

These identified constraints dictate the design of the final system. The subsequent chapter details Phase II, where the conceptual foundations validated here are deployed in a continuous aerial simulator operating on authentic orthophotos.

Chapter 6

Phase II – Active Perception in Aerial Imagery

The experimental evaluation in Chapter 5 successfully demonstrated that a reinforcement learning agent can internalize the active perception trade-off, deviating from spatially optimal paths to acquire visual features. However, the Phase I formulation relied on two major abstractions that limit its direct application to physical UAV platforms: a discrete grid-world state space and the use of heuristic spatial gradients as a proxy for landmark saliency.

As identified in section 5.6.2, relying on raw spatial gradients poses a theoretical risk of severe perceptual aliasing when applied to authentic, high-resolution aerial imagery. Highly textured but repetitive environments, such as dense forests or agricultural fields, yield strong gradient responses despite lacking the uniquely identifiable landmarks required for robust VPR. Furthermore, realistic UAV deployment requires accounting for non-holonomic unicycle kinematics, where the vehicle must smoothly adjust its longitudinal velocity and yaw rate rather than executing instantaneous orthogonal translations.

To address these constraints, this phase formalizes the transition of the active perception POMDP into a continuous simulation domain. By replacing the heuristic edge-proxy with semantic saliency heatmaps derived from the SelaVPR++ Vision Transformer, the agent is trained to optimize for true visual distinctiveness. Consequently, the objective of this chapter is to evaluate whether policy-gradient algorithms can maintain navigational stability under the continuous dynamics of aerial flight while utilizing authentic orthophotos to mitigate perceptual aliasing.

It is important to note that Phase I and Phase II are not intended as controlled experiments relative to each other. They represent successive development milestones, each introducing a set of design changes motivated by the limitations identified in the preceding phase (see section 5.6.2). The cross-phase improvements therefore reflect an engineering progression rather than an

isolated causal claim. The controlled comparisons in this chapter are strictly the within-Phase II ablations between the three architectural variants defined in section 6.5 (ISP, SADP, and SASP), which share an identical simulator and action space.

6.1 Geospatial Data and Offline Perception

The transition to a high-fidelity environment begins with the acquisition and spatial processing of authentic aerial imagery. Unlike the simplified semantic maps used in the previous phase, the system now operates on high-resolution orthophotos capturing diverse topographical features. To enable the active perception objective, these raw rasters must be converted into a representation that characterizes the visual uniqueness of the terrain, allowing the agent to distinguish between repetitive textures and localizable landmarks.

Given the scale of the imagery and the millions of transitions required for reinforcement learning, a direct integration of the deep Vision Transformer backbone into the real-time simulation loop would introduce an immense computational bottleneck. This section details the multi-stage pipeline developed to resolve this: starting with the selection and metric re-projection of the raw aerial imagery, and concluding with the offline extraction methodology used to construct a static VPR database. This architecture ensures that the simulation can sample high-fidelity semantic rewards during training without the overhead of live neural inference.

6.1.1 Aerial Imagery and Spatial Preprocessing

To transition from the conceptual grid-world to a high-fidelity continuous environment, the discrete *LandUse* categories utilized in Phase I were replaced with authentic aerial orthophotos. These high-resolution datasets were sourced as Open Data from the Czech State Administration of Land Surveying and Cadastre (ČÚZK) [50]. To facilitate domain randomization and ensure the agent learns visual features that are robust to varying topographies, we acquired five distinct $25\,\mathrm{km}^2$ regions representing specific environmental challenges. The exact geographic center points of these regions are summarized in table 6.1.

Geodetic Projection and Metric Consistency A fundamental requirement for simulating continuous UAV kinematics is the ability to evaluate Euclidean distances $\|\mathbf{p} - \mathbf{p}'\|_2$ directly within the state space. Standard global geodetic coordinates (WGS84, `EPSG:4326`) represent positions as angular degrees of latitude and longitude, introducing non-linear spatial distortions when projected onto a 2D plane, particularly at the mid-latitudes of the Czech Republic. To maintain metric consistency, the sourced map tiles were re-projected into the national Křovák projection (S-JTSK, `EPSG:5514`) [51]. This

Table 6.1 Geographic distribution and topographical characteristics of the Phase II training dataset.

Region Name	Latitude	Longitude	Primary Topography
Nature Villages	49.947° N	12.982° E	Forest and small settlements
Nature Brdy	49.717° N	13.867° E	Unstructured dense forest
Nature Fields 1	50.358° N	14.040° E	High-aliasing agricultural land
Nature Fields 2	50.420° N	14.771° E	Mixed fields and tree lines
Town 1	50.110° N	14.593° E	Urban infrastructure and roads

transformation maps the Earth's curved surface onto a conformal conic projection where the Cartesian coordinates are expressed strictly in meters.

Spatial Aggregation and Resolution Normalization The cadastral orthophoto sheets are natively provided in fragmented $2.5 \times 2.0\,\text{km}$ blocks at a resolution of $0.125\,\text{m/px}$ [50]. To establish a continuous coordinate space for long-horizon navigation, these individual blocks were spatially aggregated and stitched into final global textures of $L_{\text{total}} \times L_{\text{total}} = 5 \times 5\,\text{km}$. This ensures that the simulated UAV's state S_t remains continuous and valid even when traversing the physical boundaries of the original cadastral grids.

Processing the imagery at its native resolution during training would introduce a severe computational bottleneck. Consequently, the aggregated environments were normalized to a fixed GSD of $\rho = 0.5\,\text{m/px}$. This discretization level serves as a heuristic trade-off, significantly optimizing sample efficiency while retaining the essential semantic landmarks required for the VPR feature extraction.

Sensor Observation Model The simulator utilizes the pinhole camera formulation established in eq. (5.1) to map the continuous environment to the observation space Ω. For a drone at altitude $h = 100\,\text{m}$ and FOV $\alpha_{\text{fov}} = 84°$, the physical ground footprint evaluates to $W_{ground} = 2 \cdot 100 \cdot \tan(42°) \approx 180\,\text{m}$. The resulting unscaled RGB image is denoted as the raw physical buffer $\mathbf{I}_t^{\text{raw}} \in \mathbb{R}^{3 \times H_c \times W_c}$. Given the normalized GSD $\rho = 0.5\,\text{m/px}$, the local camera footprint dimensions $H_c \times W_c$ (in pixels) are:

$$H_c = W_c = \left\lceil \frac{W_{ground}}{\rho} \right\rceil. \tag{6.1}$$

6.1.2 Offline VPR Database

Integrating a massive Vision Transformer directly into the training loop introduces a severe computational bottleneck. Because the training process requires millions of environment transitions to converge, evaluating the SelaVPR++

architecture on high-dimensional observation tensors at every simulation step would render the optimization intractable on the available cluster infrastructure. To circumvent this limitation, the visual perception pipeline was decoupled from the agent's control loop by pre-computing a static reference database for each global map.

The offline generation process utilized a sliding window approach across the aggregated global textures ($L_{\text{total}} \times L_{\text{total}}$). The window dimensions were strictly matched to the local observation tensor size $H_c \times W_c$ in pixels, which corresponds to the physical ground footprint $W_{ground} = 180\,\text{m}$ defined by the sensor model in section 6.1.1. To ensure adequate spatial overlap while bounding the memory footprint of the resulting arrays, the window was translated using a fixed physical stride of S_m meters. A stride of $S_m = 90\,\text{m}$ was selected to precisely achieve a 50% spatial overlap relative to the 180 m. This specific configuration is identified as the best-performing setup in current aerial VPR benchmarks [52]. The physical stride was dynamically converted to a pixel stride S_{px} using the GSD ρ.

For every image crop centered at position $[x_c, y_c]^\top$, the SelaVPR++ model generates two distinct outputs according to the adaptation mechanisms described in section 4.3.2:

1. A high-dimensional global descriptor $\mathbf{z} \in \mathbb{R}^{d_{\text{emb}}}$, encapsulating the semantic signature of the captured area.
2. A localized spatial saliency matrix $\mathbf{M}_{local} \in \mathbb{R}^{16\times16}$, identifying the position of robust visual landmarks within the specific camera footprint.

These outputs, alongside their corresponding spatial coordinates, were aggregated into dense multi-dimensional matrices and serialized. During the active RL simulation phase, this offline architecture allows the continuous engine to construct dynamic episode bounds and query the pre-computed tensors with $\mathcal{O}(1)$ time complexity, entirely bypassing the need for real-time ViT inference while preserving the authenticity of the visual descriptors.

6.2 The Continuous 2D Simulation Engine

To facilitate the transition to high-fidelity aerial navigation, we developed a custom simulation environment that bridges the gap between raw geospatial rasters and the RL agent's observation space. The engine is responsible for three core tasks: valid arena extraction via rejection sampling, the integration of second-order unicycle kinematics, and the generation of a continuous, uniqueness-weighted reward surface.

6.2.1 Hierarchical Coordinate Mapping

The simulation abstracts the physical and perceptual layers into a tiered hierarchy to ensure that the agent's dynamics are decoupled from the raw raster

resolution. This is achieved through a transformation chain across three coordinate systems:

1. **Metric Layer (m):** Serves as the primary specification for the simulation's physical scale. All high-level objectives, including the target arena dimensions and minimum traversal distances, are defined in meters to ensure consistent behaviour across different map sources.
2. **Pixel Layer (px):** This layer represents the native resolution of the orthophoto rasters with a GSD of $\rho = 0.5\,\mathrm{m/px}$. It is utilized exclusively for camera patch extraction and VPR database alignment.
3. **Grid Cell Layer (cells):** The primary space for RL dynamics, reward harvesting, and the Oracle's path planning. The pixel space is discretized by a factor $s = 10\,\mathrm{px/cell}$.

The mapping between these systems is governed by the relation $1\,\mathrm{cell} = s \cdot \rho$, resulting in a fixed scale of 5.0 metres per grid cell. All agent positions $\mathbf{p}_t$ are integrated as continuous floating-point vectors in the grid cell space. This enables the environment to function as a continuous-action POMDP while maintaining a computationally efficient integer-indexed reward map. Whenever the engine performs an image crop, the cell coordinates are multiplied by s to resolve the center of the window in pixel space, preserving sub-pixel geographic fidelity.

6.2.2 Arena Selection and Rejection Sampling

To ensure the policy generalizes across diverse topographies and to prevent overfitting to specific spatial layouts, each episode utilizes a dynamically sampled flight arena of $H_a \times W_a$ grid cells, corresponding to a physical area of $L_a \times L_a$ square meters. The selection process follows a rejection sampling paradigm. The simulator samples a random candidate crop from the macroscopic global textures ($L_{\text{total}} \times L_{\text{total}}$) described in section 6.1.1. A candidate is accepted only if it satisfies two conditions:

1. **Data Integrity:** The crop must not contain any null (black) pixels, ensuring the drone never spawns at the boundary of the sourced cadastral data.
2. **Texture Density:** To ensure the environment is "perceptually navigable", we enforce a variance threshold. A crop is rejected if the pixel-wise standard deviation $\sigma_{px} < 10$. This heuristic filters out uninformative water bodies or uniform agricultural fields where VPR-based localization is theoretically impossible.

Once a valid arena passes the variance filter, the episode is initialized. A target location $\mathbf{p}_{\text{goal}}$ and a starting position $\mathbf{p}_0$ are uniformly sampled within the continuous bounds of the $H_a \times W_a$ arena. To ensure a non-trivial navigation horizon, the sampling is constrained such that the initial Euclidean distance $\|\mathbf{p}_0 - \mathbf{p}_{\text{goal}}\|_2$ is strictly greater than 40% of the physical diagonal of the arena. The agent's initial heading φ_0 is randomized to prevent the policy from overfitting to a dominant global orientation.

6.2.3 Dynamic Saliency Potential Map Generation

As established in section 5.1.3, the agent's objective is to harvest rewards from a *Feature Availability Map* $\mathcal{F}$. While Phase I utilized a binary map $\mathcal{F}_0$ derived from heuristic Sobel edges, Phase II constructs a high-fidelity, continuous resource landscape based on learned SelaVPR++ descriptors.

To provide a continuous and differentiable reward signal, the engine dynamically constructs the initial feature map $\mathcal{F}_0(\mathbf{p})$ at the start of every episode. Let $\mathcal{D}_{\text{arena}} = \{1, \dots, N_{\text{db}}\}$ be the index set of VPR database entries that fall within the boundaries of the sampled $H_a \times W_a$ grid, where each entry $i \in \mathcal{D}_{\text{arena}}$ is associated with a global descriptor $\mathbf{z}_i$, grid coordinates $\mathbf{c}_i$, and a local 16×16 saliency matrix $\mathbf{M}_i$.

1. **Arena-Relative Uniqueness Weighting:** To resolve the perceptual aliasing issues discussed in section 2.3, features must be unique within their operational context. Each database entry $i \in \mathcal{D}_{\text{arena}}$ is assigned a uniqueness weight $w_i \in [0, 1]$. We define this weight using the maximum cosine similarity to any other descriptor within the current arena:

$$w_i = \left(1 - \max_{j \in \mathcal{D}_{\text{arena}}, j \neq i} \frac{\mathbf{z}_i \cdot \mathbf{z}_j}{\|\mathbf{z}_i\|_2 \|\mathbf{z}_j\|_2}\right)^2. \tag{6.2}$$

2. **Spatial Stitching and Aggregation:** To transform the sparse database entries into a continuous field, we first upscale each local saliency matrix $\mathbf{M}_i$ to the physical camera footprint using a bicubic interpolation function f_{up}. Let $\hat{\mathbf{M}}_i(\mathbf{p})$ be the upscaled patch centered at $\mathbf{c}_i$. The raw global saliency map $\hat{\mathcal{F}}(\mathbf{p})$ is computed as the weighted arithmetic mean of all overlapping patches at coordinate $\mathbf{p}$:

$$\hat{\mathcal{F}}(\mathbf{p}) = \frac{\sum_{i \in \mathcal{D}_{\text{arena}}} w_i \cdot \hat{\mathbf{M}}_i(\mathbf{p}) \cdot \mathbb{I}[\mathbf{p} \in \text{FOV}(\mathbf{c}_i)]}{\sum_{i \in \mathcal{D}_{\text{arena}}} \mathbb{I}[\mathbf{p} \in \text{FOV}(\mathbf{c}_i)] + \epsilon}, \tag{6.3}$$

where $\mathbb{I}[\cdot]$ is the indicator function for the camera field of view centered at $\mathbf{c}_i$, and $\epsilon = 1\text{e-}6$ prevents division by zero in regions with no database coverage.

3. **Gaussian Smoothing and Normalization:** To ensure the reward surface is differentiable and to remove artifacts from the discrete stride intersections, the map $\hat{\mathcal{F}}$ is convolved with a 2D Gaussian kernel G_σ with $\sigma = 10\,\text{m}$. The final Initial Feature Map is obtained via min-max normalization:

$$\mathcal{F}_0(\mathbf{p}) = \frac{(\hat{\mathcal{F}} * G_\sigma)(\mathbf{p}) - \mathcal{F}_{min}}{\mathcal{F}_{max} - \mathcal{F}_{min}}, \tag{6.4}$$

where $\mathcal{F}_{min}$ and $\mathcal{F}_{max}$ are the scalar extrema of the smoothed field across the arena.

This formulation ensures that the visual salience $\nu(\mathbf{p}) \doteq \mathcal{F}_t(\mathbf{p})$ provides a direct high-fidelity replacement for the Sobel-based salience used in Phase I in section 5.2. By normalizing the map to $[0, 1]$, we maintain consistency in the reward weights w_{view} across different environments regardless of the absolute descriptor magnitudes. As detailed in section 6.4, this static potential field is subsequently transformed into a continuous cost landscape to guide the deterministic baseline.

6.2.4 UAV Kinematics and Actuation

To bridge the gap between the idealized grid transitions of Phase I and the physical constraints of aerial flight, the vehicle's spatial transitions are modeled using a discrete-time unicycle formulation [53] augmented with second-order acceleration constraints [54]. This effectively approximates a multirotor platform executing forward flight with coordinated turning. The internal kinematic state of the vehicle at time t is represented by the vector $\boldsymbol{\zeta}_t = [x_t, y_t, \varphi_t, v_t, \omega_t]^\top$, where $\mathbf{p}_t = [x_t, y_t]^\top$ denotes the Cartesian position in the arena and φ_t is the heading angle. The variables v_t and ω_t represent the scalar longitudinal and angular velocities respectively. The resulting velocity vector $\mathbf{v}_t = [\dot{x}_t, \dot{y}_t]^\top$ is constrained to align with the current orientation of the vehicle. The composition of this vector is governed by the non-holonomic constraint $\dot{\mathbf{p}}_t = [v_t \cos\varphi_t, v_t \sin\varphi_t]^\top$. This geometric relationship is illustrated in fig. 6.1.

