Enhancing Robot Navigation Policies with Task-Specific Uncertainty Management

Anonymous Author(s)
Submission Id: 362

ABSTRACT

Robots performing navigation tasks in complex environments face significant challenges due to uncertainty in state estimation arising from sensor noise, environmental changes, and incomplete information. Effectively managing this uncertainty is crucial, but the optimal approach varies depending on the specific details of the task: different tasks may require varying levels of precision in different regions of the environment. For instance, a robot navigating a crowded space might need precise localization near obstacles but can operate effectively with less precise state estimates in open areas. This varying need for certainty in different parts of the environment, depending on the task, calls for navigation policies that can adapt their uncertainty management strategies based on task-specific requirements.

In this paper, we present a framework for integrating task-specific uncertainty requirements directly into navigation policies. We introduce the concept of a Task-Specific Uncertainty Map (TSUM), which represents acceptable levels of state estimation uncertainty across different regions of the operating environment for a given task. Using TSUM, we propose **G**eneralized **U**ncertainty **I**ntegration for **D**ecision-Making and **E**xecution (GUIDE), a policy conditioning framework that incorporates these uncertainty requirements into the robot's decision-making process.

We find that conditioning policies on TSUMs not only provides an effective way to express task-specific uncertainty requirements but also enables the robot to reason about the context-dependent value of certainty and adapt its behavior accordingly. We show how integrating GUIDE into reinforcement learning frameworks allows the agent to learn navigation policies that effectively balance task completion and uncertainty management without the need for explicit reward engineering. We evaluate GUIDE at scale on a variety of real-world robotic navigation tasks and find that it demonstrates significant improvements in task completion rates compared to baseline methods that do not explicitly consider task-specific uncertainty.

KEYWORDS

Uncertainty-guided planning, Task-specific decision-making, Planning under uncertainty, Robot Navigation

ACM Reference Format:
Anonymous Author(s). 2025. Enhancing Robot Navigation Policies with Task-Specific Uncertainty Management. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 10 pages.

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), A. El Fallah Seghrouchni, Y. Vorobeychik, S. Das, A. Nowe (eds.), May 19 – 23, 2025, Detroit, Michigan, USA.

1 INTRODUCTION

Consider an autonomous surface vehicle (ASV) tasked with mapping underwater topography near a busy harbor. Performing such a task in a marine environment is often complicated by factors like waves and environmental changes. These elements can introduce significant uncertainty in the ASV's state estimation. To effectively carry out its mission, the ASV needs to manage this uncertainty. Importantly, constantly striving to reduce uncertainty is not always necessary for task completion. When navigating through areas with heavy boat traffic or obstacles, precise localization is crucial to avoid collisions and collect accurate data. However, while traversing open waters during this mission, the ASV can tolerate higher positional uncertainty, allowing it to focus on efficient task execution rather than achieving pinpoint accuracy. This scenario underscores a fundamental challenge in robotic navigation: the acceptable level of uncertainty at any location in the operation space is inherently *task-specific* and *context-dependent*.

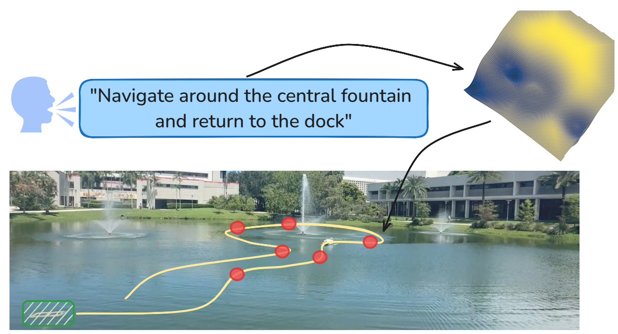

Figure 1: Illustration of our method applied to a robotic navigation task. Top Left: The ASV is assigned a navigation task. Top Right: GUIDE interprets the task and generates a representation highlighting areas where the ASV needs higher positional certainty (dark blue). Bottom: The policy executed by the ASV. The white line represents the trajectory taken by the ASV towards the dock (green rectangle), the red dots indicate locations where the ASV actively reduces its state estimation uncertainty to satisfy task-specific requirements.

Traditional approaches to uncertainty management in robotic navigation often fall into two categories: minimizing uncertainty universally [12, 13, 39, 49] or enforcing fixed uncertainty thresholds [7, 19, 53]. Such one-size-fits-all strategy may be effective in controlled settings, they lack the flexibility to adapt to tasks that demand varying degrees of uncertainty management. For instance, in the ASV's mission introduced earlier, strictly enforcing precise localization may be unnecessary when navigating open waters, where higher uncertainty does not hinder task execution.

These existing methods often neglect the varying uncertainty requirements dictated by different contexts and tasks. While some approaches attempt to balance uncertainty reduction with task performance [63, 65], they typically apply this trade-off uniformly across all situations, failing to account for task-specific needs that may vary within a single mission. Furthermore, developing effective uncertainty management strategies often require extensive reward engineering to capture the intricacies of different tasks [5, 79].

The fundamental issue lies in the disconnect between task specifications and uncertainty management. Current frameworks often treat these elements separately, resulting in sub-optimal performance when effective task completion demands varying uncertainty levels across different regions in the operating environment.

To address this disconnect, we pose the following question: *How can robots reason about the uncertainty needs of specific tasks to guide their navigation policies?*

Contributions: With the aim of bridging the gap between high-level task descriptions and low-level uncertainty handling, we propose the concept of a *Task-Specific Uncertainty Map (TSUM)*. A TSUM represents the acceptable levels of state estimation uncertainty across different regions of the environment, given a specific task. By spatially encoding the varying importance of uncertainty reduction, TSUMs provide a task-oriented representation of where precision is most critical for successful task completion.

Building upon TSUM, we introduce ***G**eneralized **U**ncertainty **I**ntegration for **D**ecision-Making and **E**xecution (GUIDE)*, a policy conditioning framework that incorporates TSUMs into navigation policies. The core proposition of GUIDE is that by incorporating the acceptable levels of state estimation uncertainty across different regions of the environment into the navigation policy, robots can better reason about the *context-dependent value of certainty*. This enables the generation of policies that adapt their behavior based on the specific uncertainty requirements of the given task.

Finally, we demonstrate how GUIDE can be integrated into a reinforcement learning framework. Specifically, we adapt the Soft Actor-Critic algorithm, resulting in the *GUIDEd Soft Actor-Critic (G-SAC)* method. G-SAC learns navigation policies that effectively balance task completion and uncertainty management without the need for explicit reward engineering tailored to each specific task.

The GUIDE framework is designed to be simple, flexible, and broadly applicable across various robotic domains. We demonstrate its effectiveness in real-time, real-world applications, focusing on marine autonomy using ASVs. Our experimental results show significant improvements in task performance when compared to state-of-the-art methods.

2 RELATED WORKS

Uncertainty Modeling in Robotic Navigation: Uncertainty management is a fundamental aspect of robotic navigation, where robots must make decisions based on imperfect information about their state and the environment [49, 57]. Probabilistic techniques such as Bayesian filters [15, 25, 41] and particle filters [11, 16, 66] have been widely used to enable robust localization and mapping in stochastic environments. These methods allow robots to estimate their position and navigate effectively despite sensor noise and environmental uncertainties. However, traditional approaches often treat uncertainty uniformly across the environment, without considering how different regions may require varying levels of certainty depending on the navigation task.

