Artificial Intelligence 319 (2023) 103903

Contents lists available at ScienceDirect

Artificial Intelligence

journal homepage: www.elsevier.com/locate/artint

Epistemic uncertainty aware semantic localization and mapping for inference and belief space planning [☆]

Vladimir Tchuiev [*], Vadim Indelman [1]

Department of Aerospace Engineering, Technion - Israel Institute of Technology, Haifa 32000, Israel

ARTICLE INFO

Article history:
Received 10 June 2021
Received in revised form 4 December 2022
Accepted 3 March 2023
Available online 9 March 2023

Keywords:
Object classification
SLAM
Active perception
Epistemic uncertainty

ABSTRACT

We investigate the problem of autonomous object classification and semantic SLAM, which generally exhibits a tight coupling between classification, metric SLAM, and planning under uncertainty. We contribute a unified framework for inference and belief space planning (BSP) that addresses prominent sources of uncertainty in this context: classification aliasing (classifier cannot distinguish between candidate classes from certain viewpoints), classifier epistemic uncertainty (classifier receives data “far” from its training set), and localization uncertainty (camera and object poses are uncertain). Specifically, two methods are developed for maintaining a joint distribution over robot and object poses, and over a posterior class probability vector that considers epistemic uncertainty in a Bayesian fashion. The first approach is Multi-Hybrid (MH), where multiple hybrid beliefs over poses and classes are maintained to approximate the joint belief over poses and posterior class probability. The second approach is Joint Lambda Pose (JLP), where the joint belief is maintained directly using a novel JLP factor. Furthermore, we extend both methods to BSP, while reasoning about future posterior epistemic uncertainty indirectly or directly via a novel information-theoretic reward function. Both inference methods utilize a novel viewpoint-dependent classifier uncertainty model that leverages the coupling between poses and classification scores and predicts the epistemic uncertainty from certain viewpoints. In addition, this model is used to generate predicted measurements during planning. To the best of our knowledge, this is the first work that reasons about classifier epistemic uncertainty within semantic SLAM and BSP. We extensively evaluate our inference and BSP approaches in simulation using real image data from the Active Vision Dataset. Results clearly indicate superior classification performance of our methods compared to an approach that is not epistemic uncertainty aware.

1. Introduction

Simultaneous localization and mapping (SLAM) is a fundamental problem in robotics and computer vision, with wide-reaching applications such as autonomous vehicles and UAVs, agriculture, medical, search and rescue, and more [1]. Specifi-

☆ This paper is part of the Special Issue: “Risk-aware Autonomous Systems: Theory and Practice”.
* Corresponding author.
E-mail addresses: vovatch@technion.ac.il (V. Tchuiev), vadim.indelman@technion.ac.il (V. Indelman).
1 This work was partially supported by US NSF/US-Israel BSF grant number 2019705, and by the Israel Ministry of Science & Technology (MOST) grant number 78557.

https://doi.org/10.1016/j.artint.2023.103903
0004-3702/© 2023 Elsevier B.V. All rights reserved.

cally, semantic SLAM, where a robot localizes itself and maps the environment using information from objects within it, is an actively researched field. For semantic SLAM, object classification is a crucial problem. With advances in recent years with deep-learning-based algorithms, classifiers today outperform humans in multiple classification tasks. However, classifiers are limited by their training and may provide unreliable results in different conditions, such as lighting, image resolution, and occlusions. In addition, the classifier may struggle to distinguish between different classes from certain viewpoints, resulting in classification aliasing. Faced with these uncertainties, classification scores may appear sporadic and unreliable, making reliable decision-making a significant challenge. State-of-the-art semantic SLAM approaches do not directly reason about these uncertainties, a gap we aim to address.

Object classification has recently seen many advances with the introduction of deep-learning-based classifiers. Most modern deep-learning-based classifiers provide a vector of class probabilities for a photographed object, given a set of candidate classes. These classifiers are trained on a set of examples for each class and, during deployment, infer the observed objects' class based on the said training set. If the observation does not match images on the training set, the classification result is unreliable and, if not accounted for, may result in erroneous classification. Consequentially, a slight variation in classifier weights or the input may significantly change the output. This variation is referred to as *epistemic* uncertainty or model uncertainty. Several approaches were proposed to identify this uncertainty, such as Monte-Carlo (MC) dropout [2] or Bootstrapping [3]. In our work, we utilize an epistemic-uncertainty-aware classifier and incorporate it within our semantic SLAM framework.