The simulation operates at a fixed frequency of 10 Hz ($\Delta t = 0.1\,\text{s}$). The vehicle accepts normalized control inputs $\mathbf{u}_t = [u_v, u_\omega]^\top \in [0, 1] \times [-1, 1]$, which are mapped to physical target velocities:

$$v_{\text{cmd}} = u_v \cdot V_{\max}, \tag{6.5}$$

$$\omega_{\text{cmd}} = u_\omega \cdot \Omega_{\max}. \tag{6.6}$$

Based on standard performance profiles for medium-scale UAVs, the limits are set to $V_{\max} = 12.0\,\text{m/s}$ and $\Omega_{\max} = \pi/2\,\text{rad/s}$. To prevent unrealistic instantaneous velocity changes, the engine enforces linear and angular acceleration

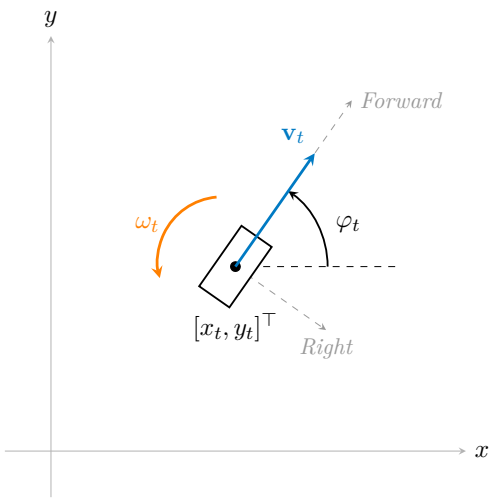


Figure 6.1 Kinematic representation of the second-order unicycle model in the continuous 2D plane. The vehicle pose is defined by the global Cartesian coordinates $[x_t, y_t]^\top$ and the heading angle φ_t relative to the x-axis. Navigational telemetry is resolved in the egocentric *Forward-Right* frame. The blue arrow represents the velocity vector $\mathbf{v}_t$ with longitudinal magnitude v_t and the orange arrow denotes the angular velocity ω_t. Adapted from [54].

bounds ($A_{\max} = 5.0\,\mathrm{m/s^2}$ and $\alpha_{\max} = \pi\,\mathrm{rad/s^2}$). The velocities are updated at each step via a clamped difference:

$$v_{t+1} = v_t + \mathrm{clip}\left(v_{\mathrm{cmd}} - v_t,\, -A_{\max}\Delta t,\, A_{\max}\Delta t\right), \tag{6.7}$$

$$\omega_{t+1} = \omega_t + \mathrm{clip}\left(\omega_{\mathrm{cmd}} - \omega_t,\, -\alpha_{\max}\Delta t,\, \alpha_{\max}\Delta t\right). \tag{6.8}$$

Following the velocity update, the geometric pose of the vehicle is resolved using the discrete-time state transition:

$$\varphi_{t+1} = (\varphi_t + \omega_{t+1}\Delta t) \bmod 2\pi, \tag{6.9}$$

$$x_{t+1} = x_t + v_{t+1}\cos(\varphi_{t+1})\Delta t, \tag{6.10}$$

$$y_{t+1} = y_t + v_{t+1}\sin(\varphi_{t+1})\Delta t. \tag{6.11}$$

6.2.5 Virtual Electronic Image Stabilization (EIS)

To maintain compatibility between continuous aerial maneuvers and the semantic VPR descriptors, the simulator implements a virtual Electronic Image Stabilization (EIS) module based on an *oversized sensor model*.

Standard VPR architectures, including the SelaVPR++ backbone, are highly sensitive to the global orientation of the input image and lack inherent rotation invariance [2]. If the reference database $\mathcal{V}$ is constructed from axis-aligned (North-Up) orthophotos while the UAV maneuvers at arbitrary yaw angles φ_t, rotational aliasing will severely degrade the descriptor matching

distance $d(\cdot, \cdot)$. Conversely, presenting a static North-Up view to the RL agent complicates the learning of relative navigational tasks, as the mapping between primitive actions and visual motion changes based on the drone's heading.

To resolve this conflict without introducing rescaling or interpolation artifacts, the EIS module cleanly decouples the *navigational reference frame* from the *perceptual reference frame* using a shared geographic footprint. Let the target observation window be $H_c \times W_c$. At each physical time step t, the simulation executes the following extraction pipeline:

1. **Oversized Buffer Extraction:** The engine queries the global raster to extract an axis-aligned (north-up) square patch centered at the drone's continuous coordinates $\mathbf{p}_t = [x_t, y_t]^\top$. To guarantee that subsequent rotations do not produce undefined corner pixels, the dimension of this source buffer is strictly bounded by the diagonal of the target window, requiring:

$$N_{\text{buf}} \geq \lceil \sqrt{H_c^2 + W_c^2} \rceil. \tag{6.12}$$

2. **Navigational View (Heading-Up):** To generate the egocentric input for the policy, the oversized buffer is rotated around its center by $-\varphi_t$. A strict center-crop of $H_c \times W_c$ is then extracted, resulting in the raw physical buffer $\mathbf{I}_t^{\text{raw}} \in \mathbb{R}^{3 \times H_c \times W_c}$. This transformation ensures the vehicle's forward velocity vector always aligns with the vertical axis of the image. This rotation and cropping protocol is applied concurrently to the raw RGB imagery, the global saliency potential map, and the spatial visitation grid. This unified transformation ensures that all visual modalities are spatially synchronized in the agent's egocentric reference frame.

3. **Perceptual View (North-Up):** To facilitate true spatial evaluation and SelaVPR++ query generation, a direct center-crop of $H_c \times W_c$ is extracted from the *unrotated* oversized buffer. Because this crop bypasses the yaw-dependent rotation, it remains perfectly aligned with the axis-parallel reference database $\mathcal{V}$, preventing the descriptor degradation associated with rotational aliasing.

Because both crops are derived from the exact same source buffer sharing the same geographic center and pixel scale, the system guarantees that the RL agent and the VPR evaluation module are analyzing the identical geographic footprint, merely projected into two distinct orientation spaces. This decoupling enables the agent to utilize a simplified egocentric observation space while ensuring that all VPR database queries remain perfectly aligned with the offline SelaVPR++ embeddings.

6.3 Continuous POMDP Formulation

Expanding the framework of Phase I (section 5.1), the navigation task is reformulated for the continuous state space and kinematics of the aerial simulator. We define this environment as a continuous POMDP.

6.3.1 State, Action, and Observation Spaces

The global state space $\mathcal{S}$ is fundamentally continuous. The true state $S_t \in \mathcal{S}$ is defined by the vehicle's kinematic vector $\boldsymbol{\zeta}_t = [x_t, y_t, \varphi_t, v_t, \omega_t]^\top$, operating within the bounds of the extracted arena, alongside the dynamic state of the global spatial memory and feature maps.

Consistent with the methodology in section 5.1, the environment is modeled as a partially observable process. The agent receives a multi-modal observation $O_t = (\mathbf{I}_t, \mathbf{M}_t, \mathbf{D}_t, \mathbf{g}_t) \in \Omega$, where each spatial component is processed through the EIS module to maintain egocentric alignment. Unlike the discrete Phase I, directly processing the high-fidelity physical footprint ($H_c \times W_c = 360 \times 360$) is computationally expensive for the RL training loop. Therefore, the physical buffer $\mathbf{I}_t^{\text{raw}}$ is downsampled via bilinear interpolation to the fixed input resolution of $H_{in} \times W_{in} = 84 \times 84$ pixels. Consequently, the observation space for the visual modalities is defined as:

- $\mathbf{I}_t \in \mathbb{R}^{3N_f \times H_{\text{in}} \times W_{\text{in}}}$: This visual tensor aggregating the $N_f = 4$ most recent RGB camera frames. This temporal concatenation allows the convolutional backbone to internalize visual flow and motion parallax, which are critical for state estimation under continuous kinematics.

- $\mathbf{M}_t \in \mathbb{R}^{N_f \times H_{\text{in}} \times W_{\text{in}}}$: The saliency prior tensor. This consists of a temporal sequence of local crops sampled from the pre-computed global saliency map $\mathcal{F}_0$. By providing the agent with the local distribution of unique landmarks, this channel serves as a spatial heuristic that identifies regions of high VPR confidence, mitigating the ambiguity of raw RGB textures.

- $\mathbf{D}_t \in \mathbb{R}^{N_f \times H_{\text{in}} \times W_{\text{in}}}$: The visitation density tensor. Similar to the implementation in chapter 5, this channel represents the history of the agent's spatial trajectory. It allows the policy to internalize the depletion of visual rewards and prevents redundant exploration of already-visited terrain.

- $\mathbf{g}_t \in \mathbb{R}^{6N_f}$: The low-dimensional telemetry vector, constructed as a flattened temporal stack of N_f states following the logic in section 5.1. The instantaneous state $\tilde{\mathbf{g}}_t = [d_t, \Delta x_{\text{ego}}, \Delta y_{\text{ego}}, \hat{v}_t, \hat{\omega}_t, e_t]^\top$ captures the normalized goal distance d_t, the remaining battery e_t, and the normalized longitudinal and angular velocities $(\hat{v}_t, \hat{\omega}_t)$. To maintain an egocentric representa-

tion, the global relative direction vector to the goal $(\Delta x, \Delta y)$ is projected into the vehicle's *Forward-Right* frame using the current heading φ_t:

$$\Delta x_{\text{ego}} = \Delta x \cos(\varphi_t) + \Delta y \sin(\varphi_t), \tag{6.13}$$
$$\Delta y_{\text{ego}} = \Delta y \cos(\varphi_t) - \Delta x \sin(\varphi_t). \tag{6.14}$$

The final input $\mathbf{g}_t$ is the concatenation of the current and $N_f - 1$ previous instantaneous vectors $\tilde{\mathbf{g}}$.

Action Space Unlike the discrete cardinal movements defined in section 5.1, the action space in Phase II is defined as a continuous subset of the real plane $\mathcal{A} \subset \mathbb{R}^2$. Specifically, the space is bounded such that $\mathcal{A} \doteq [0, 1] \times [-1, 1]$. At each time step t, the agent selects an action $A_t \in \mathcal{A}$, which is directly mapped to the normalized control vector $\mathbf{u}_t = [u_v, u_\omega]^\top$. These components represent the target linear and angular velocity commands respectively. The simulation engine subsequently maps these normalized values to the physical target velocities v_{cmd} and ω_{cmd} using the scaling laws and kinematic constraints established in section 6.2.4.

6.3.2 Continuous Reward Design

The reward function R_t maintains the multi-objective piecewise structure introduced in section 5.1.3, balancing goal-seeking progress against feature acquisition:

$$R_t = \begin{cases} R_{\text{success}}, & \text{if } S_t \in \mathcal{S}_{\text{goal}}, \\ R_{\text{fail}}, & \text{if } t \geq T_{\max} \vee S_t \notin \mathcal{S}_{\text{safe}}, \\ R_t^{(\text{step})}, & \text{otherwise}, \end{cases} \tag{6.15}$$

where the composite navigational reward $R_t^{(\text{step})}$ is defined using static weighting coefficients:

$$R_t^{(\text{step})} = w_{\text{dist}} \cdot R_t^{(\text{dist})} + w_{\text{view}} \cdot R_t^{(\text{view})} + R_{\text{time}} + R_t^{(\text{out})}. \tag{6.16}$$

The terminal success region $\mathcal{S}_{\text{goal}}$ is defined identically to the Phase I formulation (section 5.1.3). The dense distance reward $R_t^{(\text{dist})}$ and time penalty R_{time} also retain their foundational logic. However, the boundary constraints, feature harvesting mechanisms, and the dynamic application of the reward weights are significantly reformulated to support the continuous domain.

Two-Tier Boundary Constraints While Phase I utilized a simple discrete penalty $R_t^{(\text{border})}$ for arena collisions, the continuous engine implements a structured two-tier boundary system to provide a smoother learning gradient.

First, to prevent the agent's observation window from completely escaping the defined global map, we restrict the agent to a safe operational subset of the arena, denoted as $\mathcal{S}_{\text{safe}}$. This hard boundary applies an inward margin δ_{safe} from the absolute map limits, defined as one-quarter of the physical camera footprint width ($W_{ground}/4$). If the agent violates this margin ($S_t \notin \mathcal{S}_{\text{safe}}$), a terminal "crash" is triggered, yielding R_{fail}.

Second, because the camera radius ($W_{ground}/2$) is larger than the safe margin (δ_{safe}), the agent will begin to observe unmapped (null) regions before it physically crosses the hard boundary. The virtual EIS module pads these out-of-bound array accesses with black pixels. We exploit this to formulate $R_t^{(\text{out})}$ as a continuous soft penalty proportional to the fraction of null pixels in the raw, unscaled physical camera view $\mathbf{I}_t^{\text{raw}}$. Let $\mathcal{U} = \{1, \dots, W_c\} \times \{1, \dots, H_c\}$ denote the local index set of discrete pixel coordinates within this internal 360×360 simulation tensor. The penalty is formulated as:

$$R_t^{(\text{out})} = c_{\text{black}} \cdot \frac{1}{H_c W_c} \sum_{\mathbf{u} \in \mathcal{U}} \mathbb{I}[\mathbf{I}_t^{\text{raw}}(\mathbf{u}) = \mathbf{0}], \tag{6.17}$$

where c_{black} is a negative scaling constant and $\mathbf{0}$ represents a zero vector of dimension $3N_f$ corresponding to the stacked temporal RGB frames. This formulation naturally shapes the policy, providing a dense penalty signal that repels the agent from the map edges before it incurs a catastrophic terminal failure.

Continuous Feature Harvesting To formalize the active perception objective in the continuous domain, we generalize the consumable resource mechanism introduced in Phase I. However, the most significant departure is the transition from heuristic local Sobel-edges to global SelaVPR++ saliency depletion.

The global *Feature Availability Map* $\mathcal{F}_0(\mathbf{p})$, defined in section 6.2.3, represents the total unique landmark volume available in the arena. Let $\chi_t^{\text{FOV}}(\mathbf{p}) \in \{0, 1\}$ be the binary characteristic function representing the camera's spatial footprint at time t, projected onto the simulation's grid layer and rotated by the vehicle's yaw φ_t.

We model the visual reward as the volume of previously unobserved semantic features that the agent successfully harvests within its footprint. To prevent reward hacking (e.g., hovering over a single salient landmark), the availability map is dynamically depleted at each simulation step:

$$\mathcal{F}_{t+1}(\mathbf{p}) = \max(0,\, \mathcal{F}_t(\mathbf{p}) - \lambda_{\text{dep}} \cdot \chi_t^{\text{FOV}}(\mathbf{p})), \tag{6.18}$$

where λ_{dep} is the depletion rate, set empirically to $\lambda_{\text{dep}} = 0.05$. The final view-based reward $R_t^{(\text{view})}$ is the normalized sum of the harvested potential:

$$R_t^{(\text{view})} = \frac{1}{N_{\text{ref}}} \sum_{\mathbf{p}} \left(\mathcal{F}_t(\mathbf{p}) - \mathcal{F}_{t+1}(\mathbf{p})\right), \tag{6.19}$$

where N_{ref} is a normalization constant representing the theoretical maximum harvest achievable in a single step. This mechanism effectively forces the policy to continuously explore novel salient regions, as returning to a previously visited coordinate yields zero visual reward.

Dynamic Reward Scheduling A significant challenge introduced by the continuous non-holonomic environment is the substantial kinematic cost of trajectory deviation. Unlike the discrete grid transitions used in Phase I, the vehicle cannot execute instantaneous lateral translations; every maneuver requires a physical arc governed by the current velocity and maximum yaw rate. Consequently, aggressive static incentives for feature acquisition (w_{view}) may induce terminal loitering. This represents a degenerate policy where the agent indefinitely orbits high-saliency landmarks near the destination because the local reward density outweighs the marginal gain of entering the success radius ϵ_{goal}.