Reinforcement Learning for Navigation: Reinforcement Learning (RL) has been successfully applied to robotic navigation tasks, enabling agents to learn navigation policies through interaction with the environment [69, 70, 76]. Standard RL algorithms have been used to train robots to navigate in various settings, from simulated environments to real-world scenarios [32, 67, 75]. While these methods can learn effective policies in controlled environments [27, 48, 80], they often perform suboptimally in real-world scenarios due to factors like partial observability, dynamic changes, and varying levels of uncertainty [8, 40]. Additionally, standard RL typically does not account for uncertainty explicitly, which can hinder performance in complex environments where uncertainty plays a significant role.

Task-Specific Navigation Policies: Recent advances have focused on developing navigation policies that are conditioned on specific tasks or goals [10, 38, 42, 52]. Task-conditioned RL allows agents to adapt their behavior based on high-level task specifications, improving flexibility and generalization. In navigation, this has been explored by training policies capable of handling objectives, adjusting the robot's behavior according to the task [43, 68, 78]. These approaches often neglect the varying importance of uncertainty management [9, 26] across different tasks and environments.

Uncertainty-Aware Reinforcement Learning in Navigation: Incorporating uncertainty into RL for navigation has been investigated to enhance exploration, robustness, and safety [62, 76]. Some approaches introduce uncertainty penalization directly into the reward function, encouraging agents to avoid high-uncertainty regions [14, 34, 77]. Others utilize Bayesian RL methods to model uncertainty in value estimation, improving decision-making under uncertainty [1, 3]. Bootstrapped ensembles maintain multiple value function estimates to capture epistemic uncertainty, leading to more informed exploration strategies [2, 29, 45, 72]. While these methods consider uncertainty, they often do so globally or uniformly [31, 46, 50] across the environment and do not tailor the uncertainty management to specific navigation tasks or spatial regions.

Risk-Aware Planning and Navigation: Risk-aware planning introduces measures to balance performance and safety by considering the potential risks associated with different actions [18, 33]. These techniques [44, 74] enable robots to make decisions that account for uncertainty in the environment [22]. In the context of RL, risk-sensitive approaches adjust the policy to avoid high-risk actions, often through modified reward functions or policy constraints [6, 37, 64]. Although effective in managing risk, these methods typically apply uniform risk thresholds and do not adapt to the task-specific uncertainty requirements that may vary across different navigation scenarios.

Task-Specific Uncertainty Management in Navigation: Integrating high-level task specifications into uncertainty management for navigation remains an open challenge. Some recent works propose large pre-trained models to interpret complex instructions

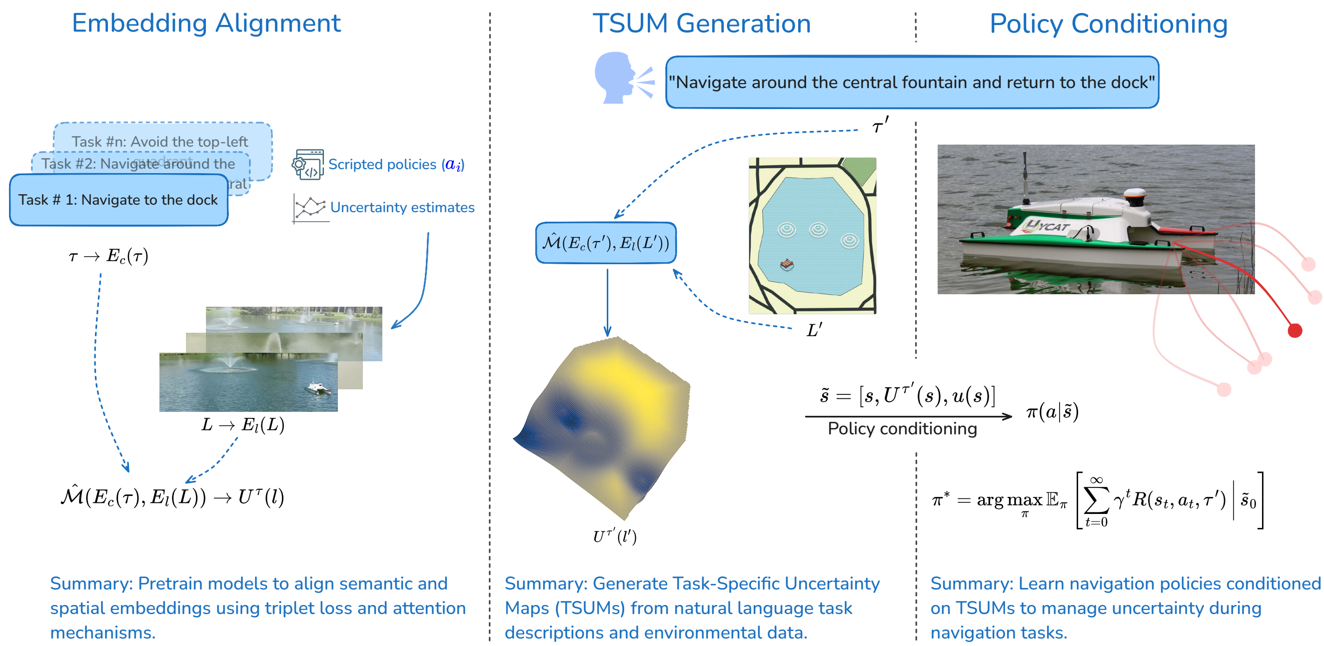

Figure 2: Overview of GUIDE. The framework consists of two phases: Embedding Alignment (Pretraining) and Deployment. In the embedding alignment phase, semantic and spatial embeddings are learned and aligned using triplet loss and attention mechanisms. During deployment, TSUMs are generated from natural language task descriptions and environmental data, and navigation policies are conditioned on these TSUMs to manage uncertainty in a task-aware manner.

and generate corresponding navigation behaviors [4, 21, 47, 58–61]. While these approaches enable robots to understand and execute a wider range of tasks, they often lack a systematic method for representing and integrating task-specific uncertainty requirements into the navigation policy. This limitation reduces their effectiveness in complex environments where the importance of uncertainty can vary significantly across different regions.

In contrast to these prior works, our proposed GUIDE framework systematically and explicitly incorporates task-specific uncertainty requirements into both the planning and learning processes for navigation. By addressing the limitations of existing methods by treating uncertainty in a task-specific manner, our approach enables robots to dynamically adjust their navigation policies according to the varying importance of uncertainty.

3 THE GUIDE FRAMEWORK

Consider a robot operating in a continuous state space S, where each state $s \in S$ represents its pose. It can execute actions $a \in A$ from a continuous action space A. Assigned a navigation task τ specified by a natural language description, the robot must achieve objectives and adhere to constraints related to features of the environment. Our goal is to develop a navigation policy $\pi(a|s)$ that efficiently accomplishes the task while managing the robot's state estimation uncertainty $u(s)$ in a task-aware manner. We aim to address the following problems using GUIDE:

(i) Interpret the task τ to determine acceptable levels of uncertainty across different regions of the environment, reflecting the relative importance of uncertainty reduction as dictated by the task requirements.
(ii) Synthesize a navigation policy that integrates this task-specific uncertainty information into the decision-making process. The policy should guide the robot to manage its uncertainty appropriately, prioritizing uncertainty reduction where necessary to ensure successful task execution, while efficiently pursuing task objectives in other areas.

Figure 2 presents an overview of our approach. The following sections detail the main components of GUIDE along with their implementation details.