The classifier output generally depends on the relative viewpoint between the camera and the object. This dependency can be modeled [4–7], and then be used to improve classification and localization accuracy within a SLAM setting. However, this type of model was not used in epistemic uncertainty-aware classification. While approaches that consider the accumulated epistemic uncertainty from multiple images, i.e., the posterior epistemic uncertainty, exist (see, e.g., [8]), they decouple the relative pose between object and camera, which is a gap we address. We introduce a viewpoint-dependent classifier uncertainty model that can be utilized for inference and later in planning.

Eventually, semantic SLAM with epistemic-uncertainty-aware classification opens the possibility of performing 'safe' decision-making based on a new type of reward function that considers epistemic uncertainty. Thereafter, this paper presents a novel active semantic SLAM approach that reasons about epistemic uncertainty. To the authors' best knowledge, this is the first work that plans over classifier epistemic uncertainty with uncertain localization.

Specifically, we formulate the active semantic SLAM problem within the belief space planning (BSP) framework, which is an instantiation of a partially observable Markov decision process (POMDP) [9], and consider belief-dependent reward functions. Our approach considers both localization and classifier epistemic uncertainty within BSP by maintaining a joint belief over the robot and object poses and, importantly, over the objects' class posterior probabilities. Having access to such a joint belief within BSP allows us to consider classifier posterior epistemic uncertainty *implicitly* using standard reward functions over the state and information-theoretic rewards. Crucially, it also enables utilizing *novel* reward functions directly over the classifier's posterior epistemic uncertainty. This paper introduces such a reward function and develops methods for its computation (Section 4.4).

Further, an inherent aspect in BSP is belief propagation and reward calculation considering different candidate actions while accounting for possible future observations (see e.g. [10,11]). As an analytical calculation of the corresponding expectation operator is generally unavailable, a common approach is to resort to a sampling-based approximation, which involves generating future observations. In our context, one may consider doing so by generative new images, such as e.g. in [12]. With this alternative, these images would be fed to a classifier to get the corresponding cloud of future semantic measurements (representing the epistemic uncertainty). However, our key observation is that we can use the viewpoint-dependent classifier uncertainty model instead to generate these future semantic measurements directly.

1.1. Related work

Various works presented approaches for sequential classification. Coates and Ng [13] presented an approach that maintains a posterior class probability by multiplying classification scores from an image with the prior class probability. Static State Bayes Filter (SSBF) by Omidshafiei et al. [14] expanded the aforementioned approach for multiple classes. Hierarchical Bayesian Noise Inference by Omidshafiei et al. [14] maintains a posterior class probability vector by utilizing a Dirichlet distributed classifier model. All of these approaches do not consider epistemic uncertainty. Tchuiev and Indelman [8] presented an epistemic-uncertainty-aware sequential method while utilizing MC-dropout by Gal et al. [2] as the mechanism for extracting the epistemic uncertainty for each image. An alternative mechanism might be, e.g., Bootstrapping [3] where multiple classifiers are trained on the same training set, or using auxiliary training techniques and post hoc statistics to detect out-of-distribution input data as proposed by Nitsch et al. [15]. Malinin and Gales [16] proposed prior networks for reasoning about epistemic uncertainty in neural network outputs. These works either did not reason about epistemic uncertainty in classification or did so without considering localization uncertainty. We propose a semantic SLAM approach that performs sequential classification, addresses localization uncertainty, and reasons about posterior epistemic uncertainty in classification.