To investigate a potential resolution for this without suppressing the agent's exploratory capabilities during the initial flight phases, we propose a non-stationary reward formulation. In this setup, the weighting coefficients are dynamically interpolated based on the agent's relative mission progress. Let d_{init} represent the Euclidean distance to the goal at the start of the episode and d_t the distance at time t. We define a progress coefficient $\Psi_t \in [0, 1]$ as follows:

$$\Psi_t = 1 - \text{clip}\left(\frac{d_t}{d_{\text{init}}},\ 0,\ 1\right). \tag{6.20}$$

The effective weights applied to the spatial and perceptual components at step t are then calculated using a linear interpolation:

$$w_{dist,t} = w_{dist,min} + \Psi_t(w_{dist,max} - w_{dist,min}), \tag{6.21}$$

$$w_{view,t} = w_{view,max} - \Psi_t(w_{view,max} - w_{view,min}). \tag{6.22}$$

These constants define a behavioural envelope that shifts over the mission duration. The parameters $w_{dist,min}$ and $w_{dist,max}$ establish the bounds of the goal-seeking drive, while $w_{view,max}$ and $w_{view,min}$ dictate the magnitude of the perceptual incentive. This schedule is designed to facilitate a smooth transition; during the initial transit phase ($\Psi_t \approx 0$), the agent operates under an exploratory prior to maximize visual feature capture. As the agent approaches the destination ($\Psi_t \to 1$), the visual incentive decays to ensure the policy prioritizes arrival and terminates the episode within the horizon $T_{\max}$.

While this mechanism provides a theoretical safeguard against loitering, its necessity is not a given. The inherent regularization provided by the unicycle kinematics might be sufficient to prevent such cycles even with static weights. This formulation is therefore treated as a candidate architecture, the utility of which is subjected to a rigorous structural ablation in section 6.6.5.

6.4 Algorithmic Baseline (Continuous Cost-Graph Oracle)

To quantitatively evaluate the agents, it is necessary to establish a baseline for feature-aware continuous navigation. While the Phase I Oracle (section 5.2) operated strictly as an optimal discrete 4-way graph solver, the Phase II baseline must satisfy two advanced requirements: it must optimize a cost field derived from deep semantic descriptors, and its output must be physically executable by the second-order unicycle kinematics. Because deriving a true analytical optimum for this non-holonomic system over a continuous cost field is intractable, we formulate a privileged heuristic baseline by combining Dijkstra's algorithm for global planning with a classical Pure Pursuit tracking algorithm [55].

6.4.1 VPR-Aware Cost Landscape

The Oracle operates on the continuous spatial cost minimization principle introduced in section 5.2. However, the semantics of the visual salience $\nu(\mathbf{p})$ are fundamentally upgraded for the aerial domain. Instead of relying on a geometric Sobel-edge proxy, the salience is directly mapped to the Initial Feature Map $\mathcal{F}_0(\mathbf{p})$ derived from SelaVPR++ descriptors in section 6.2.3.

The complete pipeline, from raw orthophoto to the planning-ready cost landscape, is visualized in fig. 6.2. Because $\mathcal{F}_0(\mathbf{p})$ encapsulates the uniqueness weighting and spatial smoothing, the Oracle derives the traversal cost map $c(\mathbf{p})$ as follows:

$$c(\mathbf{p}) = 1.0 + \exp\left(\eta \cdot (1 - \mathcal{F}_0(\mathbf{p}))\right). \tag{6.23}$$

This soft-inverse mapping ensures that visually unique, landmark-dense regions are computationally cheap to traverse. Conversely, featureless or repetitively textured terrain, such as the agricultural fields visible in fig. 6.2a, incurs an exponentially high traversal cost governed by the sensitivity parameter η. To maintain a consistent navigational objective across both developmental phases, the value $\eta = 10.0$ is retained from Phase I. Despite the transition to high-dimensional SelaVPR++ saliency, this parameterization continues to serve as an effective penalty for perceptually aliased terrain, effectively warping the Dijkstra search space into the "localizability corridors" visualized in fig. 6.2c. By prioritizing VPR confidence over naive geometric distance, the Oracle provides a deterministic upper bound for the perceptual robustness of the system.

6.4.2 Trajectory Generation via Pure Pursuit

To solve the global optimization problem, the continuous cost field is discretized into an 8-way connected grid graph. Dijkstra's algorithm then identifies the cost-minimized waypoint sequence $\mathcal{W}^* = (\mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_N)$.

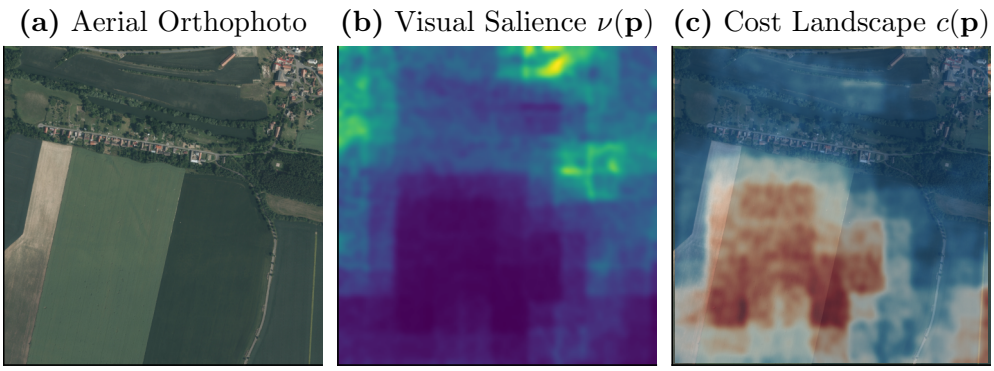


Figure 6.2 Generation of the feature-aware cost landscape for the Oracle baseline. **(a)** Raw RGB orthophoto. **(b)** Visual salience $\nu(\mathbf{p})$ (initially $\mathcal{F}_0$). **(c)** The resulting continuous cost landscape $c(\mathbf{p})$ where high-cost regions (red) correspond to perceptually aliased terrain, forcing the Dijkstra planner toward salient infrastructure and tree lines.

Because the unicycle UAV cannot instantaneously execute the sharp angular transitions inherent to grid-based waypoints, the sequence $\mathcal{W}^*$ cannot be directly used as an expert policy. To bridge the gap between discrete path planning and continuous actuation, the Oracle implements a *Pure Pursuit* path-tracking controller [55].

As illustrated in fig. 6.3, the algorithm calculates the curvature κ of an arc joining the vehicle's current position to a target waypoint on the path at a constant lookahead distance l:

$$\kappa = \frac{2x}{l^2}, \tag{6.24}$$

where x is the lateral offset in the vehicle's coordinate system. We fix the lookahead distance to $l = 75\,\mathrm{m}$ to act as a damping factor, ensuring stable VPR camera crops. To adapt this geometry for the Phase II unicycle dynamics, the Oracle calculates the heading error e_φ toward the lookahead point. The normalized angular velocity command $u_\omega^* \in [-1, 1]$ is then determined via a proportional control law:

$$u_\omega^* = \mathrm{clip}\left(\frac{k_\omega \cdot e_\varphi}{\Omega_{\max}},\ -1,\ 1\right), \tag{6.25}$$

where k_ω is the proportional gain and $\Omega_{\max}$ is the vehicle's maximum yaw rate. In our implementation, a simple P-controller proved sufficient to maintain stability, as the lookahead distance l provides inherent damping to the system.

Furthermore, as the classical model assumes constant velocity, we introduce a heuristic scheduler to provide physically realistic expert labels. The

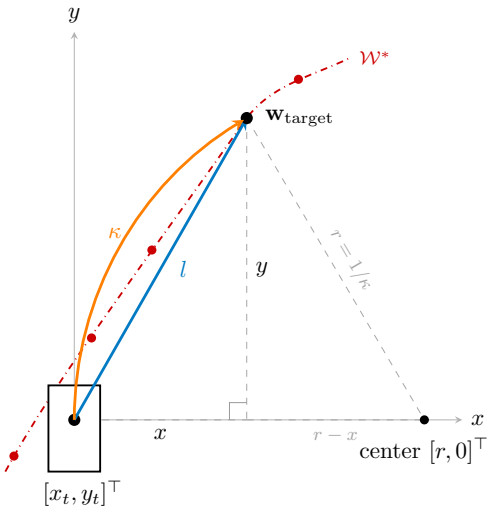


Figure 6.3 Geometric derivation of the Pure Pursuit tracking law used by the Oracle. The vehicle at $[x_t, y_t]^\top$ pursues a target waypoint $\mathbf{w}_{\text{target}}$ selected from the global Dijkstra path $\mathcal{W}^*$ at a constant lookahead distance l. The steering command is derived from the curvature κ, which defines an arc connecting the vehicle's origin to the target point based on the lateral offset x resolved in the body-fixed coordinate system. Adapted from [55].

longitudinal command $u_v^* \in [0, 1]$ is dynamically throttled based on the turn severity and goal proximity:

$$u_v^* = \text{clip}\left(\max\left(0.2,\ 1 - \frac{|e_\varphi|}{\pi}\right) \cdot \text{brake}(d_{\text{goal}}),\ 0,\ 1\right). \tag{6.26}$$

The first term mimics a coordinated turn by reducing speed during high-curvature maneuvers, while $\text{brake}(d_{\text{goal}})$ applies a linear deceleration profile as the agent enters the terminal success radius. This integrated controller ensures the Oracle generates a safe, feature-rich trajectory ξ^* that respects the simulator's acceleration limits and second-order dynamics.

6.5 Evaluated Architectures and Paradigms

To ensure a rigorous comparative analysis across the two developmental phases, the system retains the multi-modal fusion topology established in chapter 5. The high-level logic of merging spatial observations with navigational telemetry remains unchanged, though specific components of the network were refined to handle the increased input dimensionality and the requirements of continuous unicycle flight. The updated architecture is visualized in fig. 6.4.

6.5.1 Architectural Scaling and Continuous Heads

The transition from abstract 11×11 semantic crops to high-resolution aerial observations, coupled with the introduction of an explicit saliency prior, necessitated a substantial redesign of the visual encoder. As detailed in section 6.6.1, the physical observation window is 360×360 pixels. However, the input resolution processed by the neural network in this phase is $H_{\text{in}} \times W_{\text{in}} = 84 \times 84$ pixels, representing a roughly 58-fold increase in input dimensionality compared to Phase I.

Multi-modal Backbone The input to the visual branch is expanded to accept an augmented spatial tensor $\in \mathbb{R}^{5N_f \times H_{\text{in}} \times W_{\text{in}}}$, which concatenates the RGB stack $\mathbf{I}_t$ ($3N_f$ channels), the saliency prior $\mathbf{M}_t$ (N_f channels), and the visitation density $\mathbf{D}_t$ (N_f channels). Given the temporal window $N_f = 4$, this results in a 20-channel input tensor. To efficiently process this volume and internalize complex spatial relationships, the CNN utilizes deeper layers with expanded receptive fields. It employs 8×8, 4×4 and 3×3 kernels with aggressive strides to downsample the signal while preserving macroscopic semantic structures. Furthermore, the layer normalization present in the Phase I architecture was removed. This modification prevents the artificial smoothing of the sharp, high-frequency uniqueness boundaries embedded within the SelaVPR++ saliency prior $\mathbf{M}_t$. The branch outputs the latent spatial embedding $\mathbf{e}_{\text{view}} \in \mathbb{R}^{128}$.

The processing of the telemetry vector $\mathbf{g}_t$ remains structurally consistent with the MLP branch in chapter 5, though it now operates on the egocentric unicycle state defined in section 6.3. The resulting joint embedding $\mathbf{e}_t \in \mathbb{R}^{160}$ is passed through the same *Temporal Integration* configurations (Identity or LSTM) to produce the final latent vector $\mathbf{y}_t$ used by the decision heads.

Continuous Actor-Critic Head Because value-based algorithms like DQN are structurally limited to discrete action spaces, the optimization focus shifts entirely to actor-critic methodologies. Building upon the Gaussian policy formalization in section 2.4.2, the Phase II decision heads are implemented as parallel MLPs utilizing a partitioned parameter vector $\boldsymbol{\theta} = [\boldsymbol{\theta}_\mu, \boldsymbol{\theta}_\sigma]$.

The multi-modal latent vector $\mathbf{y}_t$ is processed by a two-layer MLP with hyperbolic tangent (*tanh*) activations for both the actor and the critic branches. In this architecture, the actor branch parameterized by $\boldsymbol{\theta}_\mu$ maps the current observation to the mean vector $\boldsymbol{\mu}_t$ of the continuous action distribution. Simultaneously, the exploration noise is decoupled from the input observation and governed strictly by the state-independent log-standard deviation parameters $\boldsymbol{\theta}_\sigma \doteq \log \boldsymbol{\sigma}$.

Crucially, the network explicitly enforces forward-only unicycle flight by restricting the longitudinal velocity command $u_v \in [0.0, 1.0]$, while allowing bidirectional turning via the angular velocity command $u_\omega \in [-1.0, 1.0]$. Dur-

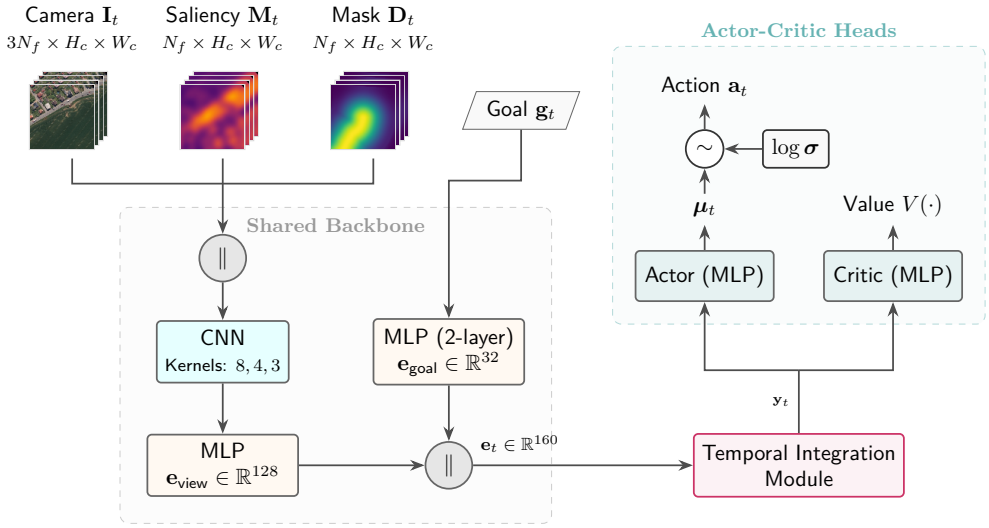


Figure 6.4 Architectural adaptations for the Phase II model. The multi-modal backbone was expanded to include the saliency prior $\mathbf{M}_t$. The decision heads were transitioned from a discrete value-based output to a continuous Actor-Critic configuration. The Temporal Integration Module remains structurally identical to the module defined in fig. 5.2, and its internal components are not repeated here for the sake of brevity.

ing training, sampled actions $A_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \text{diag}(\boldsymbol{\sigma}^2))$ are clipped to these bounds and passed to the simulation engine, where they are scaled by the dynamics constants ($V_{\max}$ and $\Omega_{\max}$) as described in section 6.2.4. Parallel to the policy generation, the critic branch independently reduces the latent embedding $\mathbf{y}_t$ to the scalar state-value estimate $V(O_t)$, serving as the baseline for generalized advantage estimation.

6.5.2 Optimization Paradigms

To identify the most robust optimization strategy for the continuous POMDP, we evaluate the following paradigms using the shared continuous architecture:

Policy-Gradient Methods The primary RL baseline utilizes PPO, which is optimized against the multi-objective reward signal R_t. To evaluate whether the unicycle kinematics and complex orthophotos require explicit temporal memory to resolve perceptual aliasing, we also evaluate a recurrent variant (LSTM-PPO). Following Phase I, this model incorporates a 128-unit LSTM bottleneck before the actor-critic heads. This recurrent layer maintains a hidden state $\mathbf{h}_t$ that serves as a proxy for the belief state by integrating observations over time.

Interactive Imitation Learning Building upon the asymmetric imitation framework introduced in section 5.4.3, we evaluate DAGGER in the continuous domain to test whether expert supervision can bypass the RL exploration ceiling. The expert policy π_E is now provided by the Continuous Pure Pursuit Oracle (section 6.4). Because the action space is continuous ($\mathcal{A} \subset \mathbb{R}^2$), the categorical classification objective used in Phase I is replaced with a MSE loss. This objective minimizes the mean squared error between the target actions $A_t = \pi_E(S_t)$ and the mean of the agent's distribution parameterized by $\boldsymbol{\theta}_\mu$.

To mitigate the compounding execution errors inherent in continuous unicycle kinematics, the training follows the rigorous DAGGER schedule defined in section 3.2. The agent is initially seeded on a dataset $\mathcal{D}$ of 4×10^5 expert transitions to stabilize the visual encoder. This is followed by 10 interactive rounds aggregating an additional 2×10^4 transitions where the agent rolls out its own policy.

Crucially, this continuous setting intensifies the theoretical loss of identifiability discussed in section 3.2. Because the agent is conditioned on the approximated belief state b_t while the expert labels depend on the true state S_t, the agent is forced to minimize divergence from the *implicit policy* $\hat{\pi}$ in regions of the state-belief occupancy $d^{\pi_{\boldsymbol{\theta}}}$ where it drifts. The comparative robustness of this paradigm, and its susceptibility to perceptual aliasing under continuous dynamics, is analyzed in section 6.6.6.