3.1 Task-Specific Uncertainty Map (TSUM)

To effectively integrate task-specific uncertainty considerations into robotic navigation, we introduce the concept of *Task-Specific Uncertainty Maps* (TSUMs). A TSUM is a function $U^\tau : L \to \mathbb{R}^+$ that assigns an acceptable level of state estimation uncertainty to each location $l \in L$ for a given navigation task τ.

Formally, the TSUM $U^\tau(l)$ is defined as:

$$U^\tau(l) = \psi\left(\Phi^\tau(l), C^\tau(l), \mathcal{E}(l)\right), \tag{1}$$

where:

Task Relevance Function $\Phi^\tau(l)$*:* This function quantifies the importance of location l for the successful completion of task τ. It measures how critical precise navigation and state estimation are at l to fulfill the task objectives.

Task-Specific Constraints $C^\tau(l)$*:* These constraints represent conditions that must be satisfied at location l for task τ. They may include requirements such as avoiding restricted zones, operating within designated perimeters, or adhering to specific operational parameters unique to the task.

Environmental Factors $\mathcal{E}(l)$: This component captures static properties of the environment at location l, such as the presence of obstacles, terrain difficulty, or regions known to be unsafe or infeasible for navigation.

The function ψ combines these components into a scalar value $U^\tau(l)$, representing the acceptable level of uncertainty at location l for task τ. We define ψ as a weighted sum:

$$U^\tau(l) = w_\Phi \Phi^\tau(l) + w_C C^\tau(l) + w_\mathcal{E} \mathcal{E}(l), \tag{2}$$

where w_Φ, w_C, and $w_\mathcal{E}$ are weighting coefficients that determine the relative influence of each component.

We now present a method to generate TSUM from any given task description.

3.1.1 Implementation Details. To generate the Task-Specific Uncertainty Map (TSUM) $U^\tau(l)$, we combine task-specific semantics with spatial context.

Task Description Processing. Similar to the work in [23], we extract m subtasks $\{t_i\}_{i=1}^m$ and c constraints $\{c_k\}_{k=1}^c$ from the natural language task description τ using dependency parsing and Named Entity Recognition (NER).

Semantic and Spatial Embeddings. Subtasks and constraints are mapped into an n-dimensional semantic embedding space using the function $E_c : C \to \mathbb{R}^n$, implemented with a fine-tuned RoBERTa model [36] trained on navigation tasks:

$$E_c(t_i),\ E_c(c_k) \in \mathbb{R}^n. \tag{3}$$

Spatial locations $l \in L$ are characterized by their coordinates and environmental features. These features are processed by a neural network f_{spatial} to produce spatial embeddings $E_l(l) \in \mathbb{R}^n$:

$$E_l(l) = f_{\text{spatial}}(\text{features}(l)). \tag{4}$$

The network f_{spatial} consists of two hidden layers with 128 and 64 units and ReLU activations.

Computing $\Phi^\tau(l)$ and $C^\tau(l)$. Attention weights α_i and β_k are assigned to subtasks and constraints using hierarchical attention networks [71]:

$$\alpha_i = \frac{\exp(u_i)}{\sum_{j=1}^m \exp(u_j)}, \quad u_i = v^\top \tanh(W E_c(t_i) + b), \tag{5}$$

$$\beta_k = \frac{\exp(u'_k)}{\sum_{k'=1}^c \exp(u'_{k'})}, \quad u'_k = v'^\top \tanh(W' E_c(c_k) + b'), \tag{6}$$

where $W \in \mathbb{R}^{h \times n}$, $v \in \mathbb{R}^h$, $b \in \mathbb{R}^h$, h is the hidden layer size and $W' \in \mathbb{R}^{h' \times n}$, $v' \in \mathbb{R}^{h'}$, $b' \in \mathbb{R}^{h'}$. The task relevance function $\Phi^\tau(l)$ and the task-specific constraints function $C^\tau(l)$ are computed as:

$$\Phi^\tau(l) = \sum_{i=1}^m \alpha_i \cdot S_c\left(E_c(t_i),\ E_l(l)\right), \tag{7}$$

$$C^\tau(l) = \sum_{k=1}^c \beta_k \cdot S_c\left(E_c(c_k),\ E_l(l)\right), \tag{8}$$

where $S_c(\cdot,\cdot)$ denotes the cosine similarity.

Environmental Factors $\mathcal{E}(l)$. Environmental factors are incorporated using a function f_{env}:

$$\mathcal{E}(l) = f_{\text{env}}(\text{env\_features}(l)) = w_{\text{env}}^\top \text{env\_features}(l) + b_{\text{env}}, \tag{9}$$

where env_features(l) include static properties such as obstacle presence, and w_{env}, b_{env} are learned parameters.

Aggregation into TSUM. The acceptable level of uncertainty at each location l is determined by aggregating the computed functions using Equation (1).

3.1.2 Training the TSUM Generation Model. The embedding functions and attention networks are trained using a dataset of paired semantic concepts and spatial locations with known relationships[1].

Triplet Loss for Embedding Alignment. To align the semantic and spatial embeddings, we employ a triplet loss function [55]:

$$L_{\text{triplet}} = \sum_{i=1}^N \left[\|E_c(t_i) - E_l(l_i^+)\|^2 - \|E_c(t_i) - E_l(l_i^-)\|^2 + \delta \right]_+, \tag{10}$$

where N is the number of training samples, l_i^+ is a location relevant to subtask t_i, l_i^- is an irrelevant location, δ is a margin parameter, and $[x]_+ = \max(0, x)$.

This loss encourages the model to bring embeddings of related subtasks and locations closer while separating unrelated pairs.

Attention Network Training. The attention networks are trained by minimizing the Kullback-Leibler divergence [28] between the predicted attention distributions and the ground truth distributions:

$$L_{\text{attention}} = \text{KL}\left(\alpha \parallel \alpha^{\text{gt}}\right) + \text{KL}\left(\beta \parallel \beta^{\text{gt}}\right), \tag{11}$$

where α^{gt} and β^{gt} are the ground truth attention weights.

Total Loss Function. The total loss for training is:

$$L = L_{\text{triplet}} + \lambda L_{\text{attention}}, \tag{12}$$

with λ being a hyperparameter balancing the two loss components.

3.2 Conditioning Navigation Policies Using Task-Specific Uncertainty Maps

To enable robots to manage uncertainty in a task-aware manner, we propose conditioning the navigation policy on the Task-Specific Uncertainty Map (TSUM). Our objective is to learn a policy π that allows the robot to efficiently accomplish the task τ while adhering to the uncertainty requirements specified by the TSUM.

To condition the policy, we augment the state representation to include both the acceptable uncertainty levels from the TSUM and the robot's current estimation of its state uncertainty. For a given task τ, the augmented state $\tilde{s}$ is defined as:

$$\tilde{s} = [s, U^\tau(s), u(s)], \tag{13}$$

where s is the original state representing the robot's observation of the environment, $U^\tau(s)$ is the acceptable uncertainty level at state s derived from the TSUM, and $u(s)$ is the robot's current state estimation uncertainty.

[1]The dataset was collected using an Autonomous Surface Vehicle (ASV) in a controlled environment. Details are provided in the supplementary. The dataset will be open-sourced after the review period.