Some works utilized a viewpoint-dependent classifier model; Velez et al. [17], and Teacy et al. [18] utilized a viewpoint-dependent classifier model in the context of active classification with known poses. Segal and Reid [19] proposed an inference approach for general hybrid beliefs based on message passing. Kopitkov and Indelman [4] presented a Gaussian viewpoint-dependent classifier model and used it for robot localization in a setting where the object class and pose are already known. Feldman and Indelman [20] presented a sequential classification approach with a viewpoint-dependent classifier model where the poses are known a-priori. Tchuiev et al. [5] showed that utilizing a viewpoint-dependent classifier model in a setting of semantic SLAM assists in solving the data association problem. The approach utilized a hybrid belief over poses and classes. This approach was expanded to a multi-robot semantic SLAM setting in [6]. Ok et al. [21] presented an approach for object-based SLAM that used a viewpoint-dependent texture plane measurement model, which is similar in concept to a viewpoint-dependent classifier model. All these approaches utilized a viewpoint-dependent classifier model that did not consider epistemic uncertainty, while on the other hand, we do consider the epistemic uncertainty for inference and planning.

Approaches that incorporate object classification within planning include Atanasov et al. [22] and Patten et al. [23], who presented approaches for active classification using a viewpoint-dependent classifier mode using a sampling-based method. The robot and object poses are known in the former, while in the latter, they are part of the state. Continuous state partially observable Markov decision process (CPOMDP) by Burks et al. [24] is also capable of reasoning about hybrid beliefs. These approaches, however, did not consider the classifier's epistemic uncertainty.

Several planning approaches that do reason about epistemic uncertainty were proposed. Faddoul et al. [25] reasoned about epistemic uncertainty in MDP and POMDP transition matrices, creating a framework for decision-making. Hayashi et al. [26] proposed an approach that actively trains uncertain dynamic models via neural network priors. These works do not consider epistemic uncertainty in the context of classification. Lutjens et al. [27] presented a reinforcement learning approach that reasons about epistemic uncertainty for obstacle avoidance with known object poses. The approach utilized MC dropout and bootstrapping to extract epistemic uncertainty from measurements. On the other hand, we consider a BSP approach with a belief over poses and class probabilities, jointly considering both localization and classifier epistemic uncertainty within a semantic SLAM framework.

To generate measurements, one may consider generating raw images when performing classifier epistemic-uncertainty-aware planning. Ha et al. [12] proposed World Models: a neural network that creates an image given a pose within the environment the network was trained on. Wang et al. [28] proposed an image extrapolation approach using Feature Expansion Network (FEN) and Context Prediction Network (CPN). Mildenhall et al. [29] presented Neural Radiant Fields (NeRF), which rendered images using volume-rendering techniques with a neural network trained on images of the environment with corresponding poses. On the other hand, we present an approach that generates measurements via our proposed viewpoint-dependent classifier uncertainty model.

1.2. Contributions

This paper contributes a unified framework for epistemic uncertainty-aware inference and belief space planning in semantic perception and SLAM. Our framework considers prominent sources of uncertainty — classification aliasing, classifier epistemic uncertainty, and localization uncertainty — within inference and BSP.

Specifically, the main contributions of this paper are as follows.

1. We develop two methods for maintaining a joint distribution over robot and object poses, and over the posterior class probability vector that considers epistemic uncertainty in a Bayesian fashion. The first approach is Multi-Hybrid (MH), where multiple hybrid beliefs over poses and classes are maintained to approximate the joint belief over poses and posterior class probability. The second approach is Joint Lambda Pose (JLP), where the joint belief is maintained directly using a novel JLP factor.
2. We extend both methods to a BSP framework, planning over posterior epistemic uncertainty indirectly or directly via a novel information-theoretic reward over the distribution of posterior class probability.
3. Our inference and BSP methods utilize a novel viewpoint-dependent classifier model that predicts epistemic classifier uncertainty given a candidate class and relative viewpoint. Allowing us to reason about the coupling between poses and classification scores and predict future epistemic classifier uncertainty while avoiding predicting and generating entire images.
4. We study our inference and BSP methods in simulation and using real data from the Active Vision Dataset [30].