6.6 Experimental Results and VPR Evaluation

This section details the empirical evaluation of the Phase II navigation system, focusing on its performance within a continuous aerial environment. Building upon the conceptual foundations of the previous phase, we evaluate the agent's ability to balance navigational efficiency with visual localization precision while subjected to non-holonomic unicycle kinematics. To ensure a consistent comparative analysis, the subsequent experiments are designed to address four core research question that mirror the developmental stages of the system:

1. **Continuous Objective Balancing (RQ 4):** How does the transition to second-order unicycle kinematics and continuous state spaces affect the trade-off between path efficiency and landmark acquisition compared to the discrete grid-world?

2. **Structural Ablation of Perceptual Priors (RQ 5):** To what extent does providing the agent with an explicit saliency prior resolve perceptual aliasing, and is a non-stationary reward schedule necessary to prevent terminal loitering?

3. **Paradigm Robustness (RQ 6):** Among the evaluated policy-gradient and imitation learning architectures, which paradigm demonstrates the

highest stability when subjected to the accumulating execution errors of continuous flight?

4. **Zero-Shot Generalization (RQ 7):** Does the learned navigational policy represent a generalized active perception heuristic that remains effective across topographically diverse regions not encountered during training?

6.6.1 Experimental Setup and Baselines

The Phase II experiments utilized the same computational infrastructure described in chapter 5. While the total project resources were shared across both developmental stages, the transition to high-resolution aerial imagery and continuous dynamics represented a significant increase in computational intensity. Despite utilizing a lower total number of environment interactions, the massive increase in input volume ($H_{\text{in}} \times W_{\text{in}} = 84^2$ versus 11^2) meant that Phase II accounted for the substantial majority of the project's wall-clock compute time per training iteration.

Continuous Mission Parameters In this phase, the mission constraints were scaled to reflect realistic UAV operations while maintaining consistency with the definitions in section 6.2.1. The drone maintained a constant simulated altitude of $h = 100\,\text{m}$ with an FOV of $\alpha_{\text{fov}} = 84°$. According to the sensor model, this results in a local observation window of $H_c \times W_c = 360 \times 360$ pixels. This GSD of $\rho = 0.5\,\text{m/px}$ represents a ground footprint of $180 \times 180\,\text{m}$. Prior to evaluation by the policy, this physical window is downsampled to the neural network input resolution of $H_{\text{in}} \times W_{\text{in}} = 84 \times 84$. This decoupling preserves the physical accuracy of the engine's reward calculations while ensuring the multi-modal convolutional backbone remains computationally tractable during training. To account for the non-holonomic unicycle constraints, the simulation frequency was fixed at 10 Hz. Consequently, the maximum episode horizon was increased to $T_{\max} = 500$ steps. This extension provided the vehicle with a sufficient temporal budget to reach the goal radius of 20 m.

Training Configuration and Hyperparameters The optimization utilized the Adam optimizer with a learning rate of $\alpha = 3 \times 10^{-4}$ for most configurations. For the reinforcement learning paradigms, a discount factor of $\gamma = 0.99$ was applied. Unlike the Phase I grid-world, the training budget was capped at 1.5×10^6 steps because the inclusion of the saliency prior $\mathbf{M}_t$ facilitated faster convergence of the visual encoder. The algorithm-specific hyperparameters are summarized in table 6.2.

Table 6.2 Hyperparameter configurations for Phase II optimization paradigms.

Hyperparameter	PPO / LSTM-PPO	DAgger
Learning Rate (α)	3×10^{-4}	3×10^{-4}
Batch Size	256	64
Discount Factor (γ)	0.99	-
Optimization Epochs	10	10
Rollout Buffer Size	8 192	-
Entropy Coefficient	0.01	-
Clip Range (ϵ_{clip})	0.2	-
LSTM Hidden Units	128 (LSTM only)	-
DAgger Iterations	-	10 (at 20k steps)
Total Training Steps	1.5×10^6	600×10^3
Depletion Rate (λ_{dep})	0.05 (empirical)	

6.6.2 Evaluation Framework and VPR Metrics

To assess the performance of the proposed system in the continuous aerial domain, we extend the standardized evaluation framework introduced in section 5.5.2. While success rate and path optimality remain essential for benchmarking navigational competence, testing the central hypothesis that active perception improves robustness requires a dedicated quantitative evaluation of the resulting localization quality. Consequently, we introduce a suite of metrics centered on the VPR retrieval performance.

This integrated evaluation compares the trained agents against two deterministic reference trajectories, adapted for the Phase II unicycle dynamics:

1. **Naive Shortest Path** ξ_{naive} – A straight-line unicycle trajectory from start to goal. It represents a baseline policy that prioritizes geometric efficiency while entirely ignoring visual salience.

2. **Privileged Feature-Aware Oracle** ξ^* – A baseline generated by the Dijkstra-Pure-Pursuit controller (section 6.4). This heuristic has privileged access to the global saliency map to plan a path that maximizes feature exposure. It serves as a high-competence reference for both visual feature capture and VPR localization quality.

A key implementation detail is that all VPR-based metrics are evaluated post-hoc using the axis-aligned *Perceptual View* generated by the virtual EIS module (section 6.2.5). This ensures that descriptors $\psi(O_t)$ are perfectly aligned with the reference database $\mathcal{V}$ while remaining strictly independent of the saliency prior $\mathbf{M}_t$ used during navigation. By deriving metrics only from the RGB data, we avoid circularity and evaluate the actual retrieval quality of

the selected path. To maintain computational tractability, the VPR pipeline is evaluated at a frequency $f_{eval} = 1\,\text{Hz}$ over a discrete set of time indices $\mathcal{T}_{sub} \subset \{0, \dots, T\}$. This sub-sampling, with cardinality $N_{sub} = |\mathcal{T}_{sub}|$, assesses the retrieval performance at physically significant intervals while accounting for the high spatial overlap between successive camera footprints.

Recall@k (R@k) We evaluate retrieval success using the Recall@k metric defined in eq. (4.10). For each observation O_t where $t \in \mathcal{T}_{sub}$, the perceptual North-Up crop is processed by the SelaVPR++ backbone to generate a query descriptor. A retrieval is considered successful if at least one of the top-k ranked database candidates falls within a spatial radius $r_{\text{vpr}} = 50\,\text{m}$ of the ground-truth position $\mathbf{p}_t$. While VPR benchmarks [52] often report higher values of k, this work focuses strictly on Recall@1, as a reactive navigation policy requires unambiguous pose identification for reliable trajectory correction.

Average Localization Error (ALE) To quantify the geometric precision of the visual navigation, we define the Average Localization Error (ALE) as the mean Euclidean drift between the agent's actual position and the retrieved visual landmarks. We express this error in meters (m) to remain consistent with the physical scale of the environment.

Specifically, let $\hat{\mathbf{p}}_t$ represent the coordinate associated with the top-ranked reference image retrieved from the database $\mathcal{V}$ at time t. Since both the ground truth $\mathbf{p}_t$ and the estimate $\hat{\mathbf{p}}_t$ are defined in the grid cell space, the error is first calculated in cells and then converted to the metric layer. We utilize the constants s and ρ from section 6.2.1 to perform this conversion. The ALE for a trajectory ξ, averaged over the N_{sub} sampled points, is calculated as:

$$\text{ALE}(\xi) = (s\rho)\frac{1}{N_{\text{sub}}} \sum_{t \in \mathcal{T}_{\text{sub}}} \|\mathbf{p}_t - \hat{\mathbf{p}}_t\|_2. \tag{6.27}$$

This formulation ensures the resulting error reflects the physical distance in the metric S-JTSK space.

6.6.3 Baseline Perceptual Performance

Before evaluating the learned policies, we establish a reference for the navigation task by comparing the two deterministic baselines: the *Naive Shortest Path* and the *Feature-Aware Oracle*. This comparison, conducted over 500 randomized validation episodes, isolates the impact of spatial path selection on VPR reliability, independent of any learning-induced kinematic errors.

The Oracle serves as the privileged baseline. Because it utilizes global knowledge of the saliency field alongside an exhaustive Dijkstra search and Pure Pursuit controller, its performance represents the empirical ceiling for any policy operating within the current SelaVPR++ perception framework.

It is important to note, however, that the Oracle enjoys three informational and kinematic privileges unavailable to the RL agent at deployment time: full access to the global saliency map $\mathcal{F}_0$, perfect metric self-positioning without localization feedback latency, and 8-way Dijkstra connectivity unconstrained by unicycle kinematics. The performance gap between the Naive baseline and the Oracle therefore measures a practical operational difference rather than a theoretical optimum, and the fraction of that gap closed by the learned policy should be interpreted accordingly. The aggregated results are summarized in table 6.3.

Table 6.3 Comparative perceptual performance of deterministic baselines over 500 validation episodes.

Metric	Naive Baseline	Oracle Baseline	Delta
Mean R@1 (↑)	0.658 ± 0.15	0.708 ± 0.13	+0.05
Mean ALE [m] (↓)	53.35 ± 23.29	46.46 ± 16.10	-6.89

The Oracle demonstrates perceptual dominance in 66.4% of the evaluated trials, achieving a 12.9% average reduction in ALE compared to the Naive baseline. The remaining cases, where the Naive path achieves comparable top-1 recall, generally involve feature-saturated scenarios where landmark density is high enough that even shortest Euclidean paths meet the VPR confidence thresholds. Consequently, the Oracle's 0.708 Recall@1 and 46.46 m ALE are established as the target performance boundaries. While the absolute gain is approximately 5 meters, this must be interpreted in the context of the 90 m database stride (section 6.1.2). Within such a sparse reference layout, the measured ALE is inherently quantized. A paired t-test on the per-episode ALE differences nonetheless confirms the Oracle's advantage as statistically significant ($t = -6.569$, $p < 0.001$), establishing the Naive–Oracle gap as a well-defined and reliable performance target. These results indicate that the feature-aware objective successfully pulls the trajectory toward cells with more distinct semantic signatures, even when the evaluation grid is coarse.

Furthermore, tests at a 5.0 Hz sampling frequency yielded negligible metric changes ($< 0.5\%$). This confirms the 1.0 Hz rate sufficiently captures spatial error variance. Even at the maximum velocity $V_{\max} = 12$ m/s, the resulting 12 m sampling interval remains significantly smaller than the database stride.

6.6.4 Objective Balancing and Reward Dynamics

The non-holonomic dynamics of the continuous environment significantly constrain objective balancing in Phase II. Unlike the discrete grid transitions of Phase I, where movement between adjacent states was direction-agnostic, the unicycle kinematics introduced in section 6.2.4 impose a substantial cost on trajectory deviation. Because altering the vehicle's heading requires a physical

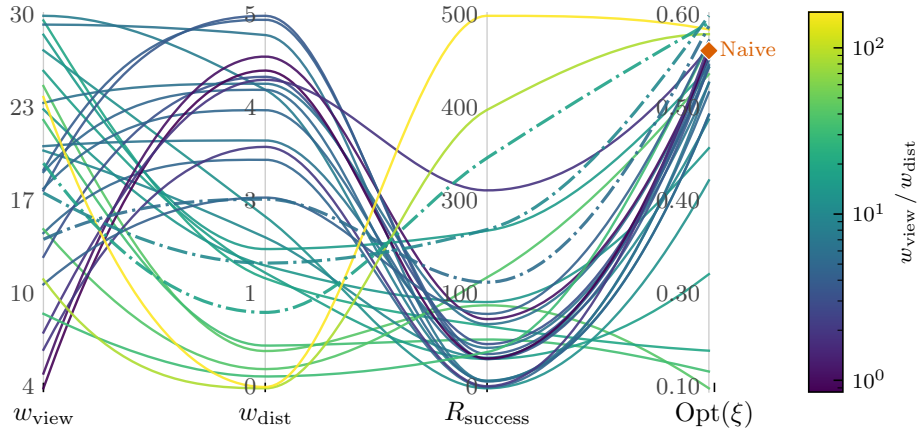


Figure 6.5 Parallel coordinate plot illustrating the global sensitivity of the continuous navigation policy ($N = 30$ runs). Trajectories are color-coded by the $w_{\text{view}}/w_{\text{dist}}$ ratio.

arc and a corresponding adjustment in longitudinal velocity, every maneuver to pursue a salient landmark consumes a portion of the fixed episode horizon $T_{\max}$. Consequently, the agent must internalize a more sophisticated trade-off between the immediate reward R_t of landmark acquisition and the global progress toward the goal.

To identify the optimal trade-off between landmark acquisition and global progress, we performed a 30-run Bayesian hyperparameter sweep utilizing PPO as the underlying learning backbone. We denote this specific configuration, characterized by its explicit saliency-augmented observation space and static reward weights, as the **Saliency-Augmented Static Policy (SASP)**. This designation allows for clear differentiation between the primary model and the experimental variants evaluated in the subsequent ablation studies. To maintain methodological consistency, the search space and bounding constraints for the SASP weights ($w_{\text{view}}, w_{\text{dist}}, R_{\text{success}}$) were kept identical to those utilized in Phase I. The resulting behavioural landscape, visualized in fig. 6.5, reveals that the continuous hyperparameter space exhibits a clear threshold in policy stability.

The continuous domain exhibits a markedly tighter "competence corridor" than observed in the discrete phase. While moderate visual weight ratios ($w_{\text{view}}/w_{\text{dist}} \in [5, 15]$) successfully pull the policy above the naive baseline, pushing the ratio higher triggers catastrophic policy collapse. Because turning to re-orient requires deceleration and consumes significant episode time, aggressive feature-seeking policies frequently become trapped in terminal loitering orbits, exhausting the time horizon before reaching the goal.

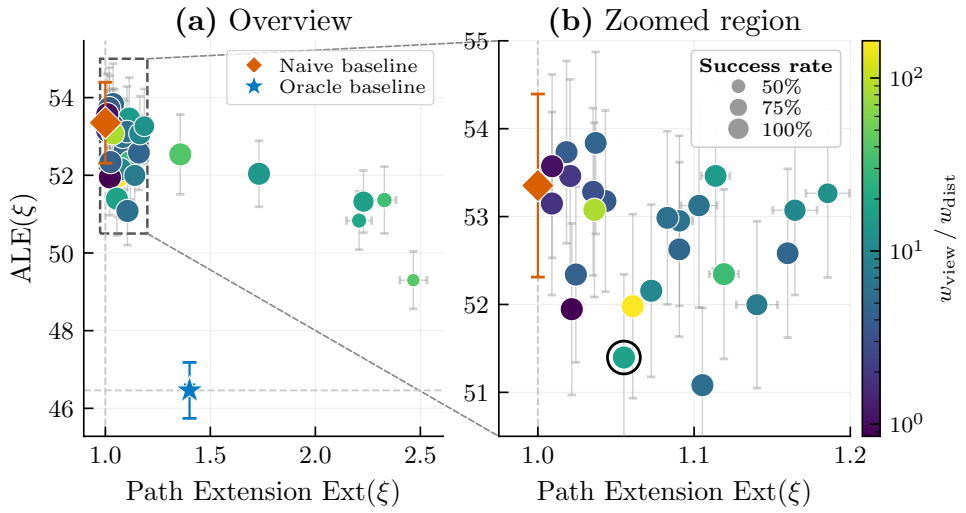


Figure 6.6 Trade-off analysis between navigational overhead $\text{Ext}(\xi)$ and localization precision $\text{ALE}(\xi)$. **(a) Overview** highlights the bounded gap between the naive shortest path ($\diamond$) and the globally-aware Feature-Aware Oracle ($\star$). **(b) Detail of the competence region** showing the learned RL policies. Marker diameters denote the empirical success rate $\text{Succ}(\pi)$. Error bars represent ± 1 standard error across $N_{\text{ev}} = 500$ evaluation episodes per configuration. The selected configuration identified via the filtering protocol in section 6.6.4 is highlighted with a ring.

Navigational Detours and Perceptual Precision To rigorously quantify the utility of these detours, we evaluate the downstream task performance. While Phase I relied on an abstract *Feature Capture* ratio, the integration of the true VPR pipeline allows us to measure the direct reduction in Average Localization Error, denoted as $\text{ALE}(\xi)$.

As shown in fig. 6.6, plotting the path extension $\text{Ext}(\xi)$ against $\text{ALE}(\xi)$ highlights the behavioural distribution of the trained policies relative to the deterministic baselines. The naive baseline ($\diamond$) represents the lower bound of navigational effort but suffers from a mean $\text{ALE}(\xi)$ of 53.35 m due to its blind traversal through perceptually aliased zones. Conversely, the Dijkstra Oracle ($\star$) utilizes global map knowledge to execute detours ($\text{Ext}(\xi) \approx 1.40$), achieving a reference $\text{ALE}(\xi)$ of 46.46 m. The RL agents identify that under partial observability and non-holonomic constraints, such extensive detours are non-productive; instead, they converge toward a constrained region where $\text{Ext}(\xi) \in [1.02, 1.10]$.

