

Article

Hybrid Vision Navigation with Hierarchical VLM–LLM Decision Making

Rihem Farkh ¹, Ghislain Oudinet ², Mohamed Adjou ³, Alaeddine Moussa ⁴ and Yasser Fouad ^{5,*}¹ KLaIM, LabISEN, ISEN Méditerranée, 83000 Toulon, France; rihem.farkh@yncrea.fr² VISION-AD, LabISEN, ISEN Méditerranée, 83000 Toulon, France³ KLaIM, LabISEN, ISEN Ouest, 29200 Brest, France⁴ Laboratoire LIS UMR CNRS 7020, Aix Marseille Université, 13397 Marseille, France⁵ Department of Applied Mechanical Engineering, College of Applied Engineering, Muzahimiyah Branch, King Saud University, Riyadh 11421, Saudi Arabia

* Correspondence: yfouad@ksu.edu.sa

Abstract

This paper presents a hybrid navigation architecture for mapless navigation based on monocular vision. The system combines perception-driven affordance control with event-triggered semantic reasoning within a unified decision framework. Navigation behavior is governed by interpretable perceptual signals and a vision-derived progress proxy that enables self-monitoring. A reactive control regime ensures real-time safety, while a semantic reasoning module is activated only under persistent navigation difficulty to provide structured guidance. Experimental results in simulation and real-world deployment demonstrate improved success rate, safety, and efficiency compared to reactive and continuously active semantic baselines, while maintaining real-time performance on embedded hardware.

Keywords: vision-only navigation; mapless autonomy; affordance-based control; event-triggered semantic reasoning; vision–language models; large language models; autonomous robots; embedded robotic systems

1. Introduction

Autonomous indoor navigation remains a fundamental challenge in robotics, particularly in environments where accurate maps are unavailable, sensor data is noisy, and obstacles are dynamic or partially occluded [1,2]. Classical navigation systems typically rely either on precise geometric maps and localization, or on purely reactive controllers driven by low-level perception. While effective in structured settings, such approaches often degrade in cluttered or unfamiliar environments, producing oscillatory behavior, repeated collisions, or deadlocks near obstacles [3].

Recent learning-based methods, including deep reinforcement learning and imitation learning, aim to replace hand-designed controllers with end-to-end policies [4]. Although promising, these systems require large amounts of training data, are difficult to interpret, and often generalize poorly to unseen environments [5]. More recently, large language models (LLMs) and vision–language models (VLMs) have been explored as high-level decision-making components for robots [6]. While these models offer strong semantic reasoning capabilities, they are computationally expensive, can hallucinate unsafe actions, and lack the fast reflexes required for real-time navigation and collision avoidance [7].



Academic Editor: Yiyang Han

Received: 5 March 2026

Revised: 9 April 2026

Accepted: 12 April 2026

Published: 14 April 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

Biological navigation systems demonstrate the importance of adaptive behavioral strategy selection. Rather than relying on a single monolithic control policy, biological systems dynamically adjust behavior according to environmental context and perceived progress, for example by slowing down under uncertainty, recovering from obstacles, or executing rapid evasive responses when danger is detected [8,9]. Inspired by these principles, modern robotic systems increasingly incorporate supervisory control layers that monitor performance and adapt behavior accordingly [10].

We propose a hybrid navigation architecture integrating perception-driven control with event-triggered semantic reasoning under a unified supervisory framework (detailed in Section 5). The system combines reactive navigation with conditional semantic guidance, enabling adaptive behavior while preserving real-time performance.

We evaluate the proposed architecture in both simulated indoor environments and real-world robotic deployment. Experimental results demonstrate improved navigation success, reduced collision frequency, and lower semantic reasoning cost compared to reactive baselines and continuously active language-guided controllers, while maintaining real-time performance on embedded hardware platforms. These results suggest that adaptive multi-regime navigation combining perception-driven control, supervisory event detection, and conditionally activated semantic reasoning provides a practical and scalable approach for vision-only autonomous navigation.

The main contributions of this work can be summarized as follows:

- A vision-only, mapless navigation architecture operating without depth sensing, localization, or metric goal information.
- A unified affordance-based framework integrating perception-driven control, supervisory modes, episodic memory, and semantic reasoning.
- An event-triggered hierarchical VLM–LLM navigation module that avoids continuous semantic inference.
- A visual progress proxy enabling self-monitoring without requiring spatial representations.
- A dual-regime adaptive navigation mechanism balancing real-time safety and high-level reasoning.

The remainder of the paper is organized as follows. Section 2 reviews related work. Sections 3 and 4 describe perceptual invariants and the visual progress proxy. Section 5 presents the system architecture. Sections 6–9 detail affordance control, supervisory modes, semantic navigation, and adaptive regime selection. Section 10 introduces episodic memory. Section 12 presents experiments and results, followed by conclusions.

2. Related Work

Autonomous robot navigation has been studied extensively, yielding approaches that can be broadly categorized as reactive control, affordance-based methods, learning-based systems, and, more recently, language-driven navigation. While each paradigm addresses specific aspects of the navigation problem, no paradigm alone provides a robust solution for mapless navigation in cluttered, unknown indoor environments.

2.1. Reactive and Invariant-Based Navigation

Early navigation systems relied on reactive control, generating actions directly from sensor measurements without constructing explicit world models. Classical approaches such as Braitenberg vehicles, potential fields, and subsumption architectures demonstrated that simple perception–action couplings can produce effective obstacle avoidance and goal-seeking behaviors in constrained settings. However, their lack of global context makes them vulnerable to oscillations and deadlocks in complex environments.

Biologically inspired extensions introduced perception-driven signals—such as looming-based visual expansion cues, motion asymmetry, and ground-plane continuity—to anticipate collisions and identify traversable space without explicit depth estimation. These methods offer fast, interpretable reflexes and low computational overhead, making them attractive for real-time control. Nevertheless, their decision-making remains inherently local, limiting their ability to recover from persistent failure cases or adapt behavior over longer time horizons [11–13].

2.2. Affordance-Based Action Selection

Affordance-based navigation formulates control as the evaluation of a small set of candidate actions based on perceptual cues, rather than explicit path planning in a metric map. This paradigm has been adopted in both classical vision-based systems and modern neural approaches that infer action likelihoods from images [14].

By constraining the action space, affordance fields enable efficient reactive decision-making. However, without higher-level supervision, affordance-based controllers remain susceptible to repetitive or counterproductive behaviors. In particular, they lack mechanisms to detect stagnation, negative progress, or repeated failure, which can lead to long-term divergence despite locally reasonable action choices [15].

2.3. Learning-Based Navigation

End-to-end learning approaches, including deep reinforcement learning and imitation learning, aim to directly map raw sensory inputs to control actions. While these methods can achieve strong performance in training environments, they typically require extensive data collection and careful reward shaping, and often generalize poorly to unseen scenes. Their internal representations are also difficult to interpret, complicating failure analysis and safety guarantees [16].

Hybrid learning-based systems attempt to combine learned components with classical control. However, many such systems still rely on a single policy or fixed control structure throughout an episode, making them vulnerable to catastrophic failures when encountering novel or ambiguous situations that fall outside the training distribution [17].

It is important to note that most reinforcement learning (RL) navigation approaches rely on reward signals derived from goal distance, privileged simulator state, or localization information. In contrast, the proposed system operates under strict vision-only constraints. While RGB-only RL is theoretically possible, designing a fair comparison under identical constraints would require a substantially different training setup and reward formulation. For this reason, RL-based methods are discussed qualitatively rather than included as direct experimental baselines.

2.4. Vision–Language Models for Robotics

Recent advances in vision–language models (VLMs) and large language models (LLMs) have motivated their use as high-level planners or decision-making modules in robotic systems. These models provide powerful semantic reasoning capabilities and can generate structured action suggestions from visual input. However, their high computational cost, latency, and susceptibility to hallucinated or unsafe outputs make them unsuitable for continuous low-level navigation control [18].

To address these limitations, several works employ language models only for high-level guidance while delegating execution to traditional controllers. Despite this separation, most existing approaches lack a principled mechanism for determining when language-based reasoning should be invoked or how its outputs should be integrated safely and efficiently into real-time control loops [19].

2.5. Hierarchical Semantic Navigation and Planning

Recent work has begun exploring hierarchical semantic navigation approaches that combine visual scene understanding with high-level reasoning to generate intermediate navigation targets such as semantic waypoints or landmark-based goals. These methods aim to bridge the gap between purely reactive navigation and full metric mapping by leveraging semantic structure in visual environments.

While promising, many existing semantic navigation systems rely on continuous semantic reasoning or assume access to structured environment representations, which can limit real-time performance and deployment on resource-constrained platforms. In addition, few systems provide mechanisms for dynamically switching between reactive control and semantic planning based on task difficulty or environmental structure.

2.6. Supervisory Control and Adaptive Strategy Selection

Layered control and subsumption-based architectures introduced the concept of behavior arbitration and multi-level decision-making. More recent work in adaptive control and meta-learning emphasizes monitoring performance and switching strategies when failures are detected. However, these concepts have rarely been applied in a unified manner to vision-based, mapless mobile robot navigation systems that integrate fast perception-driven control, hierarchical semantic reasoning, and experience-based adaptation within a single operational framework [20,21].

3. Perceptual Invariants

To enable fast, interpretable navigation without maps, localization, or depth sensors, the proposed system relies on a small set of handcrafted perceptual invariants extracted directly from consecutive monocular RGB images. These invariants are inspired by biological vision and ecological psychology, where stable visual cues guide action without requiring explicit 3D reconstruction. Rather than estimating geometry, the signals act as lightweight visual proxies that summarize local risk, motion, and environmental structure.