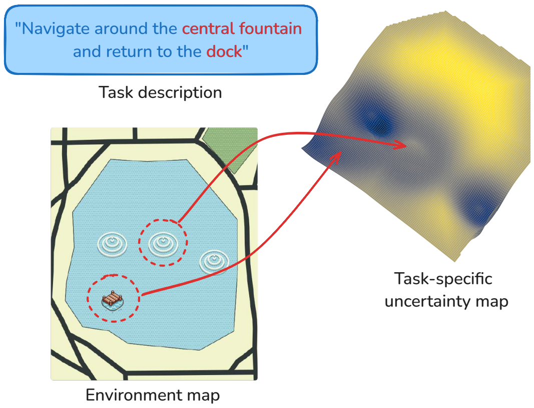

Figure 3: Example task and its TSUM, where darker areas indicate a higher need for certainty and lighter areas signify a lower need for certainty.

By conditioning the policy $\pi(a|\tilde{s})$ on the augmented state $\tilde{s}$, the agent makes decisions based not only on the environmental state but also on the acceptable, actual and future uncertainty levels at each location. This allows the agent to adjust its actions to meet the task-specific uncertainty requirements.

The reinforcement learning problem is formulated in the augmented state space. The objective is to learn an optimal policy $\pi^*(a|\tilde{s})$ that maximizes the expected cumulative reward:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, \tau) \,\middle|\, \tilde{s}_0\right], \tag{14}$$

where $\gamma \in [0, 1)$ is the discount factor, and $R(s_t, a_t, \tau)$ is the task-specific reward function.

Standard reinforcement learning algorithms can be employed to solve this optimization problem in the augmented state space. In the next section, we show how a standard RL algorithm can be adapted using our framework.

3.2.1 GUIDEd Soft Actor-Critic Algorithm. While any reinforcement learning algorithm can be used to learn the optimal policy $\pi^*(a|\tilde{s})$, we adopt the Soft Actor-Critic (SAC) algorithm [17] due to its sample efficiency and robustness. Our adapted version, referred to as *GUIDEd SAC*, integrates TSUMs by conditioning the policy and value functions on the augmented state $\tilde{s}$.

In the SAC framework, the objective is to maximize the expected cumulative reward augmented by an entropy term, which encourages exploration:

$$J(\pi) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t \left(R(s_t, a_t, \tau) + \alpha \mathcal{H}(\pi(\cdot|\tilde{s}_t))\right) \,\middle|\, \tilde{s}_0\right], \tag{15}$$

where $\mathcal{H}(\pi(\cdot|\tilde{s}_t)) = -\mathbb{E}_{a_t\sim\pi(\cdot|\tilde{s}_t)}[\log \pi(a_t|\tilde{s}_t)]$ is the entropy of the policy at state $\tilde{s}_t$, and α is the temperature parameter balancing exploration and exploitation.

GUIDEd SAC maintains parameterized function approximators for the policy $\pi_\theta(a|\tilde{s})$ and the soft Q-value functions $Q_{\phi_1}(\tilde{s}, a)$ and $Q_{\phi_2}(\tilde{s}, a)$, where θ and ϕ_i denote the parameters of the policy and value networks, respectively.

The soft Q-value networks $Q_{\phi_i}(\tilde{s}, a)$ are updated by minimizing the soft Bellman residual:

$$\mathcal{L}_Q(\phi_i) = \mathbb{E}_{(\tilde{s}_t, a_t, r_t, \tilde{s}_{t+1})\sim\mathcal{D}}\left[\left(Q_{\phi_i}(\tilde{s}_t, a_t) - y_t\right)^2\right], \tag{16}$$

where $\mathcal{D}$ is the replay buffer, and the target value y_t is computed as:

$$y_t = r_t + \gamma\left(\min_{i=1,2} Q_{\bar{\phi}_i}(\tilde{s}_{t+1}, a_{t+1}) - \alpha \log \pi_\theta(a_{t+1}|\tilde{s}_{t+1})\right), \tag{17}$$

with $a_{t+1} \sim \pi_\theta(\cdot|\tilde{s}_{t+1})$ and $Q_{\bar{\phi}_i}$ being the target Q-value networks with delayed parameters for stability.

The policy network $\pi_\theta(a|\tilde{s})$ is updated by minimizing:

$$\mathcal{L}_\pi(\theta) = \mathbb{E}_{\tilde{s}_t\sim\mathcal{D}}\left[\mathbb{E}_{a_t\sim\pi_\theta(\cdot|\tilde{s}_t)}\left[\alpha \log \pi_\theta(a_t|\tilde{s}_t) - Q_\phi(\tilde{s}_t, a_t)\right]\right], \tag{18}$$

where $Q_\phi(\tilde{s}_t, a_t) = \min_{i=1,2} Q_{\phi_i}(\tilde{s}_t, a_t)$.

The temperature parameter α is adjusted by minimizing:

$$\mathcal{L}(\alpha) = \mathbb{E}_{a_t\sim\pi_\theta(\cdot|\tilde{s}_t)}\left[-\alpha\left(\log \pi_\theta(a_t|\tilde{s}_t) + \bar{\mathcal{H}}\right)\right], \tag{19}$$

where $\bar{\mathcal{H}}$ is the target entropy. Algorithm 1 summarizes the GUIDEd SAC algorithm.

Algorithm 1 GUIDEd SAC Algorithm

Initialize policy network $\pi_\theta(a|\tilde{s})$, Q-value networks Q_{ϕ_1}, Q_{ϕ_2}, target Q-value networks $Q_{\bar{\phi}_1}, Q_{\bar{\phi}_2}$, temperature parameter α, and replay buffer $\mathcal{D}$.
for each environment interaction step **do**
    Obtain current state s_t, acceptable uncertainty $U^\tau(s_t)$, current uncertainty $u(s_t)$.
    Form augmented state $\tilde{s}_t = [s_t, U^\tau(s_t), u(s_t)]$.
    Sample action $a_t \sim \pi_\theta(\cdot|\tilde{s}_t)$.
    Execute action a_t, observe reward r_t and next state s_{t+1}.
    Compute $\tilde{s}_{t+1} = [s_{t+1}, U^\tau(s_{t+1}), u(s_{t+1})]$.
    Store transition $(\tilde{s}_t, a_t, r_t, \tilde{s}_{t+1})$ in replay buffer $\mathcal{D}$.
end for
for each gradient step **do**
    Sample minibatch of transitions from $\mathcal{D}$.
    Update Q-value networks Q_{ϕ_i} by minimizing $\mathcal{L}_Q(\phi_i)$ (Equation (16)).
    Update policy network π_θ by minimizing $\mathcal{L}_\pi(\theta)$ (Equation (18)).
    Adjust temperature parameter α by minimizing $\mathcal{L}(\alpha)$ (Equation (19)).
    Update target Q-value networks: $\bar{\phi}_i \leftarrow \tau\phi_i + (1-\tau)\bar{\phi}_i$.
end for

4 EXPERIMENTS

To evaluate the effectiveness of GUIDE, we conduct experiments in a real-world, real-time scenario using an autonomous surface vehicle (ASV). The ASV is tasked with navigating through a GPS-constrained environment, where precise localization comes at a cost. The agent operates with noisy position estimates by default and can request GPS readings when needed, reducing the uncertainty but at the cost of a higher penalty for each request.

The experimental setup is designed to mimic realistic operational constraints [51, 73], highlighting the core challenge: balancing task success with effective uncertainty management. Excessive reliance on noisy localization risks failure in critical areas, while frequent use of high-precision GPS leads to increased mission costs.

4.1 Experimental Setup

4.1.1 Environment. The ASV operates in an open lake characterized by environmental variability and human-induced disturbances. The environment features fountains that serve as obstacles and introduce additional layers of uncertainty by generating water disturbances that create both physical challenges for navigation and contribute to the overall uncertainty in state estimation.