Policy Selection and Results To identify the most robust flight policy, we applied the multi-stage filtering protocol established in section 5.5.3. We retained the strict 95% mission reliability threshold and enforced a perceptual constraint of $\text{ALE}(\xi) < 52.0$ m to ensure a definitive improvement over the

Table 6.4 Aggregated performance comparison of the selected continuous PPO configuration. For $N_{\text{ev}} = 500$ evaluation episodes, reported values represent the mean ± one standard deviation.

Method	Success [%]	Opt(ξ) (↑)	Ext(ξ) (↓)	ALE(ξ) (↓)	R@1(ξ) (↑)
Naive Baseline	100.0 ± 0.0	0.597 ± 0.22	1.000 ± 0.00	53.35 ± 23.29	0.658 ± 0.15
SASP (sweep)	**100.0** ± 0.0	**0.634** ± 0.22	**1.055** ± 0.07	**51.40** ± 21.13	**0.671** ± 0.15
Oracle Baseline	100.0 ± 0.0	1.000 ± 0.00	1.401 ± 0.45	46.46 ± 16.10	0.708 ± 0.13

naive path. Within this resulting competence corridor, the policy maximizing the Optimality Score Opt(ξ) was selected.

This process identified the best-performing policy with $w_{\text{view}} \approx 19.45$, $w_{\text{dist}} \approx 1.10$, and $R_{\text{success}} \approx 311.16$ (ratio $\approx 17.7 : 1$). It is notable that the visual weight ratio is significantly higher than the 2.6 : 1 ratio identified in the discrete environment of Phase I. This shift is primarily attributed to the increased kinematic and temporal costs of maneuvering in a continuous space; a stronger visual incentive is required to prioritize exploratory detours over the geometric shortest-path prior. The performance of this agent is benchmarked in table 6.4.

The empirical results confirm that the selected policy successfully internalized the multi-objective trade-off. By accepting a 5.5% increase in path extension, the agent achieved a reduction in ALE(ξ) from 53.35 m to 51.40 m, demonstrating that the agent discovered mathematically efficient maneuvers toward salient landmarks and successfully approximated the behaviour of a globally-informed heuristic without compromising mission safety. The selected hyperparameter configuration is carried forward to the multi-seed validation in section 6.6.5, which provides the definitive reproducibility-validated performance estimate.

6.6.5 Structural Ablation Studies

To rigorously validate the architecture of the proposed navigation system, we conduct a structural ablation study. The objective is to isolate the individual contributions of the explicit perceptual prior $\mathbf{M}_t$ and to determine whether the continuous unicycle kinematics require artificial, non-stationary reward scheduling (Ψ_t) to prevent terminal loitering. By systematically evaluating these components, we establish the minimal necessary complexity for the continuous POMDP. Crucially, each variant is validated across $N_{\text{sd}} = 5$ independent seeds, replacing the single-run sweep results above with a reproducibility-validated performance estimate.

We evaluate three distinct architectural variants:

1. **Implicit Static Policy (ISP):** This variant serves as the perceptual baseline. The saliency tensor $\mathbf{M}_t$ is entirely removed from the observation space O_t, forcing the agent to estimate landmark uniqueness implicitly from raw

RGB textures $\mathbf{I}_t$ and visitation history $\mathbf{D}_t$. It utilizes static reward coefficients to test the fundamental necessity of the perceptual prior.

2. **Saliency-Augmented Dynamic Policy (SADP):** An experimental extension of the proposed system where static reward weights are replaced by the non-stationary, progress-based decay schedule Ψ_t defined in section 6.3.2. The hypothesis is that reducing visual incentives near the goal suppresses terminal loitering and improves mission reliability.

3. **Saliency-Augmented Static Policy (SASP):** The primary proposed architecture. The agent utilizes the full saliency-augmented observation space and optimizes against fixed reward coefficients. This model represents the final system configuration selected for its balance of performance and architectural stability.

Comparison of Optimization Capacity To determine the performance limits of each architecture, we conducted independent 30-run Bayesian hyperparameter sweeps. This ensures the comparisons are not biased by weight configurations that favour one observation model over another. The resulting distribution of 90 training runs is shown in fig. 6.7.

The scatter plots reveal a clear performance ceiling for the ISP variant (purple). Although the Bayesian sampler explored configurations with extreme visual incentives, these agents failed to achieve systematic improvements in localization error. As seen in fig. 6.7b, any detour away from the naive straight-line path caused a significant drop in the Optimality Score, confirming that without the explicit $\mathbf{M}_t$ prior, the model cannot distinguish unique landmarks from repetitive textures, which causes exploratory behaviour to collapse into inefficient wandering.

Conversely, both saliency-augmented variants successfully penetrate the high-competence region where $\mathrm{Opt}(\xi) > 0.60$. Visually, the SADP configurations (light green) exhibit a higher density of successful trials at higher optimality scores compared to the static SASP counterpart (teal). This suggests that the non-stationary reward schedule provides a more forgiving optimization landscape during the initial hyperparameter search, making it easier for the sampler to identify high-performing weight ratios. However, as quantified in the subsequent multi-seed validation, this peak performance observed during the sweep does not necessarily translate to architectural stability across independent training runs.

Ablation Configuration Selection To isolate architectural capacity from sub-optimal parameterization, we selected the best-performing configuration for each variant using the filtering protocol established in section 6.6.4. These specific hyperparameter sets, which served as the basis for the multi-seed validation, are detailed in table 6.5. The weight distribution reveals how each architecture compensates for its structural limitations. The ISP variant required

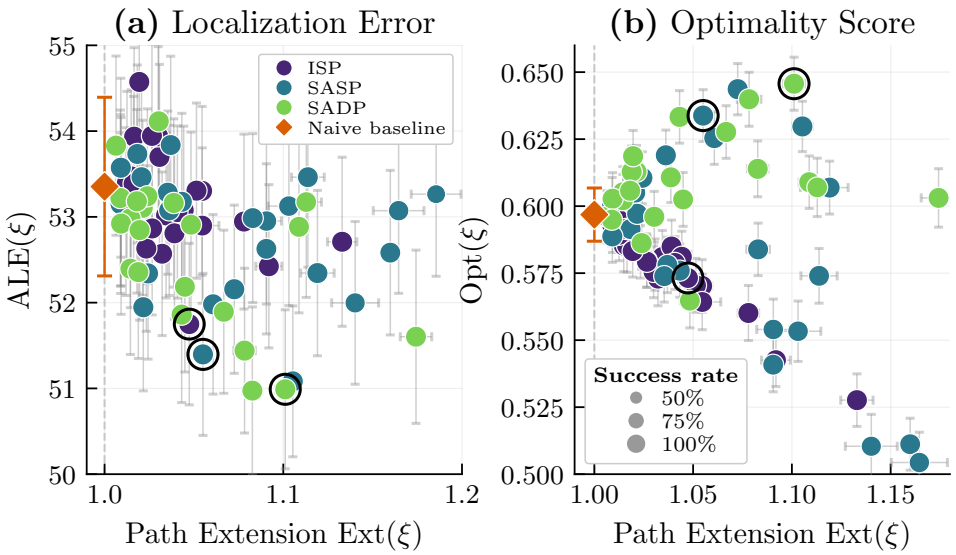


Figure 6.7 Performance distribution across 90 hyperparameter trials comparing the three architectural variants. **(a) Localization Error** and **(b) Optimality Score** are plotted against path extension. To highlight the primary competence region, the axes bounds intentionally omit extreme exploratory outliers and the Oracle baseline ($\text{Ext}(\xi) \approx 1.40$). Marker diameters denote the empirical success rate $\text{Succ}(\pi)$. Error bars represent ± 1 standard error across $N_{\text{ev}} = 500$ evaluation episodes per configuration. The highest-performing configuration from each variant is highlighted with a black ring.

an extreme visual weight (42.10) and a massive terminal reward (1143.62) merely to achieve convergence, highlighting the inefficiency of implicit feature extraction. The SASP and SADP models, guided by the perceptual prior, optimized at much more stable coefficients, though they handled the goal-seeking drive differently based on their static or dynamic natures.

Quantitative Performance Summary The selected configurations were trained across $N_{\text{sd}} = 5$ independent random seeds to account for network initialization variance. The converged policies were evaluated over 500 deter-

Table 6.5 Selected hyperparameter configurations for the structural ablation study.

Variant	w_{view}	w_{dist}	R_{success}	$w_{\text{view}}/w_{\text{dist}}$
ISP	42.10	1.85	1143.62	22.7
SADP	20.88	4.10	18.95	5.1
SASP	19.45	1.10	311.16	17.7

ministic episodes. The aggregated metrics in table 6.6 represent the definitive verdict on their architectural robustness.

The empirical results definitively prove the necessity of the explicit saliency prior while simultaneously challenging the hypothesis regarding reward dynamics. The ISP variant successfully learned basic navigational competence, achieving a 99.00% success rate. However, its performance on the active perception task exposes the severe limitations of raw RGB observations. Without the saliency prior, the agent's attempts to acquire features ($\mathrm{Ext}(\xi) = 1.063$) are effectively blind. This spatially inefficient wandering causes the ISP agent to achieve an Optimality Score (0.565) and an Average Localization Error (53.50 m) that are strictly worse than those of the naive straight-line baseline. This confirms that without the explicit $\mathbf{M}_t$ prior to resolve perceptual aliasing, exploratory detours become actively harmful to the mission's spatial efficiency.

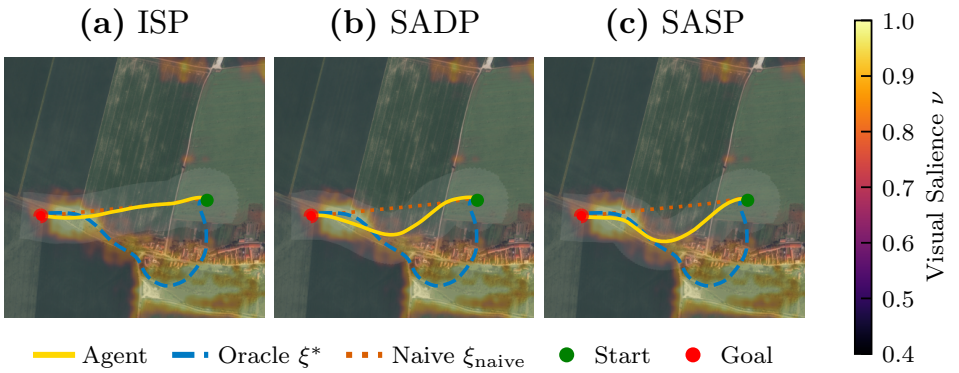
Furthermore, the comparison between SASP and SADP reveals that the static formulation is strictly superior in both performance and multi-seed reliability. The SASP model achieves the highest Optimality Score (0.635) and the lowest localization error (51.74 m), while maintaining significantly tighter standard deviations (0.91 m vs 1.21 m) than the dynamic alternative. The data suggests that unicycle kinematics serve as a natural regularizer. Since turning maneuvers incur a substantial temporal and velocity penalty, the agent inherently avoids infinite loitering cycles without requiring artificial reward decay. By closing **23.4**% of the performance gap between the naive path and the privileged Oracle, SASP is formally selected as the final architecture for the paradigm comparisons and generalization experiments.

Qualitative Behavioural Analysis To investigate the decision-making biases induced by the architectural differences, a representative episode was selected for visual comparison. The resulting trajectories are compared on a shared evaluation arena in fig. 6.8. This specific topography features a high-saliency village corridor positioned south of the naive straight-line path.

The ISP agent (fig. 6.8a) illustrates the structural limit of navigating without a perceptual prior. Lacking the saliency map $\mathbf{M}_t$, it essentially defaults to the geometric shortest path. By ignoring the distinctive semantic features

Table 6.6 Quantitative comparison of the evaluated architectural variants. Values represent the mean ± one standard deviation computed across $N_{\mathrm{sd}} = 5$ independent training seeds ($N_{\mathrm{ev}} = 500$ evaluation episodes per seed).

Model	Success [%] (↑)	Opt(ξ) (↑)	Ext(ξ) (↓)	ALE(ξ) (↓)	R@1(ξ) (↑)
Naive Baseline	100.0 ± 0.00	0.597 ± 0.00	1.000 ± 0.00	53.35 ± 0.00	0.658 ± 0.00
Oracle Baseline	100.0 ± 0.00	1.000 ± 0.00	1.401 ± 0.00	46.46 ± 0.00	0.708 ± 0.00
ISP	99.00 ± 0.37	0.565 ± 0.00	1.063 ± 0.01	53.50 ± 0.44	0.659 ± 0.00
SADP	98.16 ± 1.81	0.592 ± 0.02	**1.090** ± 0.04	52.28 ± 1.21	0.660 ± 0.01
SASP	**99.44** ± 0.52	**0.635** ± 0.01	1.095 ± 0.03	**51.74** ± 0.91	**0.669** ± 0.01



Track	Panel	Opt(ξ)(↑)	ALE(ξ) (↓)	R@1(ξ)(↑)
Naive ξ_{naive}	–	0.291	86.10	0.571
Oracle ξ^*	–	1.000	60.19	0.636
ISP Agent	(a)	0.303	127.73	0.538
SADP Agent	(b)	0.438	103.16	0.429
SASP Agent	(c)	**0.622**	**57.40**	**0.667**

Figure 6.8 Comparison of navigation trajectories across the evaluated variants on a shared arena. The table summarizes the performance of the three agents relative to the common environment baselines.

of the village, the agent suffers from severe perceptual aliasing across the homogeneous terrain, incurring a massive ALE of 127.7 m. The SADP variant (fig. 6.8b) utilizes the prior to identify the salient detour but breaks away prematurely. Because the aggressively increasing progress weight Ψ_t forces an early return to the goal-seeking vector, the agent abandons the unique landmarks before the maneuver is complete, resulting in a suboptimal ALE of 103.2 m.

In contrast, the SASP model (fig. 6.8c) maintains a more consistent alignment with the Oracle path. By preserving the visual incentive throughout the flight, the agent fully traverses the high-saliency corridor before the physical momentum and spatial rewards guide it toward the destination. This configuration achieved an ALE of 57.40 m and an Optimality Score of 0.622. In this specific instance, the SASP agent records a lower localization error than the Oracle reference (60.19 m) and a higher Recall@1 (0.667 vs 0.636). This demonstrates that while the Oracle is geometrically optimal with respect to the proxy cost map, the learned reactive policy can discover local trajectories that yield strictly superior visual feature descriptors.

6.6.6 Comparative Analysis of Learning Paradigms

With the architectural foundation of the continuous POMDP established, we evaluate how different learning paradigms navigate the non-holonomic optimization landscape. We compare standard PPO against a recurrent variant (LSTM-PPO) and an imitation learning baseline (DAgger). To ensure a rigorous evaluation, all paradigms utilize the Saliency-Augmented observation space ($\mathbf{M}_t$) and the static reward formulation validated in section 6.6.5.

The learning dynamics during training, visualized in fig. 6.9, reveal a stark contrast in optimization stability. Both PPO and LSTM-PPO exhibit smooth, rapid convergence, surpassing the naive spatial baseline after approximately 400×10^3 environment interactions and stabilizing into high-competence policies. Conversely, the DAgger agent entirely fails to establish a reliable goal-seeking strategy. Its optimality score remains highly erratic and significantly below the baseline throughout its interactive rollout phase.

Quantitative Navigational Performance To precisely quantify this divergence, the converged policies were evaluated over 500 deterministic validation episodes. The aggregated metrics across the five independent random seeds are presented in table 6.7.

The empirical data emphasizes the catastrophic collapse of imitation learning in this domain. While DAgger successfully replicated the Oracle's behaviour in the discrete grid of chapter 5, its continuous success rate plummets to $53.04 \pm 31.24\%$. This failure is a direct consequence of unicycle kinematics mathematically exacerbating the covariate shift problem. Because the agent regresses continuous actions via mean squared error against the Oracle's exact Pure Pursuit commands, infinitesimal yaw errors inevitably accumulate. Without the ability to execute instantaneous lateral corrections or an intrinsic spatial reward to pull it toward the goal coordinate, the agent quickly drifts outside the expert's nominal state distribution and enters divergent terminal spirals.