Let I_{t-1} and I_t denote two consecutive camera frames. From these, four scalar signals are computed to form a compact perceptual representation.

3.1. Looming-Inspired Brightness Expansion

Classical time-to-contact (TTC) theory estimates the imminence of collision based on the rate of image expansion [22]. Since explicit depth and dense optical flow are unavailable in our setting, we instead compute a brightness-based expansion heuristic inspired by looming perception.

Specifically, we measure the relative change in the number of bright pixels within a central image region:

$$\tau \approx \frac{S_t}{S_t - S_{t-1}} \quad (1)$$

where S_t denotes the number of pixels above a fixed brightness threshold in the central region of the image at time t .

A small value of τ corresponds to a rapid increase in bright pixels, which often correlates with forward motion toward nearby surfaces and thus elevated collision risk. When τ falls below a predefined threshold, the reflex layer suppresses forward motion and enforces a STOP or TURN action. This signal is a computationally efficient heuristic and does not represent a physically grounded estimate of time-to-collision.

3.2. Left–Right Motion Asymmetry

To capture lateral visual motion, we compute a frame-difference-based asymmetry signal between the left and right halves of the image:

$$\phi = \Delta_L - \Delta_R \quad (2)$$

where Δ_L and Δ_R denote the mean absolute pixel differences between I_{t-1} and I_t computed over the left and right image halves, respectively.

This signal does not represent true optical flow, but rather a motion energy asymmetry reflecting relative visual change across the image. A positive value indicates stronger motion on the left side, suggesting nearby structure or obstacles and a preference to turn right, and vice versa [23].

3.3. Ground Edge Density

To detect obstacles near the floor, we analyse the lower region of the image. Horizontal and vertical intensity gradients are computed, and strong edges are counted:

$$g = \frac{\text{number of strong edges}}{\text{number of pixels in bottom band}} \quad (3)$$

where edges are detected using a fixed gradient-magnitude threshold.

A high value of g indicates dense edge structure near the ground plane, which often corresponds to walls, furniture, or other obstacles that impede forward motion [24].

3.4. Image Entropy

Visual entropy measures the overall texture and complexity of the scene:

$$H = -\sum_i p_i \log_2 p_i \quad (4)$$

where p_i is the empirical probability of gray-level intensity i in the image.

High entropy values indicate visually complex or highly textured scenes, which are often associated with cluttered or ambiguous regions where navigation decisions become less reliable [25].

4. Visual Progress Proxy

In the absence of metric localization, maps, or goal-distance-based control, the robot requires an internal mechanism to assess whether its recent actions are improving the navigational situation or leading toward increased difficulty. To this end, we introduce a visual progress proxy derived exclusively from the perception-driven signals described in Section 3. This proxy provides a perception-driven estimate of navigational improvement using only monocular RGB observations and is used for supervisory monitoring, explicit event detection, and adaptive regime selection.

At each time step t , the agent computes the perceptual signal vector:

$$\mathbf{z}_t = (\tau_t, \phi_t, g_t, H_t) \quad (5)$$

4.1. Perceptual Risk Estimate

We define an instantaneous perceptual risk score as a weighted sum of normalized invariant signals [26–28]:

$$R_t = w_\tau \hat{\tau}_t + w_\phi |\hat{\phi}_t| + w_g \hat{g}_t + w_H \hat{H}_t \quad (6)$$

where $\hat{\cdot}$ denotes normalization computed over a sliding window of W frames and w_τ, w_ϕ, w_g, w_H are fixed scalar weights.

Each weight controls the relative contribution of a specific perceptual cue: the looming term captures collision imminence, motion asymmetry encodes lateral obstacle distribution, ground-edge density reflects traversability constraints, and entropy represents visual uncertainty. The weights were empirically tuned to balance safety and navigation progress, and all signals are normalized over a sliding window to ensure consistent scaling across environments.

This risk measure does not estimate geometric distance, time-to-collision, or goal proximity. Instead, it acts as a compact, vision-based indicator of how visually cluttered, uncertain, or hazardous the current situation appears.

4.2. Visual Progress Definition

Visual progress is defined as the temporal decrease in perceptual risk, gated by observed inter-frame motion [29,30]:

$$P_t = (R_{t-1} - R_t) \mathbb{I}(\|\Delta I_t\| > \epsilon) \quad (7)$$

where ΔI_t denotes the mean absolute pixel difference between consecutive frames, ϵ is a small motion threshold, and $\mathbb{I}(\cdot)$ is an indicator function that suppresses false progress estimates under near-static observations.

A positive value of P_t indicates that the agent is moving toward visually safer or less cluttered regions, while near-zero or negative values indicate stagnation or increasing local perceptual risk. This signal is used exclusively for supervisory monitoring and event detection, such as identifying stagnation, triggering supervisory mode transitions (e.g., SLOW, RECOVER, or ESCAPE), and activating semantic navigation when persistent failure is detected (Figure 1).

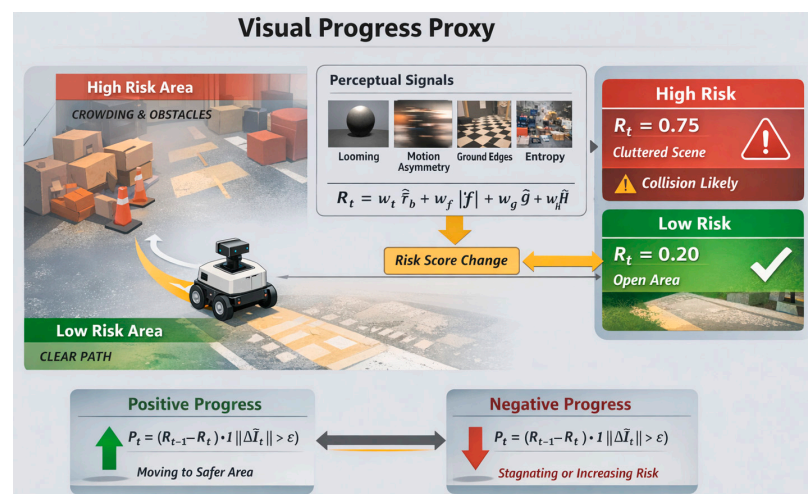


Figure 1. Visual Progress.

4.3. Vision-Based Goal Representation

In order to maintain consistency with the vision-only constraints of the proposed architecture, navigation goals are represented through visual similarity rather than metric coordinates. The controller does not have access to spatial goal positions, distance-to-goal, or global pose information. Instead, each task is defined by a goal visual representation, either as a reference image or as a semantic scene description interpreted through the VLM. At runtime, the system computes a goal similarity score directly from the current RGB observation, enabling goal-directed behavior without localization or map construction.

Ground-truth spatial goal regions, when used in simulation, serve exclusively as external evaluation criteria and are never available to the controller during execution.

5. System Architecture Overview

5.1. Overview

The proposed navigation system follows a layered architecture in which perception-driven affordances provide the primary control mechanism, while supervisory processes monitor navigation progress and adapt behavior when local perception becomes insufficient.

The system is organized around a shared perception–control backbone supporting two complementary regimes: a reactive regime for real-time navigation and a semantic regime providing higher-level guidance when required. An adaptive supervisory mechanism regulates the activation of semantic reasoning.

All components interact through a unified affordance-based decision process, ensuring that action selection remains consistent and safety-driven across both regimes.

5.2. Architectural Components

The navigation architecture is composed of tightly integrated modules operating at different temporal and functional levels. Together, these components enable perception-grounded decision making, adaptive strategy selection, event-triggered semantic reasoning, and robust safety enforcement. Table 1 summarizes the principal architectural modules and their roles.

Table 1. Main architectural components of the proposed system.

Module	Role in the System
Perception-Driven Signals	Extract compact vision-based signals describing local risk, structure, clutter, and uncertainty (Section 3)
Visual Progress Proxy	Estimates perception-driven navigation progress used for monitoring and event detection (Section 4)
Event Detection Module	Identifies navigation-critical events (collision risk, stagnation, deterioration, uncertainty, immediate danger) that drive supervisory decisions
Affordance Field	Computes action preferences from perceptual signals (Section 6)
Episodic Memory	Biases affordance scores based on past perceptual outcomes
Supervisory Mode Controller	Selects one of four supervisory modes (GO, SLOW, RECOVER, ESCAPE) based on detected events (Section 7)
Adaptive Regime Selection	Chooses between reactive and semantic navigation regimes (Section 9)
Hierarchical Semantic Planning (Conditional)	Provides semantic waypoint guidance when activated (Section 8)
Reflex Safety Gating Layer	Final action-selection gate that vetoes unsafe actions after all affordances and biases are combined
Graceful Degradation Mechanism	Progressively falls back from semantic planning to pure reactive control when failures occur

These components operate at multiple time scales, ranging from fast perception and affordance evaluation to slower supervisory monitoring, memory retrieval, and conditional semantic reasoning.

5.3. Control Loop

At each time step t , the RGB input is converted into a perceptual vector $z_t = [\tau_t, \phi_t, g_t, H_t]$. From this representation, a visual progress estimate is computed and passed to an event detection stage that explicitly identifies navigation-critical conditions. Detected events drive supervisory mode selection and adaptive regime switching.

Reactive affordance evaluation remains continuously active and produces a base affordance field over the discrete action set. Episodic memory and supervisory mode biasing reshape this field without directly selecting actions. When the semantic regime is active, visual waypoint guidance extracted from VLM scene understanding and LLM planning is blended as a soft bias into the same affordance representation rather than forming an independent controller.

After all contributions have been combined, the reflex safety gating layer performs the final action-selection check, enforcing hard safety vetoes on any action that violates immediate perceptual constraints. A single unified action selection step then chooses the highest-scoring safe action. This design ensures that semantic reasoning, memory, and supervisory modes influence behavior only through affordance modulation, while perception-driven safety remains dominant.