4.1.2 Autonomous Surface Vehicle Setup. Our experiments utilize the SeaRobotics Surveyor Autonomous Surface Vehicle (ASV), which is equipped with a Global Positioning System (GPS), an Inertial Measurement Unit (IMU), and a YSI EXO2 multiparameter water quality sonde. The GPS provides precise positioning, while the IMU assists with motion tracking and orientation. The YSI EXO2 sonde, primarily used for environmental monitoring, contributes to the overall state estimation by providing additional context.

The action space of the ASV is defined as $a = (\lambda, \alpha, \eta)$, where $\lambda \in [0, \lambda_{\max}]$ represents the propulsion torque (with $\lambda_{\max}$ being the maximum allowed torque), $\alpha \in [0, 2\pi)$ denotes the steering angle, and η is a discrete variable indicating the mode of position estimation. The ASV has a maximum speed of 2 knots.

4.1.3 Position Estimation Modes. To represent the operational challenges of managing uncertainty during navigation, we model two modes of position estimation for the ASV [51]:

(i) **Noisy Position Estimation**: By default, the ASV estimates its position using the IMU and YSI EXO2 sensors, which results in a noisy and less accurate position estimate [54]. This mode represents the low-cost but high-uncertainty estimate.

(ii) **Exact Position Estimation**: The ASV can request exact position data from GPS. This incurs a higher resource cost, leading to a reduced overall mission reward.

The reward structure is designed to encourage task completion while balancing the efficient management of positional uncertainty, without biasing the results in favor of GUIDE compared to the baselines. The reward function for task execution, r_{task}, incentivizes mission completion, while each use of exact position estimation incurs a penalty c_{exact} to reflect its associated cost. Conversely, noisy estimation incurs a smaller penalty c_{noisy}, encouraging the ASV to balance accuracy needs with operational efficiency.

4.1.4 Evaluation Protocol. The agent receives a natural language task specification, describing a navigation task using key features of the environment[2]. Task completion is evaluated based on specific criteria for each task type. For goal-reaching tasks, the agent must reach within a 1.5-meter radius of the target. For perimeter or exploration tasks, it must remain within a 3.5-meter corridor of the path, and for avoid tasks, a 3-meter minimum distance from designated areas is required. Task completion rate (TCR) is determined by the proportion of the task completed, and rewards are based on the total reward earned at task completion. All baselines interpret instructions through the same semantic parser, converting them into structured representations suitable for each algorithm.

[2]Refer to Table 1 for details about task types

4.2 Baselines and Ablations

We compare GUIDE against several baselines and conduct ablation studies.

4.2.1 Ablation Studies. We perform two ablations:

(i) *Standard RL without TSUMs (SAC, Ablation 1)*: We train SAC agents using only the original state s, without TSUMs, to assess the importance of TSUMs in managing uncertainty.

(ii) *GUIDEd PPO (G-PPO, Ablation 2)*: We implement GUIDEd PPO [56], incorporating TSUMs and agent uncertainty into the state representation, to examine the effect of the underlying RL algorithm on GUIDE's performance.

4.2.2 Baselines. We compare GUIDE against several baselines: *SAC-Based Methods.* We include SAC variants that handle uncertainty differently:

(i) *RL with Uncertainty Penalization (SAC-P)*: We modify the reward function in SAC to penalize high uncertainty: $R_{\text{SAC-P}} = R_{\text{base}} - \zeta u(s)$, where R_{base} includes task rewards and localization costs, ζ is a weighting factor, and $u(s)$ is the agent's current uncertainty. This tests traditional reward shaping versus GUIDE's approach.

(ii) *Bootstrapped Uncertainty-Aware RL (B-SAC)*: We implement Bootstrapped SAC [30], training multiple value networks to estimate epistemic uncertainty, which guides exploration and decision-making.

(iii) *Heuristic Policy (HEU)*: A hand-crafted policy where the agent plans using SAC but switches to exact position estimation near obstacles or task-critical regions.

Other Methods. Additional baselines include:

(iv) *Risk-Aware RL (CVaR)*: We train agents using a CVaR-based RL algorithm [24], optimizing a risk-sensitive objective to minimize potential high-cost outcomes, focusing on worst-case scenarios without task-specific uncertainty levels.

(v) *Uncertainty-Aware Motion Planning (RAA)*: We employ Risk-Aware A* [20] to plan paths that minimize collision probabilities under uncertainty, aiming for overall uncertainty minimization without considering task-specific requirements.

4.3 Results

The performance of GUIDE compared to various baseline methods is summarized in Table 1. Our results demonstrate that GUIDE consistently outperforms all baselines across all task categories. Notably, both GUIDEd Actor-Critic (G-SAC) and GUIDEd PPO (G-PPO) outperform the baselines; however, G-SAC generally achieves higher task completion rates and rewards.

4.3.1 Analysis of Ablation Studies.

Impact of TSUMs (Ablation 1). To assess the significance of incorporating Task-Specific Uncertainty Maps (TSUMs), we compare the standard Soft Actor-Critic (SAC) without TSUMs to GUIDEd Actor-Critic (G-SAC). As shown in Table 1, conditioning on TSUMs significantly enhances performance across all tasks. Without TSUMs,

GUIDEd SAC **SAC** **GUIDEd PPO**

Figure 4: Comparison of navigation trajectories on the task: `Start at the dock, navigate around the central fountain, then around the left fountain, and finally around the right fountain`. The white line represents the ASV's trajectory, with red points marking locations where it reduced uncertainty to meet task-specific requirements. Pink diamonds indicate collision points, the green rectangle marks the starting dock, and the TSUMs for GUIDEd agents are shown above their respective figures.

Tasks	Metric	SAC	SAC-P	B-SAC	CVaR	RAA	HEU	G-PPO	G-SAC
Goal reaching: waypoint	TCR (%) (↑)	67.2%	82.1%	75.9%	68.8%	35.3%	71.3%	83.8%	95.1%
`visit [coordinate]`	Reward (avg) (↑)	186.2	260.4	249.5	184.8	26.9	176.1	319.0	425.5
Goal reaching: context	TCR (%) (↑)	68.9%	84.3%	74.2%	66.8%	32.7%	73.5%	81.7%	92.0%
`navigate to dock`	Reward (avg) (↑)	144.1	241.8	189.2	134.6	53.5	137.5	308.3	410.4
Avoid tasks	TCR (%) (↑)	71.3%	83.2%	79.6%	78.4%	51.3%	62.4%	89.6%	88.7%
`avoid the central fountain`	Reward (avg) (↑)	177.8	199.2	277.6	220.4	107.8	174.4	437.6	482.2
Perimeter tasks	TCR (%) (↑)	44.3%	51.6%	56.3%	41.6%	39.8%	49.6%	74.9%	83.7%
`go around the left fountain`	Reward (avg) (↑)	84.4	132.8	170.4	32.8	111.2	146.8	449.2	594.6
Explore tasks	TCR (%) (↑)	88.6%	92.4%	87.6%	84.9%	71.3%	88.7%	93.8%	97.5%
`explore top-right quadrant`	Reward (avg) (↑)	486	474	526	449	402	537	548	595
Restricted areas	TCR (%) (↑)	70.8%	82.1%	78.4%	71.2%	51.1%	63.9%	90.0%	88.7%
`visit dock while avoiding top-right quadrant`	Reward (avg) (↑)	266.4	306.8	307.2	269.6	201.4	261.2	570.0	634.6
Multi-goal tasks	TCR (%) (↑)	31.3%	42.9%	37.7%	30.9%	19.5%	42.1%	72.8%	81.7%
`Combination of tasks (see Fig. 3)`	Reward (avg) (↑)	124.4	135.2	122.4	129.2	100.4	155.2	423.6	511.4

Table 1: Performance comparison between GUIDE and baseline methods on Task Completion Rate (TCR) and average cumulative reward, across different task categories. Results are averaged over 50 experiments per task category. Example tasks are provided below each task category to illustrate the types of navigation challenges evaluated.