Furthermore, the comparison between the standard and recurrent actor-critic configurations provides critical insight into the system's spatial representation. Although LSTM-PPO achieves the highest geometric optimality score (0.639), the recurrent hidden state induces a conservative, risk-averse bias. It actively suppresses exploratory detours ($\mathrm{Ext}(\xi) = 1.057$), which degrades the active perception objective and results in a higher Average Localization Error (52.06 m) compared to standard PPO (51.74 m). Since the augmented observation vector O_t already stacks temporal visual frames and the dynamically rotated saliency prior $\mathbf{M}_t$, it sufficiently resolves the partial observability. Consequently, forcing the network through an LSTM bottleneck introduces unnecessary optimization constraints without yielding any perceptual benefits, confirming standard PPO as the superior continuous architecture.

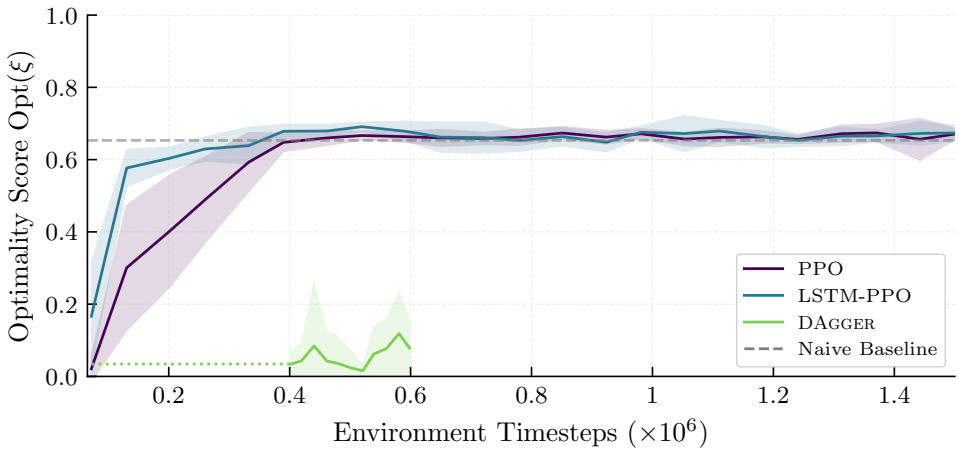


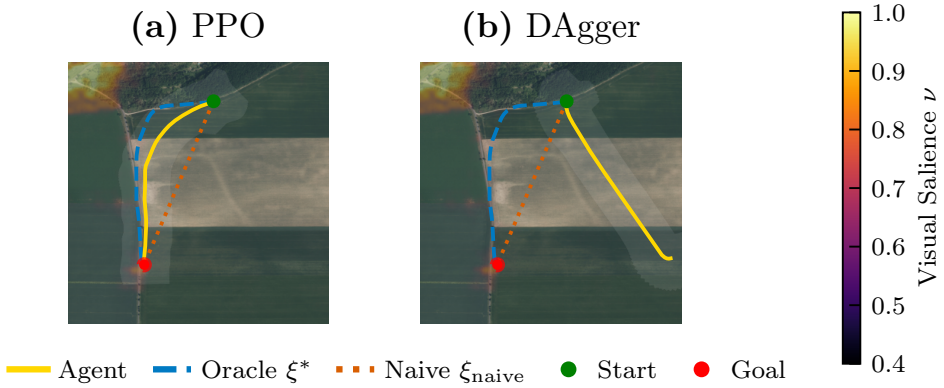
Figure 6.9 Learning dynamics across the evaluated continuous optimization paradigms. Solid lines denote the mean Optimality Score $\mathrm{Opt}(\xi)$ across $N_{\mathrm{sd}} = 5$ seeds, with shaded regions representing ± 1 standard deviation. The horizontal dashed line indicates the baseline performance of the naive shortest path. For DAgger, the curve spans the initial supervised dataset (4×10^5 steps) followed by 10 interactive rounds (2×10^4 steps each).

Qualitative Analysis of Decision Biases To investigate the quantitative disparities observed in table 6.7, we visualize the decision-making patterns of the converged policies on a shared evaluation arena in fig. 6.10. The environment features a distinct high-saliency corridor (the road network on the left) separated from the naive shortest path by featureless agricultural terrain.

As illustrated in fig. 6.10a, the PPO policy successfully internalizes the active perception trade-off. Recognizing that the geometric shortest path traverses a perceptually aliased region, the agent executes a deliberate detour to follow the salient road. By approximating the privileged Oracle's trajectory, the agent doubles its $\mathrm{Opt}(\xi)$ score to 0.495 compared to the naive baseline's 0.249. More importantly, as quantified in the accompanying table, this detour successfully reduces the Average Localization Error from 113.49 m down to

Table 6.7 Quantitative comparison of learning paradigms in the continuous domain. Results represent the mean $\pm$ one standard deviation across $N_{\mathrm{sd}} = 5$ independent training seeds ($N_{\mathrm{ev}} = 500$ episodes per seed).

Paradigm	Success [%] (↑)	Opt(ξ) (↑)	Ext(ξ) (↓)	ALE(ξ) (↓)	R@1(ξ) (↑)
Naive Baseline	100.00 ± 0.00	0.597 ± 0.00	1.000 ± 0.00	53.35 ± 0.00	0.658 ± 0.00
Oracle Baseline	100.00 ± 0.00	1.000 ± 0.00	1.401 ± 0.00	46.46 ± 0.00	0.708 ± 0.00
PPO	99.44 ± 0.52	0.635 ± 0.01	1.095 ± 0.03	**51.74** ± 0.91	**0.669** ± 0.01
LSTM-PPO	**99.72** ± 0.33	**0.639** ± 0.01	**1.057** ± 0.02	52.06 ± 0.54	0.662 ± 0.01
DAgger	53.04 ± 31.24	0.266 ± 0.15	1.170 ± 0.12	53.62 ± 1.01	0.646 ± 0.02



Track	Panel	Opt(ξ)(↑)	ALE(ξ) (↓)	R@1(ξ)(↑)
Naive ξ_{naive}	–	0.249	113.49	0.267
Oracle ξ^*	–	1.000	72.04	0.500
PPO Agent	(a)	**0.495**	**79.99**	**0.333**
DAgger Agent	(b)	0.000	120.95	0.133

Figure 6.10 Qualitative comparison of trajectory selection across paradigms on a shared evaluation arena. The semi-transparent gray overlays indicate the sequence of camera footprints sampled during the mission. The table summarizes the performance of the agents relative to the common environment baselines. **(a)** The PPO agent successfully deviates from the naive path to follow the salient road, significantly improving localization. **(b)** The DAgger agent misses the salient corridor; without a global reward signal to facilitate recovery, it drifts off-target. As noted in the table, the resulting kinematic drift makes the imitation learner actively worse than the blind naive path across all metrics.

79.99 m, demonstrating a clear empirical improvement in VPR confidence over the blind geometric path.

In contrast, fig. 6.10b demonstrates the fundamental robustness limit of behavioural cloning in a non-holonomic continuous space. The DAgger agent fails to execute the sharp initial turn required to reach the salient corridor. Once pushed into the textureless field, it encounters a severe covariate shift. Because the imitation learner lacks an intrinsic value-based "pull" toward the goal coordinates, it cannot compute a corrective recovery trajectory. Instead, a slight, uncorrected yaw bias causes the agent to drift aimlessly away from the target, resulting in complete mission failure due to episode termination ($\text{Opt}(\xi) = 0.000$). Consequently, its ALE inflates to 120.95 m, which is strictly worse than simply executing a blind, straight-line flight. This confirms that an explicit spatial reinforcement learning objective is necessary to anchor the flight policy against kinematic drift.

6.6.7 Zero-Shot Generalization to Unseen Topography

To definitively prove that the agent internalized a generalized active perception heuristic rather than merely overfitting to the spatial layouts of the training environments, we evaluate the selected SASP model on two held-out $5 \times 5\,\text{km}$ cadastral regions. These evaluation regions share zero geographic overlap with the training set. Region A (49.140° N, 16.123° E) consists predominantly of unstructured natural terrain and water bodies, providing a rigorous test of robustness against organic perceptual aliasing. Region B (49.457° N, 17.929° E) features mixed rural topography with sparse infrastructure.

Quantitative Generalization Performance The SASP model was evaluated over 500 randomly initialized episodes distributed evenly across the two held-out maps. The results are compared against the deterministic spatial baselines in table 6.8.

The metrics indicate that the transition to unseen topography incurs no significant performance degradation or loss of mission reliability. The agent maintains a perfect 100% success rate while actively extending its geometric path by 9.2% to acquire visual features, demonstrating that the learned active-perception heuristic transfers without retraining.

The large per-policy standard deviations (±18–20 m) reflect topographic heterogeneity across the 500 randomly sampled arenas rather than per-episode measurement noise: some arenas contain dense salient infrastructure and yield low ALE for any policy, whereas featureless or water-dominated environments push the quantization floor of the 90 m-stride reference database to its operational limit regardless of trajectory quality. This variance is a property of the evaluation distribution, not of the policy. To isolate the policy-level contribution, paired one-sided t-tests were conducted on the per-episode ALE differences across all 500 held-out arenas, with each pair sharing identical arena geometry. The paired design cancels the dominant inter-arena variance, and all three primary metrics confirm statistically significant improvement: Optimality Score ($t = 3.997$, $p < 0.001$), Recall@1 ($t = 2.819$, $p = 0.003$), and Average Localization Error ($t = -2.018$, $p = 0.022$). The mean ALE reduction of 1.78 m (31.2% of the Naive–Oracle gap established in section 6.6.3; note that the higher percentage relative to the 23.4% in-domain figure reflects the narrower Naive–Oracle gap in the held-out regions, not a larger absolute improvement) therefore represents a confirmed, consistent policy-level gain. Its modest absolute magnitude reflects the 90 m database stride, not the absence of a real effect.

Because the observation vector O_t is strictly egocentric and the saliency prior $\mathbf{M}_t$ is dynamically computed relative to the vehicle's footprint, the agent applies the same cost-minimization heuristic regardless of absolute geographic position. The zero-shot success rate and the directional consistency of the

Table 6.8 Quantitative performance of the SASP architecture on held-out regions. Values represent the mean ± one standard deviation computed across $N_{\text{ev}} = 500$ evaluation episodes.

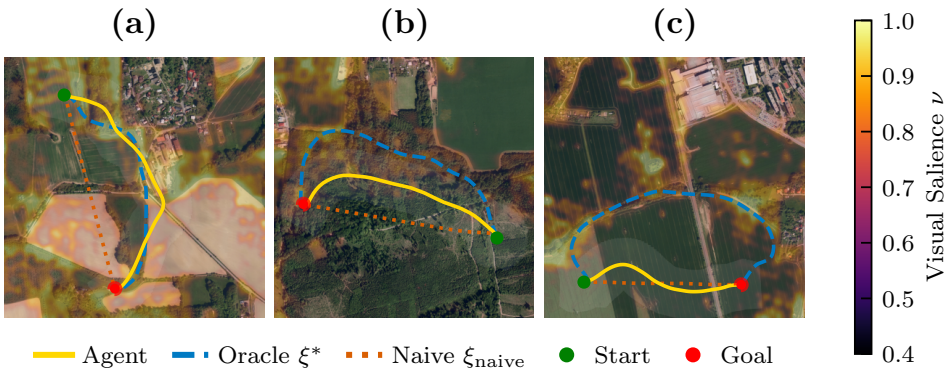
Method	Success [%] (↑)	Opt(ξ) (↑)	Ext(ξ) (↓)	ALE(ξ) (↓)	R@1(ξ) (↑)
Naive Baseline	100.0 ± 0.0	0.588 ± 0.21	1.00 ± 0.00	48.45 ± 20.16	0.686 ± 0.15
SASP	**100.0** ± 0.0	**0.621** ± 0.21	**1.09** ± 0.09	**46.67** ± 18.52	**0.705** ± 0.14
Oracle Baseline	100.0 ± 0.0	1.000 ± 0.00	1.35 ± 0.40	42.75 ± 13.20	0.737 ± 0.12

ALE improvement across diverse topographies confirm that the policy learned a generalized boundary-following behaviour rather than overfitting to specific training coordinates.

Qualitative Analysis and Feature Acquisition To visualize the generalized behaviour and the operational boundaries of the system, fig. 6.11 illustrates the agent trajectories overlaid on the continuous spatial cost fields of the held-out maps.

In environments with distinct infrastructure (fig. 6.11a and fig. 6.11b), the agent successfully internalizes the navigation-perception trade-off. As quantified in the accompanying table, the agent in scenario (a) achieves an optimality score of 0.462, which is nearly three times higher than the naive baseline. This exploratory detour results in a significant boost in localization reliability, with Recall@1 increasing from 0.562 to 0.789. Notably, in scenario (b), the learned policy achieves a Recall@1 of 0.706. This value surpasses even the privileged Oracle reference (0.696).

Scenario fig. 6.11c demonstrates the sensitivity threshold and the planning horizon limits of the reactive controller. In this instance, a massive agricultural field separates the start and goal. While a faint dirt track is visible in the center, its saliency magnitude is insufficient to override the goal-seeking incentive w_{dist}. Furthermore, the global Oracle dictates an extensive detour to the top and the right to reach a high-confidence salient corridor. Because this path lies beyond the 180×180 m observation window, the agent is forced to take a direct path across the textureless void. This results in a "greedy" failure where the agent achieves a lower optimality score than the naive baseline (0.273 vs 0.308) due to unproductive local wandering. However, even in this failure case, the active perception objective provides a safety margin. The agent still achieves an ALE of 71.83 m, which is a substantial improvement over the 85.33 m recorded by the naive path. This confirms that the learned heuristic provides better localization precision than a blind straight line even when the global optimal detour is unobserved.



Arena	Track	Opt(ξ)(↑)	ALE(ξ) (↓)	R@1(ξ)(↑)
(a)	Naive ξ_{naive}	0.176	42.63	0.562
	Oracle ξ^*	1.000	32.08	0.810
	Agent	**0.462**	**39.24**	**0.789**
(b)	Naive ξ_{naive}	0.254	41.72	0.688
	Oracle ξ^*	1.000	36.97	0.696
	Agent	**0.448**	**38.63**	**0.706**
(c)	Naive ξ_{naive}	**0.308**	85.33	0.214
	Oracle ξ^*	1.000	42.40	0.577
	Agent	0.273	**71.83**	**0.538**

Figure 6.11 Qualitative zero-shot generalization on held-out maps. The background heatmap denotes the visual salience ν. **(a, b)** The agent successfully identifies salient structural edges in unseen environments, significantly improving localization over the naive baseline. **(c)** A failure case where the agent's local planning horizon prevents it from discovering the global detour. While the path optimality (Opt) drops below the naive baseline due to exploratory loitering, the agent still achieves superior ALE and R@1.

6.7 Analysis and Preliminary Conclusions

The experimental evaluation within the continuous aerial environment provides the empirical resolution to the research questions defined at the start of this developmental phase. By transitioning to a high-fidelity simulation and utilizing SelaVPR++ saliency, we have demonstrated the feasibility of active perception for localized trajectory refinement.

6.7.1 Summary of Experimental Findings

The outcomes of this developmental cycle directly address the research questions formulated in section 6.6:

- **Kinematic Trade-offs (RQ 4):** The transition to unicycle kinematics significantly altered the optimal reward landscape compared to the grid-

world. The "competence corridor" identified in the hyperparameter sweep required a visual weight ratio ($w_{\text{view}}/w_{\text{dist}} = 17.7$) nearly seven times higher than in Phase I. This confirms that in continuous spaces, the policy requires a much stronger perceptual incentive to overcome the kinematic inertia and the curvature constraints of the vehicle.

- **Utility of Perceptual Priors (RQ 5):** The ablation study proved that the explicit saliency prior ($\mathbf{M}_t$) is the primary driver of perceptual robustness. The Implicit Static Policy (ISP) failed to reduce localization error, confirming that raw RGB features do not provide enough signal to resolve perceptual aliasing zero-shot. Furthermore, the success of the static reward formulation (SASP) indicates that the physical time-cost of turning acts as a natural regularizer, making complex non-stationary reward schedules unnecessary for preventing terminal loitering.

- **Paradigm Sensitivity to Drift (RQ 6):** The comparative analysis highlighted the catastrophic collapse of imitation learning in continuous domains. The DAgger agent's 53.04% success rate is a direct result of kinematic drift. Without the value-based "pull" of the RL distance reward, the agent could not recover from the small, compounding yaw errors that are inherent to continuous actuation. This establishes PPO as the strictly more robust paradigm for non-holonomic navigation tasks.