The Graceful Degradation Mechanism maintains robustness under repeated failures by progressively falling back from full semantic planning to reactive navigation with intensified recovery behavior. Semantic planning is invoked conditionally and persists across multiple control cycles, avoiding unnecessary computation and preserving real-time performance.

This unified decision process ensures that all system components influence behavior through a single action-selection stage, preventing conflicts between reactive control and higher-level reasoning.

Figure 2 provides a block-diagram overview of the corrected dual-regime navigation architecture and the interaction between its components. The navigation objective is provided as a goal representation defined by a textual description, consistent with the structured prompts used in the implementation. This goal input, expressed either as a short natural-language instruction or a semantic scene description, conditions the VLM scene understanding and hierarchical planning modules without directly influencing perception-driven reactive control.

We next describe how perceptual invariants are transformed into action preferences through an affordance-based control formulation.

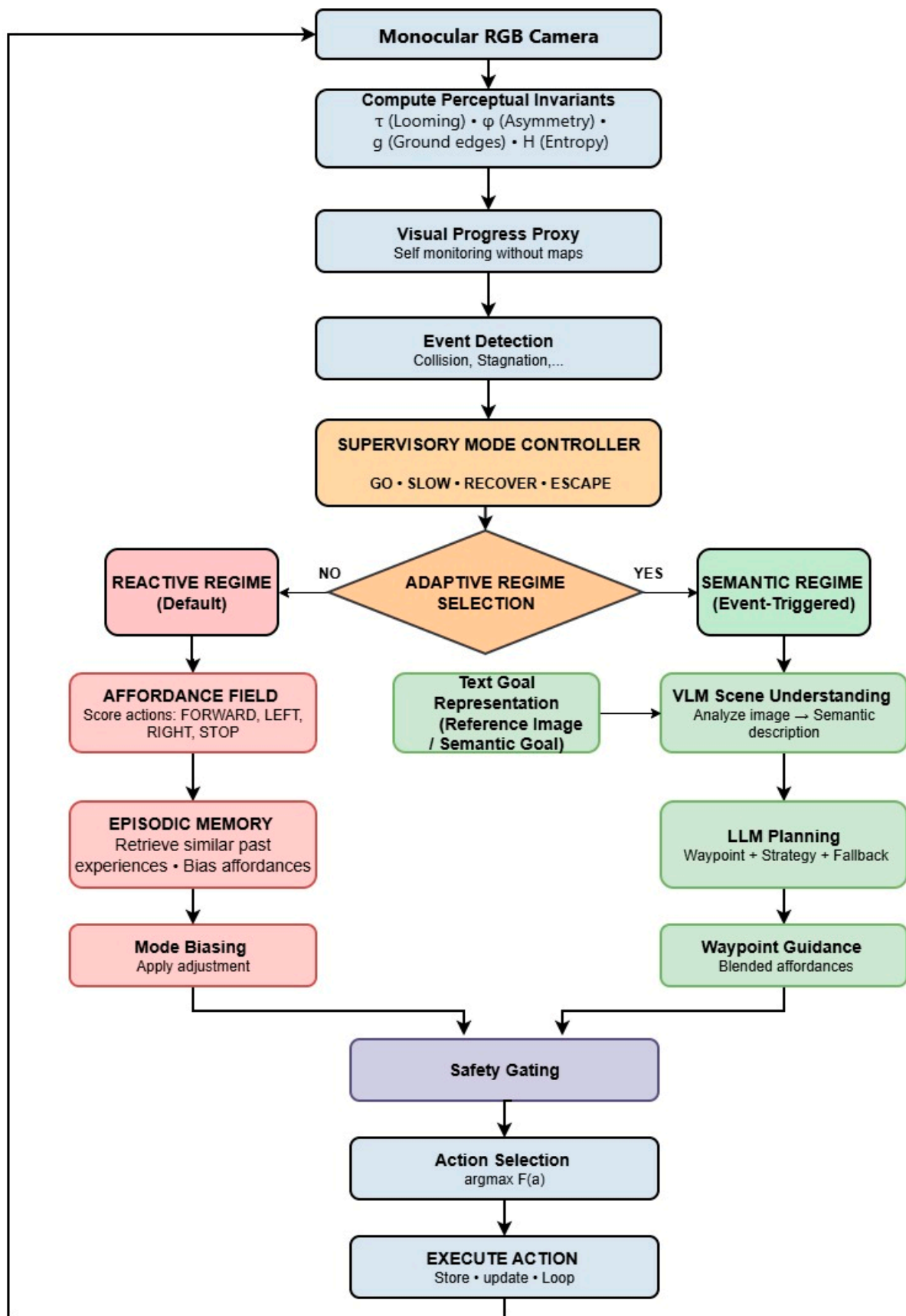


Figure 2. Block diagram of the Cognitive Hybrid Navigation System.

6. Affordance Field

Rather than selecting actions directly from raw sensory input, the proposed system transforms perceptual signals into an affordance field: a continuous scoring of candidate actions based on their local safety and utility. This field provides a structured intermediate representation between low-level perception and higher-level supervisory control.

At each time step, the affordance field assigns a score to each discrete action:

$$\mathcal{A} = \{\text{FORWARD}, \text{TURN_LEFT}, \text{TURN_RIGHT}, \text{STOP}\} \quad (8)$$

using a weighted combination of invariant-based heuristic terms.

6.1. Base Affordance Model

For each action $a \in \mathcal{A}$, an affordance score is computed as:

$$F(a \mid \mathbf{z}_t) = w_\tau f_\tau(a, \tau_t) + w_g f_g(a, g_t) + w_H f_H(a, H_t) + w_\phi f_\phi(a, \phi_t) \quad (9)$$

where $\mathbf{z}_t = (\tau_t, \phi_t, g_t, H_t)$ is the perceptual invariant vector, $f(\cdot)$ are signal-to-action mapping functions, and w_τ, w_g, w_H, w_ϕ are fixed scalar weights.

This formulation is an engineered heuristic model inspired by ecological theories of affordances in perception [31,32] and prior robotic affordance learning models [33]. The looming-inspired signal follows classical time-to-contact control principles [34]. Ground clutter and entropy capture traversability and perceptual uncertainty [35,36], while motion asymmetry reflects optical-flow-based obstacle avoidance principles [37]. Episodic memory modulation is inspired by cognitive models of episodic control [38,39].

6.2. Signal-to-Action Mappings

(a) Looming term f_τ

Encourages forward motion when the path is open and stopping under high looming risk:

$$f_\tau(\text{FORWARD}, \tau_t) = \log(1 + \tau_t), f_\tau(\text{STOP}, \tau_t) = \frac{1}{\tau_t + \varepsilon}, \quad (10)$$

Neutral for turning:

$$f_\tau(\text{TURN_LEFT}, \tau_t) = f_\tau(\text{TURN_RIGHT}, \tau_t) = 0.$$

(b) Ground-edge term f_g

Penalizes forward motion when ground clutter is high and favors turning:

$$f_g(\text{FORWARD}, g_t) = -g_t, f_g(\text{TURN_LEFT}, g_t) = g_t, \quad (11)$$

$$f_g(\text{TURN_RIGHT}, g_t) = g_t, f_g(\text{STOP}, g_t) = 0.$$

(c) Entropy term f_H

Discourages movement in visually complex scenes and favors stopping:

$$f_H(\text{FORWARD}, H_t) = -H_t, f_H(\text{STOP}, H_t) = H_t, \quad (12)$$

Neutral for turning.

(d) Motion asymmetry term f_ϕ

Steers away from the side with stronger visual motion:

$$f_{\phi}(\text{TURN_LEFT}, \phi_t) = -\phi_t, f_{\phi}(\text{TURN_RIGHT}, \phi_t) = \phi_t, \quad (13)$$

Neutral for forward motion and stopping.

6.3. Action Selection

The raw affordance scores are normalized and passed to an argmax operator:

$$a^* = \arg \max_{a \in A} F(a \mid z_t) \quad (14)$$

This action represents the default local control decision and is subsequently modulated by the supervisory controller and reflex safety layer.

6.4. Episodic Memory Bias

During execution, episodic memory introduces a multiplicative bias on the affordance scores:

$$F'(a) = F(a) \cdot m(a), \quad (15)$$

where $m(a)$ is a memory-derived scaling factor computed from prior outcomes in similar perceptual conditions.

This mechanism allows accumulated experience to influence behavior while preserving the primacy of perception-driven safety constraints.

7. Supervisory Cognitive Modes and Strategy Control

The principal contribution of this work is a supervisory control layer that dynamically selects an appropriate behavioral mode based on perception-driven event monitoring. Rather than directly replacing low-level control, the supervisor reshapes the decision landscape of the affordance field by biasing action preferences according to recent perceptual outcomes.

Unlike purely reactive navigation strategies, which continuously execute locally optimal actions, the proposed supervisor evaluates whether recent behavior produces meaningful perceptual improvement. Monitoring relies exclusively on vision-derived signals and the visual progress proxy introduced in Section 4, without spatial priors.

Supervisory decisions are driven by an explicit event detection stage that identifies navigation-critical conditions such as stagnation, deterioration, elevated uncertainty, or immediate danger. These events influence both supervisory mode selection and the activation of higher-level semantic navigation. The supervisory layer therefore acts as an intermediate strategy controller that bridges fast perception-driven affordances and slower adaptive reasoning processes.

The mechanism is formalized as a finite-state controller operating over a small set of interpretable supervisory modes. These modes address common failure patterns of reactive navigation, including repeated ineffective actions, oscillations near obstacles, and prolonged hesitation under uncertainty [40,41].