SAC cannot manage positional uncertainty in a task-specific manner, leading to suboptimal navigation decisions and lower rewards. To further illustrate the effect of TSUM integration, we present in Figure 4 the trajectories of agents with and without TSUMs on a representative task. We observe that SAC often overuses high-precision localization in areas where it is unnecessary, incurring additional costs without significant benefits. Conversely, in critical regions requiring precise navigation, SAC fails to reduce uncertainty appropriately, leading to collisions with the fountains. In contrast, the G-SAC agent effectively uses TSUMs to adapt its uncertainty management, switching to high-precision localization in areas where the TSUM indicates low acceptable uncertainty. This enables the agent to navigate safely around obstacles and complete tasks more efficiently.

Effect of RL Algorithm Choice (Ablation 2). We also investigate the impact of the underlying reinforcement learning algorithm by comparing GUIDEd PPO (G-PPO) to G-SAC, both of which integrate TSUMs but differ in their RL methodologies. This difference in the performance of G-SAC and G-PPO can be attributed to several factors inherent to the algorithms. G-SAC, based on the SAC framework, is an off-policy method that leverages entropy regularization to encourage exploration while maintaining stability. Its off-policy nature allows for more efficient sample utilization, which is particularly beneficial in continuous action spaces and when data collection is costly or limited. In contrast, PPO is an on-policy algorithm that relies on proximal updates to prevent large policy shifts, using a clipped objective function. While PPO is stable, it can be less sample-efficient [35], as it requires new data for each policy update and may not explore as effectively in complex environments.

Our empirical results suggest that the off-policy efficiency and exploration capabilities of G-SAC make it better suited for navigation tasks. Furthermore, the entropy term in G-SAC encourages the agent to consider a wider range of actions, enabling it to discover more optimal strategies for managing uncertainty in a task-specific context.

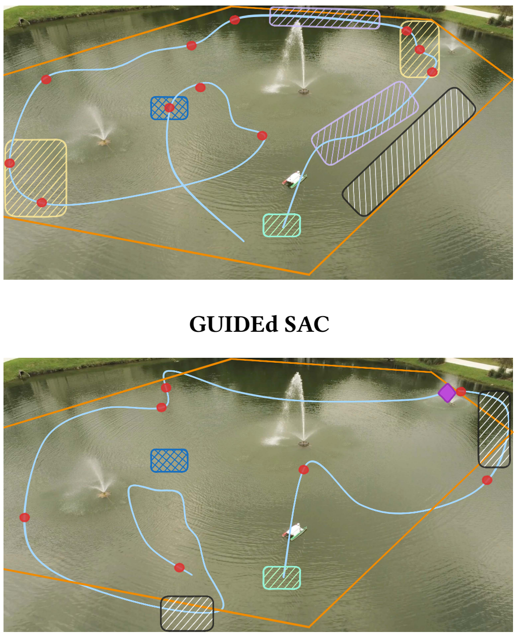

Figure 5: Comparison of navigation trajectories between GUIDEd SAC and SAC-P agents on the task: `Start and end at the dock. Go around the perimeter of the area and visit coordinates [redacted for paper anonymity].` The light blue line represents the trajectory taken by the agent. Red dots indicate locations where the agent actively reduced its state estimation uncertainty. The green rectangle denotes the dock, blue rectangle marks the specific coordinates the agent is instructed to visit.

4.3.2 Comparison with Baselines. GUIDE consistently outperforms all baselines across task types in both task completion rate and average reward (see Table 1). Standard RL methods like SAC and SAC-P lack task-specific uncertainty management. SAC-P penalizes high uncertainty uniformly, resulting in overly conservative behavior where precision is unnecessary and insufficient caution in critical regions, ultimately leading to lower performance. B-SAC estimates epistemic uncertainty but fails to adapt to task-specific requirements, leading to inefficient exploration. Risk-Aware RL methods like CVaR are uniformly risk-averse, missing opportunities for calculated risk-taking that could improve task success. RAA aims to minimize overall uncertainty without considering task context, often generating inefficient paths. The Heuristic Policy (HEU) switches to exact localization near obstacles, but lacks GUIDE's adaptability, failing to adjust to sudden changes in uncertainty requirements.

4.3.3 Behavior of GUIDEd Agents. We observed distinct behaviors in GUIDEd agents across various tasks. Analyzing the specific task shown in Figure 5, GUIDEd agents strategically adjust their reliance on precise position estimation versus noisier estimates. In areas where the TSUMs indicate high precision is necessary — such as navigating near obstacles or close to the perimeters (highlighted in yellow) — the agents opt for exact positioning despite the higher operational cost. Conversely, in less critical regions (shown in purple), they rely on less precise, noisy estimates. This adaptability allows GUIDEd agents to manage uncertainty more efficiently than baselines, resulting in faster completion times and smoother trajectories. Although not perfect — occasionally missing sections of the perimeter (indicated by black shaded regions in Figure 5) — GUIDEd agents significantly outperform baselines like SAC-P with engineered rewards. Baseline methods, lacking adaptive uncertainty management, frequently fail to complete tasks safely. They often collide with obstacles (Figure 5 pink diamond) or take inefficient paths (Figure 5 black region).

5 CONCLUSION

In this paper, we introduced GUIDE, a framework that integrates task-specific uncertainty requirements into robotic navigation policies. Central to our approach is the concept of Task-Specific Uncertainty Maps (TSUMs), which represent acceptable levels of state estimation uncertainty across different regions of the environment based on the given task. By conditioning navigation policies on TSUMs, we enable robots to reason about the context-dependent importance of certainty and adapt their behavior accordingly.

We demonstrated how GUIDE can be incorporated into reinforcement learning frameworks by augmenting the state representation to include both the acceptable uncertainty levels from TSUMs and the robot's current state estimation uncertainty. Specifically, we adapted the Soft Actor-Critic (SAC) algorithm to operate in this augmented state space, resulting in the GUIDEd SAC algorithm. Our extensive experiments on a variety of real-world robotic navigation tasks showed that policies conditioned on TSUMs significantly improve task completion rates and overall performance compared to baseline methods that do not explicitly consider task-specific uncertainty. The results indicate that incorporating task-specific uncertainty requirements allows robots to effectively balance task objectives with appropriate uncertainty management. GUIDE enhances navigation strategies by enabling robots to focus their uncertainty reduction efforts where it is most critical for task success, leading to improved efficiency.

GUIDE is designed to be flexible and can be applied to various robotic platforms and tasks. Future work includes extending the framework to handle dynamic environments where uncertainty requirements may change over time.

REFERENCES

[1] Mohammad Alali and Mahdi Imani. 2024. Bayesian reinforcement learning for navigation planning in unknown environments. *Frontiers in Artificial Intelligence* (2024).

[2] Chenjia Bai, Lingxiao Wang, Zhuoran Yang, Zhihong Deng, Animesh Garg, Peng Liu, and Zhaoran Wang. 2022. Pessimistic bootstrapping for uncertainty-driven offline reinforcement learning. *arXiv preprint arXiv:2202.11566* (2022).