- **Generalization and Information Horizons (RQ 7):** The zero-shot evaluation on unseen maps confirmed that the learned policy is a generalized heuristic for following semantic boundaries. The agent successfully closed 31.2% of the local Naive–Oracle perceptual gap (a ratio driven by the narrower baseline in these regions; the absolute ALE reduction of 1.78 m is comparable to the 1.61 m improvement in the training domain). However, the failure cases reveal a fundamental limitation: the agent is a greedy local optimizer. Because its observation window is restricted to 180×180 m, it cannot perceive distal high-saliency "islands" that a global planner would utilize, resulting in sub-optimal paths across expansive, featureless terrain.

The Phase II developmental cycle confirms that active perception, when informed by a semantic saliency prior and optimized via policy gradients, successfully improves VPR reliability in realistic aerial scenarios. While the agent cannot solve the global long-horizon planning problem, it demonstrates a robust ability to capture identifiable landmarks along its local trajectory. This validation completes the realization phase of the thesis, providing a functional bridge between raw visual perception and autonomous continuous control.

Chapter 7

Discussion

The empirical evaluation across the discrete grid and continuous aerial simulation confirms that reinforcement learning agents can internalize active perception to improve localization. However, the data also exposes significant theoretical boundaries regarding algorithmic stability, the role of physical constraints in reward shaping, and the effective planning horizons of local policies.

7.1 Asymmetry and Policy Stability

The failure of DAgger in the continuous domain provides a technical insight into the system's operational limits. While the imitation approach was functional in the discrete grid of Phase I, its collapse in Phase II to a success rate of 53.04% results from the interaction of two compounding mechanisms. The first is a fundamental *informational asymmetry* [30]: because the Oracle plans using the global state S_t, its policy is non-Markovian relative to the agent's localized observation space Ω. Identical local observations O_t frequently correspond to divergent expert actions A_t depending on the vehicle's position in the global coordinate frame, causing the agent to converge to an averaged, sub-optimal implicit policy in perceptually aliased regions.

This informational asymmetry was already present during Phase I, where DAgger retained a 90.2% success rate. The approximately 40-point collapse between phases is therefore not attributable to the asymmetry alone. It is amplified by the second mechanism: continuous non-holonomic kinematics. In the discrete grid-world, each action is atomic and directionally precise, so the implicit policy's averaged output remains spatially bounded. In continuous flight, an averaged yaw command produces a persistent angular bias that integrates over time via the unicycle dynamics. Without an intrinsic reward signal R_t to pull the vehicle back toward the goal and correct this drift, the agent spirals into divergent terminal trajectories from which it cannot recover. This confirms that for non-holonomic active perception, an explicit reward-

driven objective is strictly required to quantify terrain utility independently of a privileged expert.

A natural question is whether this failure could be reversed by extending the training budget. The learning dynamics in fig. 6.9 argue against this: unlike PPO and LSTM-PPO, whose optimality scores rise monotonically, the DAgger agent's score remains erratic and oscillates around 0.1 across all ten interactive rounds without any upward trend. This pattern is characteristic of policy saturation rather than under-training. The failure is therefore structural and is not expected to resolve with a larger compute budget.

Furthermore, the redundancy of the LSTM-PPO variant across both phases suggests that the multi-modal observation O_t provides a sufficient proxy for the belief state b_t. In POMDP theory, the objective is to recover a sufficient statistic for the history H_t [10]. While frame-stacking does not formally restore the Markov property of the environment, the temporal window of $N_f = 4$ frames allows the visual branch to internalize first and second-order spatial derivatives of the vehicle motion. This aligns with findings in early deep RL applications where stacked frames capture enough context to resolve local transition ambiguities [12]. The empirical results indicate that for this specific navigation task, the memory required for VPR is predominantly spatial. The marginal gain of maintaining a long-term recurrent history $\mathbf{h}_t$ is outweighed by the optimization stability of a feed-forward architecture. It should be noted that the metric differences between the PPO variants fall within the reported standard deviations and are not statistically distinguishable without formal hypothesis testing, so the selection of standard PPO is based on directional consistency across all metrics and lower computational overhead rather than confirmed superiority. A recurrent architecture would only constitute a necessary addition for missions involving long-duration occlusions or non-linear sensor gaps that exceed the temporal coverage of the frame-stack window.

7.2 Kinematics and Domain Transfer

The superiority of the static SASP model over the dynamic SADP variant implies that physical dynamics act as a natural regularizer for the reward function. The non-holonomic constraints of the unicycle model impose a high temporal cost on turning maneuvers [53]. Since any deviation from the goal-seeking vector requires physical deceleration and increases the risk of reaching the episode horizon $T_{\max}$, the agent is implicitly penalized for infinite reward harvesting. This suggests that complex, non-stationary reward schedules Ψ_t are often redundant when the agent is subjected to realistic kinematic constraints. It is notable that this regularization is a function of the vehicle's actuation limits. If the system were deployed on a platform with near-instantaneous yaw authority, the static visual incentive might overpower the spatial objective.

The rejection of the curriculum learning strategy in Phase I further highlights the risk of representation over-specialization. Although training on syn-

thetic B-splines provided a smooth optimization landscape, it caused the visual encoder to adapt exclusively to clean, high-contrast gradients. When the policy was transferred to the target domain, it suffered a failure due to the fractal complexity and spatial noise of the *OSM LandUse* boundaries. This suggests that for active perception, direct optimization on the target domain is necessary, even if it contradicts the general benefits of curriculum-based ordering [48]. It forces the convolutional layers to learn robust, noise-invariant features of terrain uniqueness from the outset, rather than settling on sterile geometric priors that do not generalize to real-world orthophotos.

7.3 Perceptual Priors and Horizons

The structural ablation of the observation space confirms that reinforcement learning cannot practically deduce semantic uniqueness from raw RGB pixels $\mathbf{I}_t$. The Implicit Static Policy (ISP), which lacked the saliency prior $\mathbf{M}_t$, failed the active perception task and achieved a localization error worse than the naive baseline. Standard convolutional architectures detect geometric structures like edges, but they cannot evaluate global landmark distinctiveness without a high-level semantic prior. By incorporating the SelaVPR++ saliency map $\mathbf{M}_t$ [2] as an explicit input, the model effectively decouples the task of feature identification from the task of trajectory optimization. This allows the RL process to focus on the non-holonomic control problem rather than struggling with the computational complexity of deriving geographic uniqueness from high-dimensional visual data.

While providing $\mathbf{M}_t$ directly to the agent might invite the critique that the observation space reveals the reward function $R_t^{(\text{view})}$, it is critical to distinguish between the reward signal and the state representation. Three properties of the framework jointly bound this potential circularity. First, the ISP variant is trained under an identical reward structure to SASP but its observation contains only the aerial image $\mathbf{I}_t$, the visitation density $\mathbf{D}_t$, and the telemetry vector $\mathbf{g}_t$. Despite receiving the same visual reward signal during optimization, ISP achieves a localization error worse than the naive baseline. If the active perception task were as trivial as maximizing local pixel contrast or heuristics, the ISP variant could have learned a proxy from $\mathbf{I}_t$. The failure of ISP demonstrates that extracting global landmark uniqueness from raw pixels is severely hindered by perceptual aliasing. Thus, $\mathbf{M}_t$ functions as an essential semantic state representation that resolves this aliasing, rather than acting as a trivial shortcut to the reward. Second, within the POMDP formulation, $\mathbf{M}_t$ is a spatially local crop of the pre-depletion saliency field $\mathcal{F}_0$, centred on the agent's current footprint. It communicates the landmark density of the *current* position, not a readout of the global cost landscape or distal rewards. Third, the scalar that actually determines $R_t^{(\text{view})}$ is the *depleted* field $\mathcal{F}_t(\mathbf{p}_t)$, which is never directly observed. The agent can infer that a high-saliency region is

nearby from $\mathbf{M}_t$, but it cannot observe how much of that saliency has already been consumed by prior occupancy. The reward is therefore dependent on the agent's full trajectory history, a quantity absent from any single visual observation. The agent must successfully integrate its spatial memory ($\mathbf{D}_t$) with the static prior ($\mathbf{M}_t$) to avoid redundant exploration. The reliance on a precomputed database for this prior is strictly a computational optimization for the simulator; in a deployed flight stack, the UAV would generate this saliency signal in real-time from the onboard VPR encoder.

A critical aspect of the evaluation framework that warrants discussion is the absolute scale of the Average Localization Error. The recorded mean ALE values consistently range between 45 and 55 m, and the absolute improvement of the proposed SASP architecture over the naive baseline is approximately 2 m. An observer might question the practical significance of such a margin. Two factors contribute to this picture and must be considered jointly. First, the metric is heavily quantized by the geometric setup of the offline VPR database. To simulate a physically realistic reference map, the database was constructed using a 90 m spatial stride for a 180 m camera footprint. This sparse sampling inherently bounds the lower limit of the measurable localization error, regardless of the trajectory quality. Second, the large standard deviations observed across evaluation episodes (typically $\pm$15–20 m) are driven primarily by topographic heterogeneity rather than per-episode measurement noise. Each episode samples a distinct arena from a diverse set of environments that vary substantially in saliency density and baseline VPR difficulty. Arenas dominated by featureless terrain impose an irreducible localization floor that no navigation policy can overcome given the current database resolution. Within this context, the statistical significance of the SASP improvement is confirmed by paired t-tests on the per-episode differences across all 500 held-out arenas, which yield significant results across all three primary metrics ($p \leq 0.022$). The paired design cancels the dominant inter-arena topographic variance, confirming that the 1.78 m mean ALE reduction represents a genuine policy-level effect.

Finally, the generalization failures observed in Phase II define the effective spatial horizon of the learned policy. Since the agent's receptive field is limited to the local observation window, it cannot react to distal salient landmarks that lie beyond its immediate sensor range. This confirms that the proposed system is a reactive local controller rather than a global topological planner [54]. In a production environment, this agent would occupy the mid-level of a navigation hierarchy. A high-level path planner would provide a coarse route through the global cost landscape, while the RL agent would execute the local, feature-aware deviations necessary to maintain VPR stability against sensor noise and GPS drift.

Chapter 8

Conclusion

This thesis addressed the challenge of autonomous UAV navigation in environments susceptible to severe perceptual aliasing. By formulating the navigation task as a POMDP, the objective was to develop an active perception system capable of mitigating state uncertainty. The proposed solution required the agent to deliberately deviate from geometrically optimal paths to acquire visually unique landmarks to maximize the reliability of VPR systems.

The development and evaluation of this system were conducted across two distinct phases. Phase I established a conceptual baseline within a discrete grid-world using semantic abstractions. This phase validated the multi-objective reward formulation. It demonstrated that a reinforcement learning agent could successfully internalize the trade-off between path efficiency and feature acquisition. Furthermore, these experiments exposed the fragility of curriculum learning in this domain. The results proved that direct optimization on noisy data distributions is strictly necessary to prevent representation over-specialization.

Phase II transitioned the system to a high-fidelity continuous aerial simulator operating on authentic orthophotos. This phase introduced second-order unicycle kinematics and a high-dimensional observation space augmented by a pre-computed SelaVPR++ saliency prior. This transition highlighted the limitations of imitation learning for continuous flight control. The DAgger agent failed due to the interaction of informational asymmetry, which was already present in Phase I but manageable in a discrete action space, with the continuous unicycle kinematics, which transform any averaged yaw command into an integrating angular bias. Without an intrinsic value-based correction signal, this drift is irrecoverable and resulted in catastrophic trajectory failure. Consequently, PPO was established as the strictly superior paradigm for non-holonomic active perception.

Furthermore, the structural ablations revealed that deep reinforcement learning agents cannot practically deduce global semantic uniqueness from raw RGB pixels alone. The explicit saliency prior was essential for resolving percep-

tual aliasing. Interestingly, the integration of non-stationary reward schedules proved unnecessary. Within the kinematic envelope of the simulated unicycle UAV, the physical temporal cost associated with turning maneuvers acted as a natural regularizer, effectively preventing the agent from falling into infinite reward-hacking loops. This finding is specific to platforms with limited yaw authority. Near-holonomic vehicles with near-instantaneous heading correction may still require an explicit non-stationary reward schedule.

The final selected architecture, the SASP model, proved highly competent. It successfully maintained a perfect mission reliability rate while executing mathematically efficient detours that closed 23.4% of the perceptual performance gap in the training domain. Zero-shot evaluation on held-out topographies produced statistically confirmed improvements over the naive baseline across all three reported metrics (ALE, Recall@1, and Optimality Score) without any retraining. The modest absolute ALE reduction of 1.78 m reflects the 90 m database stride quantization floor rather than the absence of a real effect.

8.1 Future Work

While the proposed system successfully performs local trajectory refinement, the empirical data highlights a fundamental limitation regarding its spatial horizon. The agent operates as a reactive controller incapable of perceiving distal regions of high VPR confidence because its decisions are strictly bounded by its local observation window.

Future research should focus on integrating this active perception agent into a hierarchical navigation stack. A global topological planner could be utilized to compute a coarse route through a macroscopic cost landscape while the trained RL policy executes the local feature-aware deviations. Additionally, extending the simulation engine to support full 6-DOF multi-rotor kinematics in a 3D environment would bridge the final gap between this simulated proof-of-concept and deployment on physical hardware.

Appendix A

Experimental Tracking and Reproducibility

This appendix provides the technical references and digital logs required to replicate the experiments and verify the performance of the trained agents. All software artifacts and converged model parameters are hosted on public repositories to ensure full transparency.

A.1 Software Implementation

The complete source code for simulation environments, training pipelines, and data acquisition scripts is hosted on the CIIRC GitLab instance. The repository includes setup instructions and configuration files required to replicate both developmental phases.

`https://gitlab.ciirc.cvut.cz/bestavoj/information-driven-uav-navigation`

A.2 Pre-trained Model Parameters

The final weights for the primary agents discussed in the text are deposited on the Hugging Face Model Hub. These repositories contain the model binaries in a format compatible with *Stable Baselines3* alongside standalone inference scripts. The SASP model specifically incorporates the SelaVPR++ feature extraction backbone validated during the Phase II evaluation.

Agent Identification	Hugging Face Repository URL
HASP (Phase I)	`https://huggingface.co/bestak/uav-navigation-hasp`
SASP (Phase II)	`https://huggingface.co/bestak/uav-navigation-sasp`

Table A.1 Summary of pre-trained agents deposited on the Hugging Face Hub.

A.3 Weights & Biases Training Histories

All optimization phases and hyperparameter sweeps were logged using the Weights & Biases (WandB) platform. The project history includes hardware utilization and training telemetry accessible via the project dashboard:

`https://wandb.ai/bestak-team/drone-navigation`

Phase I Experiment Logs

The records in Table A.2 represent the final validated experimental series for the discrete semantic grid. The primary metric refers to the converged value at the conclusion of the training budget.

Experiment Label	WandB Group Name (Hyperlink)	Runs	Primary Metric
1.1 HASP Sweep	landuse_sweep_6	35	Opt: 0.775
1.2 Curriculum (Full)	exp1.2-ppo-cur0123_5	5	Succ: 83.8%
1.2 Curriculum (2-Stage)	exp1.2-ppo-cur03_5	5	Succ: 76.6%
1.2 Direct Training	exp1.2-ppo-cur3_5	5	Succ: 98.9%
1.3 PPO	exp1.3-ppo_2	5	Opt: 0.779
1.3 DQN	exp1.3-dqn_2	5	Opt: 0.820
1.3 LSTM-PPO	exp1.3-lstm-ppo_2	5	Opt: 0.767
1.3 DAgger	exp1.3-dagger_6	5	Opt: 0.698
1.3 Hybrid (FT)	exp1.3-ppo-il-ft_2	5	Opt: 0.784

Table A.2 WandB logging summary for Phase I experiments.

Phase II Experiment Logs

The records in Table A.3 represent the validated runs for the continuous aerial environment. During the architectural search, approximately 2,100 additional interactions were logged which remain available in the project history but are omitted here for brevity.

Experiment Label	WandB Group Name (Hyperlink)	Runs	Primary Metric
2.1 SASP Sweep	exp2.1-sweep-ppo-saliency_1	30	Opt: 0.634
2.1 SADP Sweep	exp2.1-sweep-ppo-saliency_dynamic_1	30	-
2.1 ISP Sweep	exp2.1-sweep-ppo_9	30	-
2.2 Ablation ISP	exp2.2-aerial-ppo-ablation-isp_7	5	ALE: 53.50 m
2.2 Ablation SADP	exp2.2-aerial-ppo-ablation-sadp_5	5	ALE: 52.28 m
2.2 Ablation SASP	exp2.2-aerial-ppo-ablation-sasp_5	5	ALE: 51.74 m
2.3 PPO	exp2.2-aerial-ppo-ablation-sasp_5	5	Succ: 99.4%
2.3 LSTM-PPO	exp2.3-aerial-lstm-ppo_4	5	Succ: 99.7%
2.3 DAgger	exp2.3-aerial-dagger_2	5	Succ: 53.0%

Table A.3 WandB logging summary for Phase II experiments.