7.1. Mode Set

The controller operates over four supervisory modes:

$$\mathcal{M} = \{\text{GO}, \text{SLOW}, \text{RECOVER}, \text{ESCAPE}\}$$

Each mode represents a short-horizon behavioral strategy rather than a direct control policy (Table 2).

Table 2. Modes set.

Mode	Primary Trigger	Behavioral Role
GO	Positive visual progress	Default navigation using affordance-driven control
SLOW	Uncertainty or mild stagnation	Reduce forward bias and encourage cautious motion
RECOVER	Collision or sustained deterioration	Escape local traps through controlled turning
ESCAPE	Immediate danger or high-risk perception	Rapid evasive maneuver with strong safety bias

This reduced mode set consolidates earlier exploratory and waiting behaviors into unified safety-oriented strategies, simplifying supervisory control while preserving adaptability.

7.2. Event Triggers

Transitions between modes are governed by events computed exclusively from perceptual invariants and the visual progress proxy P_t :

- **Collision:** physical contact detected by the platform.
- **Stagnation:** smoothed visual progress remains near zero for N_s consecutive steps.
- **Deterioration:** sustained negative visual progress indicating increasing perceptual risk.
- **Uncertainty:** elevated image entropy H_t and ground-edge density g_t indicating ambiguous or cluttered scenes.
- **Immediate Danger:** high looming or ground-edge signals indicating imminent collision risk.

All triggers rely solely on monocular vision-derived signals and do not use spatial coordinates, maps, or goal-distance information.

7.3. Priority Hierarchy

When multiple events occur simultaneously, mode selection follows a strict safety-first hierarchy:

$$\text{ESCAPE} > |\text{RECOVER}| > |\text{SLOW}| > |\text{GO}|$$

This ordering ensures that rapid evasive behavior takes precedence over recovery or exploratory adjustments, reflecting the safety-dominant design of the architecture [42,43].

7.4. Timed Persistence

To prevent rapid oscillations between strategies, each non-GO mode persists for a bounded duration (typically 3–18 control steps). Temporal persistence allows each behavioral strategy sufficient time to influence perceptual feedback before reevaluation, stabilizing navigation while maintaining responsiveness.

This persistence mechanism also stabilizes regime selection by preventing rapid switching between reactive and semantic navigation. Semantic activation is maintained until clear exit conditions—such as sustained positive progress or waypoint completion—are observed.

7.5. Mode-Specific Control Policies

Supervisory modes do not replace the affordance controller. Instead, they bias or constrain action scoring within the shared affordance field:

- **GO:** standard processing pipeline with minimal bias.
- **SLOW:** reduces forward-motion weighting and increases turning diversity under uncertainty.

- **RECOVER:** suppresses forward actions and promotes controlled turning to escape local traps.
- **ESCAPE:** strongly penalizes forward motion and prioritizes evasive turning actions.

Because all modes operate through affordance modulation, final action selection remains unified and subject to reflex safety gating.

7.6. Discussion and Human Analogy

The supervisory modes implement a perception-driven self-monitoring process that separates strategy selection from low-level action execution. This design reflects functional similarities to human navigation behavior, where individuals slow down under uncertainty, adjust direction after repeated failure, and execute rapid evasive actions when danger is detected [44,45]. The analogy is conceptual rather than biological; the system remains grounded in engineered perceptual signals and deterministic supervisory logic.

8. Hierarchical Semantic Navigation

This section describes the semantic navigation component within the architecture introduced in Section 5. Semantic reasoning is activated selectively to complement perception-driven control when local visual cues are insufficient for effective navigation.

8.1. Motivation

Perception-driven navigation can fail in environments that lack clear directional cues or exhibit ambiguous visual structure. Typical failure cases include symmetric layouts, repeated ineffective actions, or situations where perceptual risk remains moderate while visual progress is near zero. In such scenarios, local perception alone does not provide sufficient information to guide navigation effectively.

Semantic interpretation of the scene can provide higher-level structure, such as identifying navigable regions, openings, or landmarks. However, continuous semantic inference is computationally expensive and may introduce instability. For this reason, semantic reasoning is activated only under persistent navigation difficulty.

8.2. Event-Triggered Semantic Activation

Semantic navigation is activated when supervisory monitoring detects persistent navigation difficulty (Section 5). Typical triggers include prolonged stagnation, sustained deterioration, elevated uncertainty, or repeated recovery behaviors without improvement.

This event-driven activation ensures that semantic reasoning is grounded in observable performance limitations rather than applied continuously, preserving computational efficiency and stability.

8.3. Hierarchical Planning Structure

Semantic navigation operates through a three-level planning hierarchy:

Level 2—Global Goal

Represents the abstract navigation objective defined through visual or semantic conditions.

Level 1—Visual Waypoints

Intermediate targets defined as visually recognizable structures such as doorways, corridor openings, or navigable regions. The robot progresses toward a waypoint until it is reached or invalidated.

Level 0—Immediate Action Bias

Short-horizon directional preferences (typically 3–5 steps) that influence action selection without overriding perception-driven control.

This hierarchical formulation enables multi-step reasoning while maintaining responsiveness and safety.

8.4. VLM Scene Understanding

When semantic navigation is activated, the current RGB observation is analyzed by the VLM to extract structured scene information, including:

- navigable regions and relative direction,
- obstacle layout and blocking severity,
- scene type (corridor, room, junction),
- distinctive landmarks suitable for waypoint definition.

The VLM output is converted into a structured semantic scene description that serves as input to the LLM planning stage.

Example VLM Query

You assist a mobile robot navigating indoors.

From this image, extract navigation-relevant information only:

- *Scene type (corridor, room, junction, open space)*
- *Navigable regions (type, direction, distance: close/medium/far)*
- *Major obstacles*
- *Distinctive landmarks useful for navigation*

Return structured JSON.

8.5. LLM Hierarchical Planning

The LLM receives:

- VLM scene description,
- current supervisory mode,
- recent visual progress history,
- episodic memory summary,
- robot capability constraints.

The LLM generates a structured semantic plan consisting of:

1. Immediate action bias (short-horizon directional preference),
2. Visual waypoint description,
3. High-level navigation strategy explanation,
4. Fallback guidance if execution fails.

The LLM does not generate executable control commands. Instead, its output is interpreted as symbolic guidance that biases the affordance field prior to final safety gating and action selection.

Example LLM Planning Query

You assist a vision-only indoor mobile robot.

Constraints:

- *No map or localization*
- *Discrete actions only: FORWARD, TURN_LEFT, TURN_RIGHT, STOP*
- *Low-level safety is handled separately*

Given scene description and recent progress:

Return:

1. *Immediate action bias (3–5 steps)*
2. *Visual waypoint description*
3. *Short strategy explanation*
4. *Fallback action if plan fails*

The LLM acts as an intermediate reasoning layer between perception and control. It does not generate executable motor commands but instead produces structured semantic guidance, including waypoint descriptions and short-horizon action biases. These outputs are integrated as soft constraints within the affordance field, ensuring that perception-driven safety and reflex gating remain dominant in the final action selection process.

8.6. Semantic Goal Stack

A semantic goal stack maintains hierarchical planning state, including:

- global navigation objective,
- active visual waypoint,
- immediate action bias,
- fallback plan,
- waypoint execution history.

The stack enables consistent multi-step execution while allowing adaptive updates driven by perceptual feedback.

8.7. Waypoint Monitoring

During semantic navigation, waypoint status is periodically evaluated using low-frequency VLM analysis. Waypoint states include:

- Reached,
- Progressing,
- Stagnating,
- Diverging,
- Lost.

Waypoint monitoring allows early failure detection and adaptive strategy adjustment without interrupting reactive control.

8.8. Fallback Strategy and Replanning

Fallback activation occurs when:

- a waypoint is not detected across multiple checks,
- execution exceeds expected duration,
- perceptual risk increases significantly,
- repeated RECOVER or ESCAPE events occur during waypoint pursuit.

Fallback strategies include switching to an alternative waypoint or triggering semantic replanning.

8.9. Safety and Grounding Constraints

Semantic navigation operates under strict grounding rules:

- semantic outputs are structured symbolic biases only,
- all affordance modifications remain subject to reflex safety gating,
- unsafe actions are vetoed before execution.

This ensures that semantic reasoning enhances navigation without compromising stability or safety.

8.10. Computational Efficiency

Semantic planning operates at a significantly lower frequency than the reactive control loop:

- full semantic planning: infrequent event-driven calls,
- waypoint monitoring: periodic lightweight checks,

- reactive navigation remains continuously active.

This separation preserves real-time performance on embedded platforms.

8.11. Role Within the Overall Architecture

Hierarchical semantic navigation functions as a strategic planning layer rather than a low-level controller. It does not estimate position, construct maps, or bypass safety constraints. Instead, it provides structured semantic guidance that biases perception-driven control during challenging navigation phases, enabling recovery from ambiguous or repetitive failure states while maintaining real-time responsiveness.

9. Adaptive Regime Selection

The navigation system operates through a cooperative interaction between perception-driven control and higher-level semantic reasoning. Rather than functioning as independent controllers, these processes are integrated within the unified affordance-based framework described in Section 5. Adaptive regime selection regulates the influence of semantic guidance based on observed navigation performance.

Reactive affordance-based control remains continuously active and serves as the primary mechanism for real-time navigation. This ensures low-latency response and robust safety behavior under normal conditions. Semantic reasoning is introduced only when reactive behavior alone becomes insufficient to produce meaningful progress.

The activation of semantic guidance is governed by supervisory events derived from perception-driven monitoring. Persistent navigation difficulty—such as sustained stagnation, repeated deterioration, or elevated uncertainty—indicates that local perceptual cues are insufficient for effective decision making. Under these conditions, semantic reasoning is activated to provide structured guidance that complements reactive control.