[3] Matthew Budd, Paul Duckworth, Nick Hawes, and Bruno Lacerda. 2023. Bayesian reinforcement learning for single-episode missions in partially unknown environments. In *Conference on Robot Learning*.

[4] Matthew Chang, Theophile Gervet, Mukul Khanna, Sriram Yenamandra, Dhruv Shah, So Yeon Min, Kavit Shah, Chris Paxton, Saurabh Gupta, Dhruv Batra, et al. 2023. GOAT: Go to any thing. *arXiv preprint arXiv:2311.06430* (2023).

[5] Tao Chen, Saurabh Gupta, and Abhinav Gupta. 2019. Learning exploration policies for navigation. *arXiv preprint arXiv:1903.01959* (2019).

[6] Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. 2018. Risk-constrained reinforcement learning with percentile risk criteria. *Journal of Machine Learning Research* (2018).

[7] Gerald Cook and Feitian Zhang. 2020. *Mobile robots: Navigation, control and sensing, surface robots and AUVs.* John Wiley & Sons.

[8] Aidan Curtis, George Matheos, Nishad Gothoskar, Vikash Mansinghka, Joshua Tenenbaum, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. 2024. Partially Observable Task and Motion Planning with Uncertainty and Risk Awareness. *arXiv preprint arXiv:2403.10454* (2024).

[9] Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. 2018. Automatic goal generation for reinforcement learning agents. In *International conference on machine learning*.

[10] Sébastien Forestier, Rémy Portelas, Yoan Mollard, and Pierre-Yves Oudeyer. 2022. Intrinsically motivated goal exploration processes with automatic curriculum learning. *Journal of Machine Learning Research* (2022).

[11] Dieter Fox, Sebastian Thrun, Wolfram Burgard, and Frank Dellaert. 2001. Particle filters for mobile robot localization. In *Sequential Monte Carlo methods in practice.* Springer.

[12] Alejandro Gonzalez-Garcia and Herman Castañeda. 2021. Guidance and control based on adaptive sliding mode strategy for a USV subject to uncertainties. *IEEE Journal of Oceanic Engineering* (2021).

[13] Faiza Gul, Wan Rahiman, and Syed Sahal Nazli Alhady. 2019. A comprehensive study for robot navigation techniques. *Cogent Engineering* (2019).

[14] Siyu Guo, Xiuguo Zhang, Yiquan Du, Yisong Zheng, and Zhiying Cao. 2021. Path planning of coastal ships based on optimized DQN reward function. *Journal of Marine Science and Engineering* (2021).

[15] Fredrik Gustafsson, Fredrik Gunnarsson, Niclas Bergman, Urban Forssell, Jonas Jansson, Rickard Karlsson, and P-J Nordlund. 2002. Particle filters for positioning, navigation, and tracking. *IEEE Transactions on signal processing* (2002).

[16] Fredrik Gustafsson, Fredrik Gunnarsson, Niclas Bergman, Urban Forssell, Jonas Jansson, Rickard Karlsson, and P-J Nordlund. 2002. Particle filters for positioning, navigation, and tracking. *IEEE Transactions on signal processing* (2002).

[17] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*.

[18] Astghik Hakobyan, Gyeong Chan Kim, and Insoon Yang. 2019. Risk-aware motion planning and control using CVaR-constrained optimization. *IEEE Robotics and Automation letters* (2019).

[19] Marc Hanheide, Moritz Göbelbecker, Graham S Horn, Andrzej Pronobis, Kristoffer Sjöö, Alper Aydemir, Patric Jensfelt, Charles Gretton, Richard Dearden, Miroslav Janicek, et al. 2017. Robot task planning and explanation in open and uncertain worlds. *Artificial Intelligence* (2017).

[20] Chuanhui Hu and Yan Jin. 2023. Long-Range Risk-Aware Path Planning for Autonomous Ships in Complex and Dynamic Environments. *Journal of Computing and Information Science in Engineering* (2023).

[21] Wenlong Huang, Fei Xia, Dhruv Shah, Danny Driess, Andy Zeng, Yao Lu, Pete Florence, Igor Mordatch, Sergey Levine, Karol Hausman, et al. 2023. Grounded decoding: Guiding text generation with grounded models for robot control. *arXiv preprint arXiv:2303.00855* (2023).

[22] Jeffrey Hudack. 2016. *Risk-aware planning for sensor data collection.* Ph.D. Dissertation. Syracuse University.

[23] Zhanming Jie, Aldrian Muis, and Wei Lu. 2017. Efficient dependency-guided named entity recognition. In *Proceedings of the AAAI conference on artificial intelligence*.

[24] Gregory Kahn, Adam Villaflor, Vitchyr Pong, Pieter Abbeel, and Sergey Levine. 2017. Uncertainty-aware reinforcement learning for collision avoidance. *arXiv preprint arXiv:1702.01182* (2017).

[25] Jonathan Ko and Dieter Fox. 2009. GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models. *Autonomous Robots* (2009).

[26] Henrik Kretzschmar, Markus Spies, Christoph Sprunk, and Wolfram Burgard. 2016. Socially compliant mobile robot navigation via inverse reinforcement learning. *The International Journal of Robotics Research* (2016).

[27] Jonáš Kulhánek, Erik Derner, and Robert Babuška. 2021. Visual navigation in real-world indoor environments using end-to-end deep reinforcement learning. *IEEE Robotics and Automation Letters* (2021).

[28] Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics* (1951).

[29] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. 2019. Stabilizing off-policy Q-Learning via bootstrapping error reduction. *Advances in neural information processing systems* (2019).

[30] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. 2019. Stabilizing off-policy Q-Learning via bootstrapping error reduction. *Advances in neural information processing systems* (2019).

[31] Charline Le Lan, Stephen Tu, Mark Rowland, Anna Harutyunyan, Rishabh Agarwal, Marc G Bellemare, and Will Dabney. 2023. Bootstrapped representations in reinforcement learning. *arXiv preprint arXiv:2306.10171* (2023).

[32] Min-Fan Ricky Lee and Sharfiden Hassen Yusuf. 2022. Mobile robot navigation using deep reinforcement learning. *Processes* (2022).

[33] Guofa Li, Yifan Yang, Shen Li, Xingda Qu, Nengchao Lyu, and Shengbo Eben Li. 2022. Decision making of autonomous vehicles in lane change scenarios: Deep reinforcement learning approaches with risk awareness. *Transportation research part C: emerging technologies* (2022).

[34] Yueming Li, Mingquan Ma, Jian Cao, Guobin Luo, Depeng Wang, and Weiqiang Chen. 2024. A Method for Multi-AUV Cooperative Area Search in Unknown Environment Based on Reinforcement Learning. *Journal of Marine Science and Engineering* (2024).

[35] Shijie Liu. 2024. An Evaluation of DDPG, TD3, SAC, and PPO: Deep Reinforcement Learning Algorithms for Controlling Continuous System. In *2023 International Conference on Data Science, Advanced Algorithm and Intelligent Computing (DAI 2023)*. Atlantis Press.

[36] Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[37] Björn Lütjens, Michael Everett, and Jonathan P How. 2019. Safe reinforcement learning with model uncertainty estimates. In *2019 International Conference on Robotics and Automation*.

[38] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. 2020. Learning latent plans from play. In *Conference on robot learning*.

[39] Christoforos Mavrogiannis, Francesca Baldini, Allan Wang, Dapeng Zhao, Pete Trautman, Aaron Steinfeld, and Jean Oh. 2023. Core challenges of social robot navigation: A survey. *ACM Transactions on Human-Robot Interaction* (2023).