Bibliography

1. SAMY, M.; AMER, K.; SHAKER, M.; ELHELW, M. *Drone Path-Following in GPS-Denied Environments using Convolutional Networks*. 2019. Available from arXiv: 1905.01658 [cs.CV].
2. LU, Feng; JIN, Tong; LAN, Xiangyuan; ZHANG, Lijun; LIU, Yunpeng; WANG, Yaowei; YUAN, Chun. SelaVPR++: Towards Seamless Adaptation of Foundation Models for Efficient Place Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2026, vol. 48, no. 3, pp. 2731–2748. Available from DOI: 10.1109/TPAMI.2025.3629287.
3. SUTTON, Richard S; BARTO, Andrew G. *Reinforcement Learning: An Introduction*. 2nd ed. Cambridge, MA: The MIT Press, 2018. Available also from: http://incompleteideas.net/book/the-book-2nd.html.
4. BELLMAN, Richard. A Markovian Decision Process. *Indiana University Mathematics Journal*. 1957, vol. 6, no. 4, pp. 679–684.
5. THOMAS, Philip S.; OKAL, Billy. *A Notation for Markov Decision Processes*. 2016. Available from arXiv: 1512.09075 [cs.AI].
6. STRAKA, Milan. *Introduction to Reinforcement Learning*. Institute of Formal and Applied Linguistics, Charles University, 2026. Available also from: https://ufal.mff.cuni.cz/~straka/courses/npfl139/2526/slides.pdf/npfl139-2526-01.pdf.
7. BERTSEKAS, D.P. *Dynamic Programming and Optimal Control: Approximate dynamic programming. Volume 2*. Athena Scientific, 2012. ISBN 9781886529083. Available also from: https://books.google.cz/books?id=0JqfswEACAAJ.
8. CHRISMAN, Lonnie. Reinforcement Learning with Perceptual Aliasing: The Perceptual Distinctions Approach. *Proc. Natl. Conf. Artif. Intell.* 1994.

9. CROOK, Paul A.; HAYES, Gillian. Learning in a State of Confusion: Perceptual Aliasing in Grid World Navigation. In: *Proceedings of Towards Intelligent Mobile Robots (TIMR 2003)*. Bristol, UK, 2003. Informatics Research Report EDI-INF-RR-0176, School of Informatics, University of Edinburgh.

10. KAELBLING, Leslie Pack; LITTMAN, Michael L.; CASSANDRA, Anthony R. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*. 1998, vol. 101, no. 1, pp. 99–134. ISSN 0004-3702. Available from DOI: https://doi.org/10.1016/S0004-3702(98)00023-X.

11. HAUSKNECHT, Matthew; STONE, Peter. *Deep Recurrent Q-Learning for Partially Observable MDPs*. 2017. Available from arXiv: 1507.06527 [cs.LG].

12. MNIH, Volodymyr; KAVUKCUOGLU, Koray; SILVER, David; GRAVES, Alex; ANTONOGLOU, Ioannis; WIERSTRA, Daan; RIEDMILLER, Martin. *Playing Atari with Deep Reinforcement Learning*. 2013. Available from arXiv: 1312.5602 [cs.LG].

13. SCHULMAN, John; WOLSKI, Filip; DHARIWAL, Prafulla; RADFORD, Alec; KLIMOV, Oleg. *Proximal Policy Optimization Algorithms*. 2017. Available from arXiv: 1707.06347 [cs.LG].

14. LIN, Long-Ji. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*. 1992, vol. 8, no. 3, pp. 293–321. ISBN 1573-0565. Available from DOI: 10.1007/BF00992699.

15. SCHULMAN, John; MORITZ, Philipp; LEVINE, Sergey; JORDAN, Michael I.; ABBEEL, P. High-Dimensional Continuous Control Using Generalized Advantage Estimation. *CoRR*. 2015, vol. abs/1506.02438. Available also from: https://api.semanticscholar.org/CorpusID:3075448.

16. SCHULMAN, John; LEVINE, Sergey; ABBEEL, Pieter; JORDAN, Michael; MORITZ, Philipp. Trust Region Policy Optimization. In: BACH, Francis; BLEI, David (eds.). *Proceedings of the 32nd International Conference on Machine Learning*. Lille, France: PMLR, 2015, vol. 37, pp. 1889–1897. Proceedings of Machine Learning Research. Available also from: https://proceedings.mlr.press/v37/schulman15.html.

17. WILLIAMS, RONALD J.; PENG, JING. Function Optimization using Connectionist Reinforcement Learning Algorithms. *Connection Science*. 1991, vol. 3, no. 3, pp. 241–268. Available from DOI: 10.1080/09540099108946587.

18. MNIH, Volodymyr; BADIA, Adria Puigdomenech; MIRZA, Mehdi; GRAVES, Alex; LILLICRAP, Timothy; HARLEY, Tim; SILVER, David; KAVUKCUOGLU, Koray. Asynchronous Methods for Deep Reinforcement Learning. In: BALCAN, Maria Florina; WEINBERGER, Kilian Q. (eds.). *Proceedings of The 33rd International Conference on Machine Learning*. New York, New York, USA: PMLR, 2016, vol. 48, pp. 1928–1937. Proceedings of Machine Learning Research. Available also from: https://proceedings.mlr.press/v48/mniha16.html.

19. GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

20. ELMAN, Jeffrey L. Finding structure in time. *Cognitive Science*. 1990, vol. 14, no. 2, pp. 179–211. ISSN 0364-0213. Available from DOI: https://doi.org/10.1016/0364-0213(90)90002-E.

21. WERBOS, P.J. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*. 1990, vol. 78, no. 10, pp. 1550–1560. Available from DOI: 10.1109/5.58337.

22. HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. Long Short-Term Memory. *Neural Computation*. 1997, vol. 9, no. 8, pp. 1735–1780. Available from DOI: 10.1162/neco.1997.9.8.1735.

23. GREFF, Klaus; SRIVASTAVA, Rupesh; KOUTNÍK, Jan; STEUNEBRINK, Bas; SCHMIDHUBER, Jürgen. LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems*. 2015, vol. 28. Available from DOI: 10.1109/TNNLS.2016.2582924.

24. PLEINES, Marco; PALLASCH, Matthias; ZIMMER, Fränk; PREUSS, Mike. Generalization, Mayhems and Limits in Recurrent Proximal Policy Optimization. *ArXiv*. 2022, vol. abs/2205.11104. Available also from: https://api.semanticscholar.org/CorpusID:248987115.

25. YUAN, Yinlong; YU, Zhu Liang; GU, Zhenghui; DENG, Xiaoyan; LI, Yuanqing. A novel multi-step reinforcement learning method for solving reward hacking. *Applied Intelligence*. 2019, vol. 49, no. 8, pp. 2874–2888. ISBN 1573-7497. Available from DOI: 10.1007/s10489-019-01417-4.

26. OSA, Takayuki; PAJARINEN, Joni; NEUMANN, Gerhard; BAGNELL, J. Andrew; ABBEEL, Pieter; PETERS, Jan. An Algorithmic Perspective on Imitation Learning. *Foundations and Trends® in Robotics*. 2018, vol. 7, no. 1–2, pp. 1–179. ISSN 1935-8261. Available from DOI: 10.1561/2300000053.

27. ZHENG, Boyuan; VERMA, Sunny; ZHOU, Jianlong; TSANG, Ivor W.; CHEN, Fang. Imitation Learning: Progress, Taxonomies and Challenges. *IEEE Transactions on Neural Networks and Learning Systems*. 2024, vol. 35, no. 5, pp. 6322–6337. Available from DOI: 10.1109/TNNLS.2022.3213246.

28. HO, Jonathan; ERMON, Stefano. *Generative Adversarial Imitation Learning*. 2016. Available from arXiv: 1606.03476 [cs.LG].

29. ROSS, Stephane; GORDON, Geoffrey J.; BAGNELL, J. Andrew. *A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning*. 2011. Available from arXiv: 1011.0686 [cs.LG].

30. WARRINGTON, Andrew; LAVINGTON, Jonathan Wilder; SCIBIOR, A.; SCHMIDT, Mark W.; WOOD, Frank D. Robust Asymmetric Learning in POMDPs. In: *International Conference on Machine Learning*. 2020. Available also from: https://api.semanticscholar.org/CorpusID:229923742.

31. BAJCSY, Ruzena. Active perception. *Proc. IEEE*. 1988, vol. 76, pp. 966–1005. Available also from: https://api.semanticscholar.org/CorpusID:12977798.

32. LECUN, Yann. Generalization and Network Design Strategies. In: PFEIFER, R.; SCHRETER, Z.; FOGELMAN, F.; STEELS, L. (eds.). *Connectionism in Perspective*. Zurich, Switzerland: Elsevier, 1989. an extended version was published as a technical report of the University of Toronto.

33. LECUN, Yann; HAFFNER, Patrick; RACHMAD, Yoesoep; BOTTOU, Leon. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*. 1998, vol. 86, pp. 2278–2324. Available from DOI: 10.1109/5.726791.

34. VASWANI, Ashish; SHAZEER, Noam; PARMAR, Niki; USZKOREIT, Jakob; JONES, Llion; GOMEZ, Aidan N; KAISER, Łukasz; POLOSUKHIN, Illia. Attention is All you Need. In: GUYON, I.; LUXBURG, U. Von; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S.; GARNETT, R. (eds.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017, vol. 30. Available also from: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

35. DOSOVITSKIY, Alexey; BEYER, Lucas; KOLESNIKOV, Alexander; WEISSENBORN, Dirk; ZHAI, Xiaohua; UNTERTHINER, Thomas; DEHGHANI, Mostafa; MINDERER, Matthias; HEIGOLD, Georg; GELLY, Sylvain; USZKOREIT, Jakob; HOULSBY, Neil. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. Available from arXiv: 2010.11929 [cs.CV].

36. OQUAB, Maxime; DARCET, Timothée; MOUTAKANNI, Théo; VO, Huy; SZAFRANIEC, Marc; KHALIDOV, Vasil; FERNANDEZ, Pierre; HAZIZA, Daniel; MASSA, Francisco; EL-NOUBY, Alaaeldin; ASSRAN, Mahmoud; BALLAS, Nicolas; GALUBA, Wojciech; HOWES, Russell; HUANG, Po-Yao; LI, Shang-Wen; MISRA, Ishan; RABBAT, Michael; SHARMA, Vasu; SYNNAEVE, Gabriel; XU, Hu; JEGOU,

Hervé; MAIRAL, Julien; LABATUT, Patrick; JOULIN, Armand; BOJANOWSKI, Piotr. *DINOv2: Learning Robust Visual Features without Supervision*. 2024. Available from arXiv: 2304.07193 [cs.CV].

37. XIONG, Ruibin; YANG, Yunchang; HE, Di; ZHENG, Kai; ZHENG, Shuxin; XING, Chen; ZHANG, Huishuai; LAN, Yanyan; WANG, Liwei; LIU, Tie-Yan. On layer normalization in the transformer architecture. In: *Proceedings of the 37th International Conference on Machine Learning*. JMLR.org, 2020. ICML'20.

38. MITCHELL, Tom M. *The Need for Biases in Learning Generalizations*. New Brunswick, NJ, USA, 1980-05. Tech. rep., CBM-TR-117. Rutgers University, Computer Science Department.

39. LOWRY, Stephanie; SÜNDERHAUF, Niko; NEWMAN, Paul; LEONARD, John J.; COX, David; CORKE, Peter; MILFORD, Michael J. Visual Place Recognition: A Survey. *IEEE Transactions on Robotics*. 2016, vol. 32, no. 1, pp. 1–19. Available from DOI: 10.1109/TRO.2015.2496823.

40. ARANDJELOVIĆ, Relja; GRONAT, Petr; TORII, Akihiko; PAJDLA, Tomas; SIVIC, Josef. NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2018, vol. 40, no. 6, pp. 1437–1451. Available from DOI: 10.1109/TPAMI.2017.2711011.

41. CAO, Bingyi; ARAUJO, André; SIM, Jack. Unifying Deep Local and Global Features for Image Search. In: 2020, pp. 726–743. ISBN 978-3-030-58564-8. Available from DOI: 10.1007/978-3-030-58565-5_43.

42. GONZALEZ, R.C.; WOODS, R.E. *Digital Image Processing*. Prentice Hall, 2008. ISBN 9780131687288. Available also from: https://books.google.cz/books?id=8uGOnjRGEzoC.

43. KEETHA, Nikhil; MISHRA, Avneesh; KARHADE, Jay; JATAVALLABHULA, Krishna Murthy; SCHERER, Sebastian; KRISHNA, Madhava; GARG, Sourav. Anyloc: Towards universal visual place recognition. *IEEE Robotics and Automation Letters*. 2023.

44. LU, Feng; ZHANG, Lijun; LAN, Xiangyuan; DONG, Shuting; WANG, Yaowei; YUAN, Chun. *Towards Seamless Adaptation of Pre-trained Models for Visual Place Recognition*. 2024. Available from arXiv: 2402.14505 [cs.CV].

45. RADENOVIĆ, Filip; TOLIAS, Giorgos; CHUM, Ondřej. Fine-tuning CNN Image Retrieval with No Human Annotation. *arXiv e-prints*. 2017, arXiv:1711.02512. Available from DOI: 10.48550/arXiv.1711.02512.

46. SCHULTZ, Michael; LI, Hao; WU, Zhaoyhan; WIELL, Daniel; AUER, Michael; ZIPF, Alexander. *OpenStreetMap land use for Europe "Research Data"* [heiDATA repository]. 2024. Version V2 [visited on 2025-12-02]. Available from DOI: 10.11588/DATA/IUTCDN.

47. AKIBA, Takuya; SANO, Shotaro; YANASE, Toshihiko; OHTA, Takeru; KOYAMA, Masanori. *Optuna: A Next-generation Hyperparameter Optimization Framework*. 2019. Available from arXiv: 1907.10902 [cs.LG].

48. BENGIO, Yoshua; LOURADOUR, Jérôme; COLLOBERT, Ronan; WESTON, Jason. Curriculum learning. In: *Proceedings of the 26th Annual International Conference on Machine Learning*. Montreal, Quebec, Canada: Association for Computing Machinery, 2009, pp. 41–48. ICML '09. ISBN 9781605585161. Available from DOI: 10.1145/1553374.1553380.

49. WANG, Xianglong; QIU, Lukuan; HU, Zhaozheng; LI, Yin; WANG, Ganlong. A Review for Visual Place Recognition: From Hand-Crafted and Deep Learning Perspective. In: *2025 8th International Conference on Transportation Information and Safety (ICTIS)*. 2025, pp. 1454–1463. Available from DOI: 10.1109/ICTIS68762.2025.11214850.

50. ČESKÝ ÚŘAD ZEMĚMĚŘICKÝ A KATASTRÁLNÍ. *Ortofoto České republiky [Dataset]* [Geoportál ČÚZK]. 2025. Available also from: https://geoportal.cuzk.cz/Default.aspx?mode=TextMeta&text=ortofoto_info. Data retrieved via Open Data portal (openzu.cuzk.cz). Accessed: 2026-04-01.

51. IOGP GEODETIC PARAMETER DATASET. *EPSG:5514: S-JTSK / Krovak East North*. 2025. Available also from: https://epsg.org/crs_5514/S-JTSK-Krovak-East-North.html. Geodetic coordinate system used for Cartesian spatial operations in the simulator. Accessed: 2026-04-01.

52. MOSKALENKO, Ivan; KORNILOVA, Anastasiia; FERRER, Gonzalo. Visual place recognition for aerial imagery: A survey. *Robotics and Autonomous Systems*. 2025, vol. 183, p. 104837. ISSN 0921-8890. Available from DOI: 10.1016/j.robot.2024.104837.

53. ADZKIYA, Dieky; MUFID, Muhammad; SAPUTRI, Febrianti; ABATE, Alessandro. Control design of discrete-time unicycle model using satisfiability modulo theory. *Systems Science & Control Engineering*. 2024, vol. 12. Available from DOI: 10.1080/21642583.2024.2316166.

54. LAVALLE, Steven. *Planning Algorithms*. 2006. ISBN 9780521862059. Available from DOI: 10.1017/CBO9780511546877.

55. COULTER, R. Craig. *Implementation of the Pure Pursuit Path Tracking Algorithm*. Pittsburgh, PA, 1992-01. Tech. rep., CMU-RI-TR-92-01.

Enclosed Media Contents

```
/
├── README.md ..... brief description of the media root
├── src ..... implementation of the simulation engine and RL/IL pipelines
│   └── README.md ..... installation and usage instructions
└── text ..... thesis text
    ├── latex ..... source code of the thesis in LaTeX format
    └── thesis.pdf ..... final version of the thesis in PDF format
```