Regime selection is therefore implemented as a continuous modulation process rather than a discrete switch. When active, semantic outputs influence behavior by biasing the shared affordance representation, while perception-driven evaluation remains dominant. This ensures that higher-level reasoning enhances navigation without compromising responsiveness or safety.

To maintain stability, regime transitions incorporate temporal persistence constraints. Once semantic guidance is activated, it remains active for a bounded duration to allow consistent execution of the current strategy. Deactivation occurs when navigation performance improves, as indicated by sustained positive progress or a return to stable supervisory conditions. Conversely, increasing perceptual risk or immediate danger reduces the influence of semantic guidance, allowing reactive safety responses to dominate.

This adaptive mechanism enables efficient allocation of computational resources by limiting semantic reasoning to situations where it provides clear benefit. At the same time, it preserves the robustness of perception-driven control by maintaining a unified, safety-constrained decision process throughout execution.

10. Episodic Memory and Bias Adaptation

To enable long-term adaptation without retraining any neural model, the proposed navigation system incorporates an episodic memory module that records past experiences and uses them to bias future decisions. Rather than learning through gradient updates, the robot improves by remembering the perceptual consequences of its actions in similar visual situations. This memory functions as a lightweight, lifelong knowledge base that captures both successful and failed navigation behaviors, enabling gradual environment-specific adaptation without explicit mapping or localization [46,47].

Each interaction step is stored as an episode described by a compact perceptual state–action–outcome tuple:

$$e_i = (\tau_i, \phi_i, g_i, H_i, a_i, \text{blocked}_i, \Delta R_i, \text{outcome}_i) \quad (16)$$

where τ_i denotes the looming-inspired brightness expansion heuristic, ϕ_i the left–right motion asymmetry, g_i the ground-edge density, and H_i the image entropy. The term a_i is the executed action, and blocked_i indicates whether forward motion was physically blocked by a collision. The quantity ΔR_i represents the change in perceptual risk following the action, computed from the visual progress proxy defined in Section 4, while outcome_i encodes qualitative results such as safe progress or collision.

Importantly, no goal coordinates, goal distances, or metric spatial information are stored or used. All memory content is derived exclusively from monocular visual perception and observed action outcomes. Episodes are stored in a bounded, JSON-based memory buffer, allowing thousands of past experiences to be retained with minimal computational overhead.

At runtime, the current perceptual invariant vector is compared to stored episodes using a similarity metric in invariant space. A small support set of the most similar past experiences is retrieved. When sufficient matches are available, memory is considered reliable and is used to generate experience-based behavioral biases.

These biases operate in two complementary ways. First, affordance multipliers rescale current affordance scores, reinforcing actions that previously reduced perceptual risk and suppressing actions that frequently led to collisions or stagnation under similar visual conditions. Second, memory induces adaptive trigger shifts, slightly adjusting uncertainty thresholds—such as entropy or ground-edge limits—so that the robot behaves more cautiously in perceptually risky contexts and more permissively in historically safe ones.

Rather than directly commanding actions, episodic memory softly reshapes the decision landscape. It operates in synergy with the affordance field, reflex safety layer, and cognitive modes, ensuring that accumulated experience influences behavior while immediate safety constraints remain dominant. Over time, this mechanism reduces repeated failure patterns and enables environment-specific adaptation, effectively turning the system into a self-improving cognitive navigator capable of lifelong learning without retraining.

11. Algorithm: Event-Driven Dual-Regime Navigation

This section summarizes the complete control logic of the proposed dual-regime navigation architecture. The algorithm integrates perception-driven affordance control, supervisory mode selection, episodic memory bias, and event-triggered hierarchical semantic planning within a unified control loop.

The system operates as an event-driven perception-first controller in which reactive affordance evaluation remains continuously active, while semantic reasoning is conditionally activated when supervisory monitoring detects persistent navigation difficulty.

11.1. Notation

Let:

- I_t —RGB image at time step t
- $z_t = (\tau_t, \phi, g, H)$ —perception-driven signal vector
- R_t —perceptual risk estimate
- P_t —visual progress proxy
- $M_t \in \{GO, SLOW, RECOVER, ESCAPE\}$ —supervisory mode
- $\mathcal{R}_t \in \{Reactive, Semantic\}$ —active navigation regime

- $A = \{FORWARD, TURN_LEFT, TURN_RIGHT, STOP\}$ —discrete action set
- $F(a)$ —affordance score for action a

11.2. Main Control Loop

Initialize:

EpisodicMemory $\leftarrow$ empty buffer

Regime $\leftarrow$ Reactive

Mode $\leftarrow$ GO

SemanticPlan $\leftarrow$ None

Loop at each control step t:

1. PERCEPTION

Capture RGB image I_t

Compute perceptual signals $z_t = (\tau, \phi, g, H)$

Compute perceptual risk R_t

Compute visual progress P_t

2. EVENT DETECTION

Detect events from perception signals:

Collision

Stagnation

Deterioration

High Uncertainty

Immediate Danger

3. SUPERVISORY MODE SELECTION

Mode $\leftarrow$ ModeController (Events, P_t , History)

«Modes $\in \{GO, SLOW, RECOVER, ESCAPE\}$ »

4. ADAPTIVE REGIME SELECTION

if PersistentFailure (Events, History):

Regime $\leftarrow$ Semantic

else if Stable PositiveProgress():

Regime $\leftarrow$ Reactive

«Reactive affordance evaluation remains active regardless of regime.»

5. REACTIVE AFFORDANCE EVALUATION (ALWAYS ACTIVE)

For each action $a \in A$: $F(a) \leftarrow$ BaseAffordance(a, z_t)

6. EPISODIC MEMORY MODULATION

MemorySet $\leftarrow$ RetrieveSimilarEpisodes(z_t)

if MemoryReliable(MemorySet):

$F(a) \leftarrow F(a) * \text{MemoryBias}(a, \text{MemorySet})$

AdjustUncertaintyThresholds(MemorySet)

7. SUPERVISORY MODE BIASING

$F(a) \leftarrow \text{ApplyModeBias}(F(a), \text{Mode})$

«Modes reshape affordance scores but never replace action selection»

8. SEMANTIC NAVIGATION (CONDITIONAL)

if Regime == Semantic:

if NeedNewSemanticPlan():

SceneDesc $\leftarrow$ VLM_Analyze(I_t)

```

SemanticPlan  $\leftarrow$  LLM_GeneratePlan(SceneDesc, History, Constraints)
SemanticBias  $\leftarrow$  ExtractActionBias(SemanticPlan)
F(a)  $\leftarrow$  Blend(F(a), SemanticBias)
WaypointStatus  $\leftarrow$  CheckWaypointProgress(SemanticPlan, It)
if WaypointReached(WaypointStatus):
    UpdatePlanOrReturnToReactive()
if SemanticFailureDetected(WaypointStatus):
    SemanticPlan  $\leftarrow$  ReplanOrFallback()
«Semantic outputs act only as affordance biases».
9. REFLEX SAFETY GATING
F(a)  $\leftarrow$  ApplySafetyOverrides(F(a), zt)
10. ACTION SELECTION
a*  $\leftarrow$  argmaxa F(a)
11. EXECUTION
Execute action a*
12. MEMORY UPDATE
Store episode: (zt, a*, blocked,  $\Delta R_t$ , outcome)
End Loop

```

12. Experiments and Results

We evaluate the proposed dual-regime navigation architecture in both simulation and real-world deployment. The experiments assess navigation success, safety, efficiency, computational cost, and the functional contributions of (i) supervisory mode control, (ii) episodic memory bias adaptation, and (iii) event-triggered hierarchical semantic navigation (VLM–LLM).

The proposed architecture is designed for mapless navigation using monocular RGB vision as the primary sensing modality. The system explicitly excludes spatial priors, focusing instead on perception-grounded decision-making derived directly from camera observations. Under this formulation, direct comparison with SLAM-based navigation systems is difficult to interpret, as SLAM methods rely on accurate state estimation, map construction, and often multi-sensor fusion—capabilities intentionally outside the scope of this work.

Similarly, modern reinforcement learning and imitation learning navigation approaches typically require large-scale training data, task-specific reward design, and substantial computational resources for policy training and deployment [48]. In contrast, the proposed system operates in a training-free manner, enabling incremental adaptation through episodic memory without offline policy optimization. It is therefore targeted at deployment scenarios where only a monocular camera is available and mapping or training infrastructure may be impractical. Accordingly, evaluation focuses on comparisons within perception-driven and hybrid semantic navigation architectures operating under similar sensing and mapless constraints.

In this mapless formulation, navigation goals are not specified as metric coordinates or global spatial targets. Instead, they are defined through visual or semantic conditions observable from monocular RGB input. During deployment, goal completion is achieved when the robot detects scene characteristics consistent with the objective, such as an open navigable area, a distinctive landmark, or a semantically defined configuration inferred through VLM–LLM reasoning. Navigation thus emerges from perception-driven

exploration combined with event-triggered semantic guidance, rather than explicit metric path planning.

In the simulation experiments, the goal is represented by the green spherical marker shown in Figure 3. This marker defines the target region solely for evaluation and visualization purposes. The controller does not receive the sphere's coordinates, distance, or any explicit spatial information. Instead, goal-directed behavior relies entirely on the vision-based formulation described in Section 4.3, where progress is determined through visual similarity and semantic consistency derived from monocular RGB observations.