[40] Steven Morad, Ryan Kortvelesy, Matteo Bettini, Stephan Liwicki, and Amanda Prorok. 2023. POPGym: Benchmarking partially observable reinforcement learning. *arXiv preprint arXiv:2303.01859* (2023).

[41] Kevin P Murphy. 1999. Bayesian map learning in dynamic environments. *Advances in neural information processing systems* (1999).

[42] Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. 2018. Visual reinforcement learning with imagined goals. *Advances in neural information processing systems* (2018).

[43] Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. 2018. Visual reinforcement learning with imagined goals. *Advances in neural information processing systems* (2018).

[44] Masahiro Ono, Thoams J Fuchs, Amanda Steffy, Mark Maimone, and Jeng Yen. 2015. Risk-aware planetary rover operation: Autonomous terrain classification and path planning. In *2015 IEEE aerospace conference*.

[45] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. 2016. Deep exploration via bootstrapped DQN. *Advances in neural information processing systems* (2016).

[46] Alvaro Ovalle and Simon M Lucas. 2020. Bootstrapped model learning and error correction for planning with uncertainty in model-based RL. In *2020 IEEE Conference on Games*.

[47] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. 2023. Open X-Embodiment: Robotic learning datasets and RT-X models. *arXiv preprint arXiv:2310.08864* (2023).

[48] Utsav Patel, Nithish K Sanjeev Kumar, Adarsh Jagan Sathyamoorthy, and Dinesh Manocha. 2021. DWA-RL: Dynamically feasible deep reinforcement learning policy for robot navigation among mobile obstacles. In *2021 IEEE International Conference on Robotics and Automation*.

[49] BK Patle, Anish Pandey, DRK Parhi, AJDT Jagadeesh, et al. 2019. A review: On path planning strategies for navigation of mobile robot. *Defence Technology* (2019).

[50] Niklas Paulig and Ostap Okhrin. 2024. Robust path following on rivers using bootstrapped reinforcement learning. *Ocean Engineering* (2024).

[51] Gokul Puthumanaillam, Manav Vora, and Melkior Ornik. 2024. ComTraQ-MPC: Meta-Trained DQN-MPC Integration for Trajectory Tracking with Limited Active Localization Updates. *arXiv preprint arXiv:2403.01564* (2024).

[52] Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. 2019. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*.

[53] Krishan Rana, Ben Talbot, Vibhavari Dasagi, Michael Milford, and Niko Sünderhauf. 2020. Residual reactive navigation: Combining classical and learned navigation strategies for deployment in unknown environments. In *2020 IEEE International Conference on Robotics and Automation*.
[54] Cesar A Rojas, Paulo Padrão, Jose Fuentes, Gregory M Reis, Arif R Albayrak, Batuhan Osmanoglu, and Leonardo Bobadilla. 2024. Combining multi-satellite remote and in-situ sensing for unmanned underwater vehicle state estimation. *Ocean Engineering* 310 (2024), 118708.
[55] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.
[56] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).
[57] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. 2018. Planning and decision-making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems* (2018).
[58] Dhruv Shah, Michael Robert Equi, Błażej Osiński, Fei Xia, Brian Ichter, and Sergey Levine. 2023. Navigation with large language models: Semantic guesswork as a heuristic for planning. In *Conference on Robot Learning*.
[59] Dhruv Shah, Błażej Osiński, Sergey Levine, et al. 2023. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *Conference on robot learning*.
[60] Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. 2023. GNM: A general navigation model to drive any robot. In *2023 IEEE International Conference on Robotics and Automation*.
[61] Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. 2023. ViNT: A foundation model for visual navigation. *arXiv preprint arXiv:2306.14846* (2023).
[62] Siddharth Singi, Zhanpeng He, Alvin Pan, Sandip Patel, Gunnar A Sigurdsson, Robinson Piramuthu, Shuran Song, and Matei Ciocarlie. 2024. Decision making for human-in-the-loop robotic agents via uncertainty-aware reinforcement learning. In *2024 IEEE International Conference on Robotics and Automation*.
[63] Matthijs TJ Spaan, Tiago S Veiga, and Pedro U Lima. 2015. Decision-theoretic planning under uncertainty with information rewards for active cooperative perception. *Autonomous Agents and Multi-Agent Systems* 29 (2015), 1157–1185.
[64] Krishnan Srinivasan, Benjamin Eysenbach, Sehoon Ha, Jie Tan, and Chelsea Finn. 2020. Learning to be safe: Deep RL with a safety critic. *arXiv preprint arXiv:2010.14603* (2020).
[65] Annalisa T Taylor, Thomas A Berrueta, and Todd D Murphey. 2021. Active learning in robotics: A review of control principles. *Mechatronics* (2021).
[66] Sebastian Thrun. 2002. Particle Filters in Robotics.. In *UAI*.
[67] Ning Wang, Yabiao Wang, Yuming Zhao, Yong Wang, and Zhigang Li. 2022. Sim-to-real: Mapless navigation for USVs using deep reinforcement learning. *Journal of Marine Science and Engineering* (2022).
[68] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. 2019. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.
[69] Zixiang Wang, Hao Yan, Zhuoyue Wang, Zhengjia Xu, Zhizhong Wu, and Yining Wang. 2024. Research on autonomous robots navigation based on reinforcement learning. In *3rd International Conference on Robotics, Artificial Intelligence and Intelligent Control*. IEEE, 78–81.
[70] Zifan Xu, Bo Liu, Xuesu Xiao, Anirudh Nair, and Peter Stone. 2023. Benchmarking reinforcement learning techniques for autonomous navigation. In *2023 IEEE International Conference on Robotics and Automation*.
[71] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*.
[72] Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. 2021. Improving sample efficiency in model-free reinforcement learning from images. In *Proceedings of the AAAI conference on artificial intelligence*.
[73] Kexiong Curtis Zeng, Shinan Liu, Yuanchao Shu, Dong Wang, Haoyu Li, Yanzhi Dou, Gang Wang, and Yaling Yang. 2018. All your GPS are belong to us: Towards stealthy manipulation of road navigation systems. In *27th USENIX security symposium (USENIX security 18)*.
[74] Yu Zhang, Sarath Sreedharan, Anagha Kulkarni, Tathagata Chakraborti, Hankz Hankui Zhuo, and Subbarao Kambhampati. 2016. Plan explicability for robot task planning. In *Proceedings of the RSS Workshop on Planning for Human-Robot Interaction: Shared Autonomy and Collaborative Robotics*.
[75] Zhiqian Zhou, Zhiwen Zeng, Lin Lang, Weijia Yao, Huimin Lu, Zhiqiang Zheng, and Zongtan Zhou. 2022. Navigating robots in dynamic environment with deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems* (2022).
[76] CY Zhu. 2023. Intelligent robot path planning and navigation based on reinforcement learning and adaptive control. *Journal of Logistics, Informatics and Service Science* (2023).
[77] Kai Zhu and Tao Zhang. 2021. Deep reinforcement learning based mobile robot navigation: A review. *Tsinghua Science and Technology* (2021).
[78] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. 2017. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation*.
[79] Yuanyang Zhu, Zhi Wang, Chunlin Chen, and Daoyi Dong. 2021. Rule-based reinforcement learning for efficient robot navigation with space reduction. *IEEE/ASME Transactions on Mechatronics* (2021).
[80] P Zieliński and Urszula Markowska-Kaczmar. 2021. 3D robotic navigation using a vision-based deep reinforcement learning model. *Applied Soft Computing* (2021).