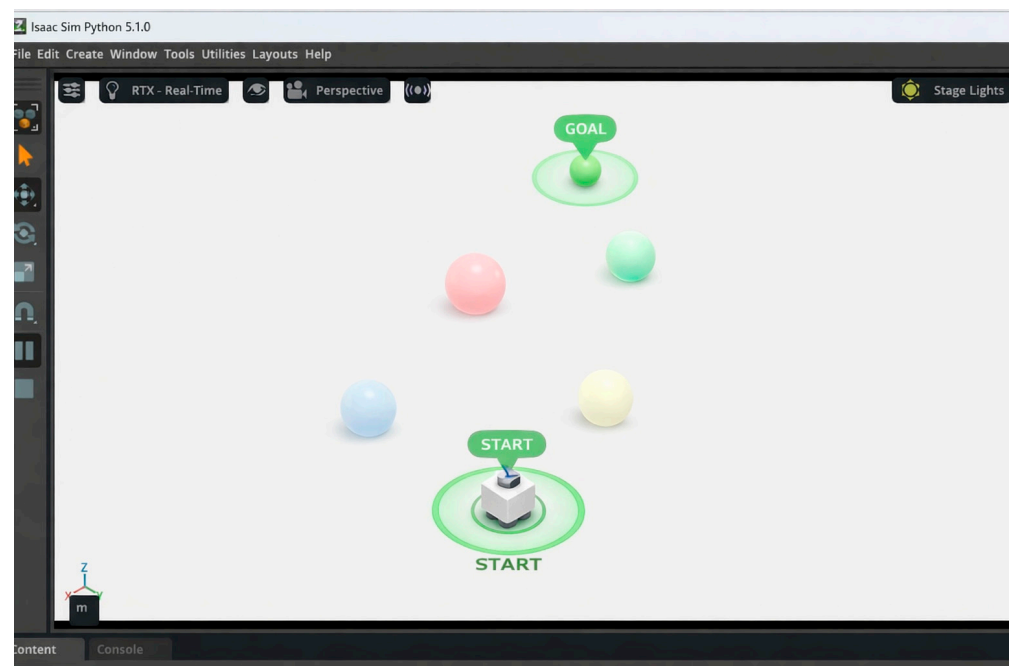


Figure 3. Simulation environment with goal marker (green sphere used for evaluation only).

We compare the proposed method with three categories of navigation approaches: (i) perception-driven reactive methods, (ii) continuous language-guided navigation using always-on VLM–LLM inference, and (iii) learning-based approaches discussed qualitatively. Direct comparison with RL-based methods is limited by differences in sensing assumptions, as many such methods rely on goal-distance signals or privileged state information not available in our formulation.

12.1. Simulation Environment

All simulation experiments are conducted in NVIDIA Isaac Sim using a lightweight indoor navigation benchmark with metric world units (1 unit = 1 m). The environment consists of a flat planar scene with fixed lighting and four static spherical obstacles placed at predefined locations.

The robot is modeled as a simple rigid-body proxy controlled in discrete time using the action set:

$$\{\text{FORWARD}, \text{TURN_LEFT}, \text{TURN_RIGHT}, \text{STOP}\}.$$

A FORWARD action translates the robot by 0.12 m along its heading, while turning actions rotate it by $\pm 15^\circ$. The robot is equipped with a monocular RGB camera tilted downward by 10° , producing 128×128 images at 30 Hz. No depth sensors, LiDAR, odometry, maps, or localization signals are provided to the controller.

For evaluation purposes only, a circular region of radius 0.35 m in the horizontal plane is defined in the simulator to verify successful goal attainment. This region corresponds

to the green spherical marker shown in Figure 3 and is used exclusively for performance measurement and visualization. The controller does not receive the region's coordinates or distance information during execution. Each episode terminates when the robot enters this region or after a maximum of 1000 time steps. Collisions are detected using the simulator contact signal and are exposed as binary events used by the recovery mode and evaluation metrics. An example configuration is shown in Figure 4.

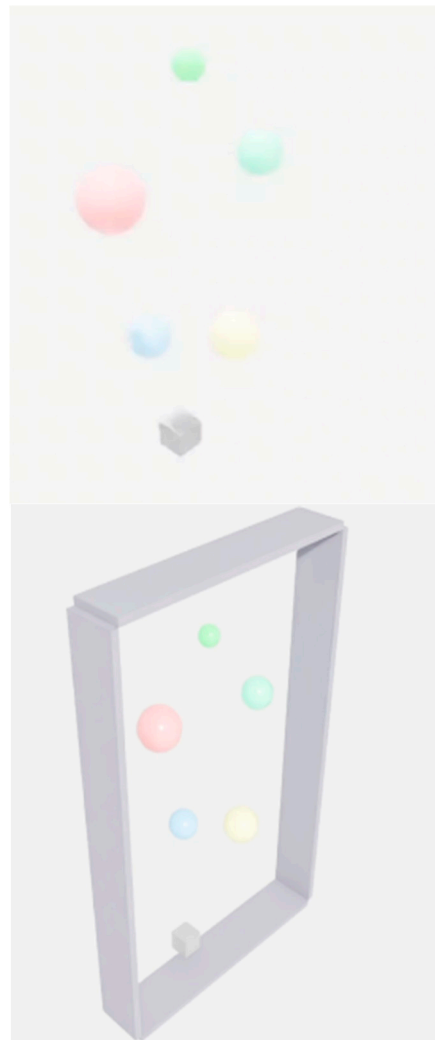


Figure 4. Example start and goal configuration. The goal is not directly observable by the agent and is shown for visualization purposes only.

We compare the proposed system against baselines and ablations designed to isolate the contributions of each architectural component:

- **Reactive Affordance Only:** affordance-based action selection without supervisory modes, memory, or semantic navigation.
- **Reactive + Reflex Safety:** adds the reflex safety gating layer but no supervisory modes, memory, or semantic navigation.
- **Reactive + Supervisory Modes:** adds supervisory mode biasing (GO/SLOW/RECOVER/ESCAPE) but no memory or semantic navigation.
- **Reactive + Always-On LLM:** continuous VLM–LLM guidance applied throughout the episode (no event-triggering, no semantic waypoint stack).

- **Dual-Regime (Ours):** full system with supervisory modes, episodic memory bias, and event-triggered hierarchical semantic navigation (VLM scene understanding + LLM hierarchical planning + waypoint monitoring + fallback).

This set separates (i) safety reflexes, (ii) strategy-level supervision, (iii) experience-based biasing, and (iv) semantic planning and its activation policy.

Key implementation parameters include empirically tuned weights for perceptual signals, normalization window size for risk estimation, and thresholds for event detection (stagnation, uncertainty, and danger conditions). The control loop operates at 10–15 Hz, while semantic reasoning is triggered at a lower frequency (0.1–0.4 Hz), ensuring real-time performance.

12.2. Evaluation Metrics

We measure complementary metrics capturing task success, efficiency, safety, and internal system behavior:

- **Success Rate:** percentage of episodes reaching the goal within the step limit
- **Steps to Goal:** number of steps required in successful runs
- **Collision Count:** total obstacle contacts per episode
- **Backtracking Ratio:** fraction of steps that increase ground-truth distance to the goal
- **STOP Loop Ratio:** fraction of time spent issuing repeated STOP actions
- **Semantic Calls:** number of semantic navigation activations per episode (VLM–LLM planning calls).

Although ground-truth distance is used to compute efficiency and backtracking metrics, it is never available to the controller during execution.

12.3. Implementation Parameters

The system uses the following parameter values:

- Risk weights: $w_1 = 0.4$, $w_2 = 0.2$, $w_3 = 0.25$, $w_4 = 0.15$
- Normalization window: $W = 20$ frames
- Motion threshold: $\delta = 0.01$
- Stagnation condition: $P_t < 0.01$ for 10 consecutive steps
- Entropy threshold: $H > 4.5$
- Looming threshold: $\tau < 0.2$
- Ground-edge threshold: $g = 25$
- Mode persistence: 5–10 control steps depending on mode
- Memory retrieval: $k = 5$ nearest neighbors using cosine similarity

12.4. Simulation Results

Table 3 reports average performance over 100 episodes per method. The proposed dual-regime system consistently outperforms baselines, achieving the highest success rate, the lowest collision frequency, and improved path efficiency while maintaining low semantic reasoning frequency.

The reported values correspond to averages over 100 episodes, with variability indicated by standard deviations where applicable. Additional analysis confirms consistent performance improvements across different initial conditions, demonstrating robustness of the proposed architecture.

The affordance-only controller frequently becomes trapped in oscillatory behaviors near obstacles. Adding reflex safety reduces collisions but does not prevent prolonged stagnation. Introducing supervisory modes substantially improves stability by detecting stagnation and deterioration through the vision-based progress proxy and reshaping affordance preferences accordingly. Continuous LLM guidance improves success mod-

estly but incurs high call frequency and can amplify STOP loops due to inconsistent or latency-prone suggestions.

Table 3. Average performance of navigation methods in simulation.

Method	Success Rate	Steps to Goal	Collisions	Backtracking	STOP Loops	Semantic Calls
Reactive Affordance Only	42%	310 ± 95	14.3	38%	27%	0
Reactive + Reflex Safety	55%	280 ± 82	9.1	29%	19%	0
Reactive + Supervisory Modes	74%	235 ± 64	4.6	15%	7%	0
Reactive + Always-On LLM	61%	260 ± 76	7.8	24%	31%	120
Dual-Regime (Ours)	92%	195 ± 41	1.9	6%	2%	18

All evaluated baselines operate under the same monocular, mapless sensing constraints, ensuring a fair comparison with the proposed method.

In contrast, the full system benefits from event-triggered hierarchical semantic navigation, which is activated only when persistent failure is detected. When active, semantic waypoint guidance provides multi-step directional bias within the shared affordance field, helping the agent escape ambiguous layouts without dominating control. Episodic memory further reduces repeated failure patterns across episodes by biasing affordance scores and adapting uncertainty thresholds based on past outcomes.

Trajectory visualizations (Figure 5) show smoother convergence with fewer oscillations under the proposed system compared to reactive baselines.

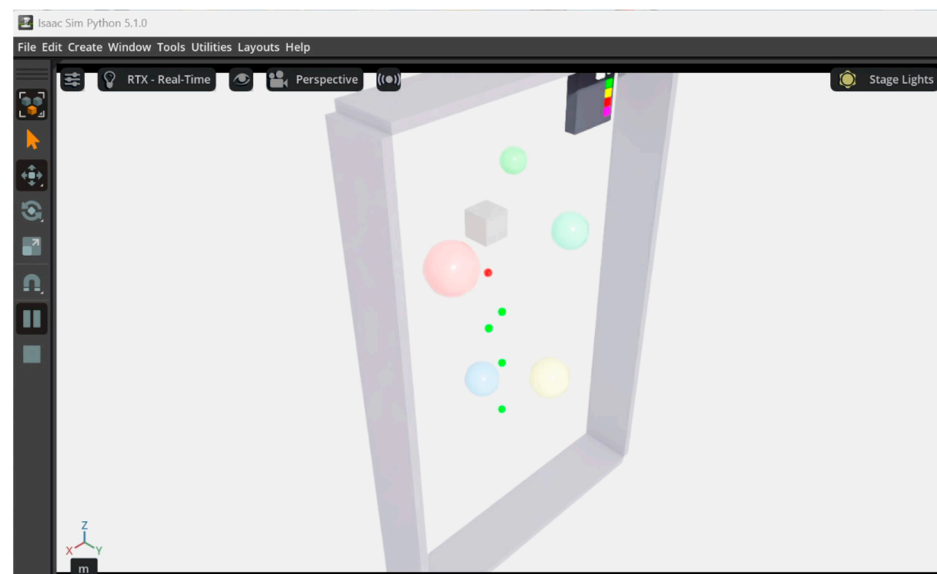


Figure 5. Trajectory visualization with supervisory modes (GO, SLOW, RECOVER, ESCAPE) and regime transitions (Reactive (Sphere) vs. Semantic (Cube)).

12.5. Real-World Deployment on Embedded Platform

To evaluate feasibility on resource-constrained hardware, we deployed the system on a mobile robot (Figure 6) equipped with an STM32N6 microcontroller and a Raspberry Pi 5 (16 GB RAM), along with a monocular RGB camera directly interfaced to the STM32N6. In contrast to the simulation setup, the embedded architecture was split across two processing units. Low-level perception, affordance estimation, and motor control were fully implemented on the STM32N6, enabling real-time execution of the control loop.

High-level semantic reasoning modules, including the Vision–Language Model (VLM) and Large Language Model (LLM), were executed on the Raspberry Pi 5. As in simulation, navigation relied exclusively on monocular vision without LiDAR, depth sensing, or metric localization.

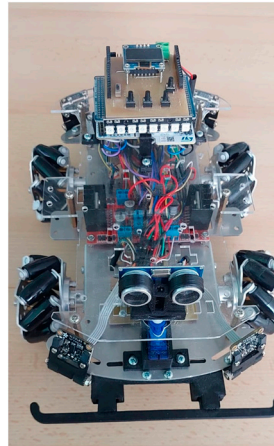


Figure 6. Robot used for navigation.

Vision–language and language models were executed locally on the Raspberry Pi 5 using a containerized runtime supporting quantized GGUF models. Scene understanding used a compact 4-bit VLM (e.g., Qwen-VL-2B or LLaVA-1.5-7B), while hierarchical planning used a compact 4-bit instruction-tuned LLM (e.g., Phi-3-mini or Qwen-2.5-3B). Average inference latency was approximately 180–250 ms per VLM call and 250–350 ms per LLM call for short prompts (≤ 128 tokens). Due to event-triggered activation, the semantic module operated at low frequency (0.1–0.4 Hz) during difficult navigation phases and remained inactive during most normal motion, acting asynchronously as a high-level bias layer. Meanwhile, the main perception–affordance–control loop was executed on the STM32N6 at 10–15 Hz, ensuring stable real-time motor control independent of high-level semantic processing latency.

Our architecture uses strict temporal decoupling between the reactive control loop (STM32N6) and semantic reasoning (Raspberry Pi 5), ensuring that reactive navigation runs at 10–15 Hz deterministically and is never blocked by VLM/LLM latency.

Across 100 real-world navigation trials in cluttered indoor environments (Table 4), the robot achieved an 80% success rate with a mean collision count of 1.3 per episode (compared to 92% success and 1.9 collisions in simulation under the same controller). Failures were primarily caused by extreme lighting variation and motion blur degrading perception-driven signal estimation. Nevertheless, supervisory modes and recovery behaviors remained stable, and event-triggered semantic navigation successfully resolved several local dead-ends and ambiguous scenes.

These results indicate that the proposed dual-regime architecture is not only effective in simulation but also practical for real-world deployment on heterogeneous low-power embedded platforms. By decoupling real-time perception and motor control (STM32N6) from computationally heavier semantic reasoning (Raspberry Pi 5), the system validates the scalability of event-triggered semantic reasoning under tight computational and energy constraints.

Table 4. Real-World Embedded Deployment Performance.

Metric	Value
VLM	Qwen-VL-2B/LLaVA-1.5-7B
LLM	Phi-3-mini/Qwen-2.5-3B
Quantization	4-bit (llama.cpp/Ollama)
VLM Latency	180–250 ms
LLM Latency	250–350 ms
Semantic Trigger Rate	0.1–0.4 Hz
Real-World Success Rate	80%
Mean Collisions	1.3
Control Loop Frequency	10–15 Hz

13. Conclusions

This paper presented a dual-regime navigation architecture that combines perception-driven control with event-triggered semantic reasoning within a unified decision framework. The proposed system achieves robust mapless navigation by integrating affordance-based action evaluation, supervisory monitoring of navigation progress, and experience-driven bias adaptation through episodic memory.

The architecture maintains real-time performance through a perception-first control loop, while selectively incorporating higher-level semantic guidance only when navigation difficulty is detected. This adaptive interaction enables the system to retain the responsiveness and safety of reactive navigation while benefiting from structured reasoning in complex or ambiguous environments.

Experimental results in both simulation and real-world deployment demonstrate consistent improvements in success rate, collision reduction, and navigation efficiency compared to reactive and continuously active semantic baselines. The event-triggered design also significantly reduces the computational cost associated with high-level reasoning, enabling practical deployment on resource-constrained embedded platforms.

These results highlight the effectiveness of combining fast perception-driven control with selective semantic reasoning under a unified action-selection mechanism. More broadly, the work demonstrates that reliable navigation behavior can emerge from tightly integrated perception, supervision, and conditional reasoning, without relying on explicit spatial representations or training-based policies.

Future work will investigate extensions to multi-goal navigation, more robust semantic perception under challenging visual conditions, and the integration of longer-term memory structures to support richer environment understanding and adaptation.

Author Contributions: Conceptualization, R.F., G.O., M.A., A.M. and Y.F.; methodology, R.F., G.O., M.A., A.M. and Y.F.; software, R.F., G.O. and A.M.; validation, R.F. and G.O.; formal analysis, R.F., G.O., M.A. and A.M.; investigation, R.F., G.O., M.A. and A.M.; resources, R.F., G.O., M.A., A.M. and Y.F.; data curation, R.F.; writing—original draft preparation, R.F., G.O., M.A. and A.M.; writing—review and editing, R.F., G.O., M.A., A.M. and Y.F.; visualization, R.F., G.O., M.A., A.M. and Y.F.; supervision, R.F., G.O., M.A., A.M. and Y.F.; project administration, R.F.; funding acquisition, R.F., G.O. and Y.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Ongoing Research Funding program, (ORF-2026-698), King Saud University, Riyadh, Saudi Arabia and The APC was funded by Ongoing Research Funding program, (ORF-2026-698), King Saud University, Riyadh, Saudi Arabia and ISEN Toulon Méditerranée, France.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sun, J.; Zhao, J.; Hu, X.; Gao, H.; Yu, J. Autonomous Navigation System of Indoor Mobile Robots Using 2D LiDAR. *Mathematics* **2023**, *11*, 1455. [CrossRef]
2. Farkh, R.; Oudinet, G.; Deleruyelle, T. Evaluating a Hybrid LLM Q-Learning/DQN Framework for Adaptive Obstacle Avoidance in Embedded Robotics. *AI* **2025**, *6*, 115. [CrossRef]
3. Aremu, M.B.; Ahmed, G.; Elferik, S.; Saif, A.-W.A. Autonomous Mobile Robot Path Planning Techniques—A Review: Metaheuristic and Cognitive Techniques. *Robotics* **2026**, *15*, 23. [CrossRef]
4. Han, D.; Mulyana, B.; Stankovic, V.; Cheng, S. A Survey on Deep Reinforcement Learning Algorithms for Robotic Manipulation. *Sensors* **2023**, *23*, 3762. [CrossRef] [PubMed]
5. Levine, S.; Pastor, P.; Krizhevsky, A.; Ibarz, J.; Quillen, D. Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection. *Int. J. Robot. Res.* **2018**, *37*, 421–436. [CrossRef]
6. Huang, J.; Limberg, C.; Arshad, S.M.N.; Zhang, Q.; Li, Q. Combining VLM and LLM for Enhanced Semantic Object Perception in Robotic Handover Tasks. In *Proceedings of the 2024 WRC Symposium on Advanced Robotics and Automation (WRC SARA), Beijing, China, 23 August 2024*; IEEE: Piscataway, NJ, USA, 2024; pp. 135–140. [CrossRef]
7. Wang, J.; Shi, E.; Hu, H.; Ma, C.; Liu, Y.; Wang, X.; Yao, Y.; Liu, X.; Ge, B.; Zhang, S. Large Language Models for Robotics: Opportunities, Challenges, and Perspectives. *J. Autom. Intell.* **2025**, *4*, 52–64. [CrossRef]
8. Kessler, F.; Frankenstein, J.; Rothkopf, C.A. Human Navigation Strategies and Their Errors Result from Dynamic Interactions of Spatial Uncertainties. *Nat. Commun.* **2024**, *15*, 5677. [CrossRef]
9. Qiu, Z.; Liu, Z.; Niu, W.; Bhattacharjee, T.; Kalantari, S. EgoCogNav: Cognition-Aware Human Egocentric Navigation. *arXiv* **2025**, arXiv:2511.17581. Available online: <https://arxiv.org/abs/2511.17581> (accessed on 10 February 2026).
10. Vernon, D. The Future of Research in Cognitive Robotics: Foundation Models or Developmental Cognitive Models? *Adv. Robot. Res.* **2025**, *1*, e202500066. [CrossRef]
11. Boretti, C.; Bich, P.; Zhang, Y.; Baillieul, J. Visual Navigation Using Sparse Optical Flow and Time-to-Transit. *arXiv* **2021**, arXiv:2111.09669. [CrossRef]
12. Alfaro, A.; Maiano, L.; Papa, L.; Amerini, I. Estimating Optical Flow: A Comprehensive Review of the State of the Art. *Comput. Vis. Image Underst.* **2024**, *249*, 104160. [CrossRef]
13. Beintema, J.A. Self-Motion Perception from Optic Flow and Rotation Signals. Ph.D. Thesis, Erasmus University Rotterdam, Rotterdam, The Netherlands, 2000.
14. Wang, M.; Luo, R.; Önl, A.Ö.; Padir, T. Affordance-Based Mobile Robot Navigation among Movable Obstacles. In *Proceedings of Robotics: Science and Systems (RSS)*; IEEE: Piscataway, NJ, USA, 2021.
15. Birr, T.; Pohl, C.; Younes, A.; Asfour, T. AutoGPT+P: Affordance-Based Task Planning Using Large Language Models. *arXiv* **2024**, arXiv:2402.10778.
16. Zhou, X.; Weber, C.; Wermter, S. Vision-Based Robot Navigation through Combining Unsupervised Learning and Hierarchical Reinforcement Learning. *Sensors* **2019**, *19*, 1576. [CrossRef]
17. Schaal, S.; Atkeson, C.G. Learning Control in Robotics. *IEEE Robot. Autom. Mag.* **2010**, *17*, 20–29. [CrossRef]
18. Sapkota, R.; Cao, Y.; Roumeliotis, K.I.; Karkee, M. Vision-Language-Action Models: Concepts, Progress, Applications and Challenges. *arXiv* **2025**, arXiv:2505.04769.
19. Zhou, X.; Liu, M.; Yurtsever, E.; Zagar, B.L.; Zimmer, W.; Cao, H.; Knoll, A.C. Vision Language Models in Autonomous Driving: A Survey and Outlook. *IEEE Trans. Intell. Veh.* **2024**, 1–20. [CrossRef]
20. Tang, X.; Yan, Y.; Wang, B. Trajectory Tracking Control of Autonomous Vehicles Combining ACT-R Cognitive Framework and Preview Tracking Theory. *IEEE Access* **2023**, *11*, 137067–137082. [CrossRef]
21. Lieto, A.; Lebiere, C.; Oltramari, A. The Knowledge Level in Cognitive Architectures: Current Limitations and Possible Developments. *Cogn. Syst. Res.* **2018**, *48*, 39–55. [CrossRef]
22. Lee, D.N. A Theory of Visual Control of Braking Based on Information about Time to Collision. *Perception* **1976**, *5*, 437–459. [CrossRef]
23. Chang, J.; Li, Q.; Liang, Y.; Zhou, L. SC-AOF: A Sliding Camera and Asymmetric Optical-Flow-Based Blending Method for Image Stitching. *Sensors* **2024**, *24*, 4035. [CrossRef]
24. Häne, C.; Heng, L.; Lee, G.H.; Fraundorfer, F.; Furgale, P.; Sattler, T.; Pollefeys, M. 3D Visual Perception for Self-Driving Cars Using a Multi-Camera System. *Image Vis. Comput.* **2017**, *68*, 14–27. [CrossRef]

25. Luo, K.; Lin, M.; Wang, P.; Zhou, S.; Yin, D.; Zhang, H. Improved ORB-SLAM2 Algorithm Based on Information Entropy and Image Sharpening Adjustment. *Math. Probl. Eng.* **2020**, 2020, 4724310. [\[CrossRef\]](#)
26. Blake, A.; Bordallo, A.; Brestnichki, K.; Hawasly, M.; Penkov, S.V.; Ramamoorthy, S.; Silva, A. FPR—Fast Path Risk Algorithm to Evaluate Collision Probability. *arXiv* **2018**, arXiv:1804.05384. [\[CrossRef\]](#)
27. Majumdar, A.; Pavone, M. How Should a Robot Assess Risk? *arXiv* **2017**, arXiv:1710.11040. [\[CrossRef\]](#)
28. Xiao, X.; Dufek, J.; Murphy, R. Robot Motion Risk Reasoning Framework. *arXiv* **2019**, arXiv:1909.02531. [\[CrossRef\]](#)
29. Ma, C.-Y.; Wu, Z.; AlRegib, G.; Xiong, C.; Kira, Z. The Regretful Agent: Heuristic-Aided Navigation through Progress Estimation. *arXiv* **2019**, arXiv:1903.0160. [\[CrossRef\]](#)
30. Horn, B.K.P.; Schunck, B.G. Determining Optical Flow. *Artif. Intell.* **1981**, 17, 185–203. [\[CrossRef\]](#)
31. Gibson, J.J. *The Ecological Approach to Visual Perception*; Houghton Mifflin: Boston, MA, USA, 1979.
32. Şahin, E.; Çakmak, M.; Doğan, M.R.; Uğur, E.; Üçoluk, G. To Afford or Not to Afford: A New Formalization of Affordances in Robotics. *Robot. Auton. Syst.* **2007**, 55, 425–435.
33. Montesano, L.; Lopes, M.; Bernardino, A.; Santos-Victor, J. Learning Object Affordances: From Sensory–Motor Coordination to Imitation. *IEEE Trans. Robot.* **2008**, 24, 15–26. [\[CrossRef\]](#)
34. Tresilian, J.R. Visually Timed Action: Time-Out for Tau? *Trends Cogn. Sci.* **1999**, 3, 301–310. [\[CrossRef\]](#) [\[PubMed\]](#)
35. Ulrich, I.; Nourbakhsh, I. Appearance-Based Obstacle Detection with Monocular Vision. In *Proceedings of the AAAI Conference on Artificial Intelligence*; AAAI: Washington, DC, USA, 2000; pp. 866–871.
36. Hoiem, D.; Efros, A.A.; Hebert, M. Geometric Context from a Single Image. In *Proceedings of the IEEE ICCV*; IEEE: Piscataway, NJ, USA, 2005; pp. 654–661.
37. Srinivasan, M.V.; Zhang, S.; Lehrer, M.; Collett, T.S. Honeybee Navigation: Visual Flight Control and Odometry. *J. Exp. Biol.* **1997**, 200, 237–244. [\[CrossRef\]](#)
38. Tulving, E. Episodic and Semantic Memory. In *Organization of Memory*; Academic Press: New York, NY, USA, 1972; pp. 381–403.
39. Blundell, C.; Uria, B.; Pritzel, A.; Li, Y.; Ruderman, A.; Leibo, J.Z.; Rae, J.W.; Wierstra, D.; Hassabis, D. Model-Free Episodic Control. In *Proceedings of the 33rd International Conference on Machine Learning ICML*, New York, NY, USA, 19–24 June 2016; pp. 1331–1340.
40. Thrun, S.; Burgard, W.; Fox, D. *Probabilistic Robotics*; MIT Press: Cambridge, MA, USA, 2005.
41. Brock, O.; Khatib, O. High-Speed Navigation Using the Global Dynamic Window Approach. In *Proceedings of IEEE ICRA*; IEEE: Piscataway, NJ, USA, 1999; pp. 341–346.
42. Brooks, R.A. A Robust Layered Control System for a Mobile Robot. *IEEE J. Robot. Autom.* **1986**, 2, 14–23. [\[CrossRef\]](#)
43. Arkin, R.C. *Behavior-Based Robotics*; MIT Press: Cambridge, MA, USA, 1998.
44. Montello, D.R. Navigation. In *The Cambridge Handbook of Visuospatial Thinking*; Cambridge University Press: Cambridge, UK, 2005; pp. 257–294.
45. Kuipers, B. The Spatial Semantic Hierarchy. *Artif. Intell.* **2000**, 119, 191–233. [\[CrossRef\]](#)
46. Zou, Q.; Cong, M.; Liu, D.; Du, Y. A Neurobiologically Inspired Mapping and Navigating Framework for Mobile Robots. *Neurocomputing* **2021**, 460, 181–194. [\[CrossRef\]](#)
47. Huang, W.; Chella, A.; Cangelosi, A. A Cognitive Robotics Implementation of Global Workspace Theory for Episodic Memory Interaction with Consciousness. *IEEE Trans. Cogn. Dev. Syst.* **2024**, 16, 266–283. [\[CrossRef\]](#)
48. Kumar, A.; Zhou, A.; Tucker, G.; Levine, S. Conservative Q-Learning for Offline Reinforcement Learning. *Adv. Neural Inf. Process. Syst.* **2020**, 33, 1179–1191.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.