1.3. Paper structure

This paper is structured as follows: We cover preliminary material, formulate the addressed problem in Sec. 2, and then provide a brief approach overview in Sec. 2.6. In Sec. 3, we address epistemic-uncertainty-aware inference. MH and JLP are introduced, first for the single object case and afterward for the multiple objects case. In Sec. 4, we expand both approaches to BSP; specifically, in Sec. 4.4 we introduce and develop the calculation of our novel information-theoretic reward over the distribution of posterior class probability. Finally, we validate our approaches first in simulation in Sec. 5.2, then using Active Vision Dataset and BigBIRD in Sec. 5.3.

Table 1
Main notations used in the paper.

Parameters	
x	Robot pose
x^o	Object o's pose
$\mathcal{X}_k$	All robot and object poses up to k
x^{rel}	Relative pose between x and x^o
O_k	Set of all objects observed at time k
x_k^{inv}	Set that contains the last robot pose and all object poses from O_k
c^o	Object o's class
C	Class realization of all objects
z^g	Geometric measurement for a specific object
Z_k^g	All geometric measurements for all objects at time k
z^s	Semantic measurement for a specific object
Z_k^s	All semantic measurements for all objects at time k
n	The amount of all objects in the environment
n_k	Number of objects observed at time k
N_k	Number of objects observed up to time k
$\mathcal{M}_k$	Motion model from x_{k-1} to x_k
a	Robot action
$\mathcal{H}_k$	History of measurements and actions up to time k
$\mathcal{H}_k^g$	History of geometric measurements and action up to time k
$\mathcal{L}^s$	Semantic measurement likelihood
h_c	Expectation of class c's classifier uncertainty model
Σ_c	Covariance of class c's classifier uncertainty model
$\mathcal{L}_k$	Geometric and semantic measurement likelihood at time k
D	Classifier training dataset
$\{\cdot\}$	Set or point cloud
I	Raw image
$l\Box$	Logit transformation of probability vector
γ	Probability vector classifier output
γ^c	Element of γ of class c
Γ_k	Set of all γ observed at time k, one per object
$l\Gamma_k$	Set of all logit transformations for all $\gamma_k \in \Gamma_k$
λ	Posterior class probability vector
λ^c	Element of λ of class c
Λ_k	Posterior probability vector for class realizations
$l\bar{\lambda}_k$	Set of $l\lambda$ of all objects observed up to k
W	Set of all possible classifier weight realizations w
$b[\cdot]$	Belief, probability conditioned on history $\mathbb{P}(\cdot\|I_{1:k}, \mathcal{H}_k^g, D)$.
b_w^c	Continuous belief conditioned on history, c, and w
hb_w	Hybrid belief conditioned on w
$l\mathcal{L}^s$	Logit transformation of semantic measurement likelihood
Subscripts	
w	Classifier weight realization
k	Time step
L	Planning horizon
Superscript	
o	Object o
c	Class hypothesis of an object
C	Class hypothesis of all objects

2. Background and problem formulation

In this section, we introduce notations, provide preliminary material, and formulate the problem addressed in this work. First, we introduce our setting and simultaneous localization and mapping (SLAM) notations. Afterward, we introduce notations specifically for classification in the context of epistemic uncertainty. Finally, we briefly introduce belief space planning (BSP), and present the problem formulation for epistemic uncertainty-aware semantic inference and planning.

For the reader's convenience, the main notations used in this paper are summarized in Table 1.

2.1. Simultaneous localization and mapping (SLAM)

Consider a robot operating in an unknown environment represented by object landmarks. The robot's and objects' pose & objects' classes are all unknown. Let x_k denote the robot pose at time k; let x^o and c denote object pose and class respectively. Denote $\mathcal{X}_x \triangleq \{x^o, x_{0:k}\}$ as all poses of the robot and the (expanded later to multiple) objects up until time k.

The notation $\mathbb{P}(\cdot)$ describes a probability density function (PDF) for continuous random variables, and a probability mass function (PMF) in the case of discrete random variables. For hybrid probabilities of both continuous and discrete random variables, $\mathbb{P}(\cdot)$ is the PDF of the continuous variable conditioned on the discrete variable, multiplied by the PMF of the discrete variable. In other words, consider a discrete random variable a and a discrete random variable b, then $\mathbb{P}(a,b) \triangleq \mathbb{P}(a|b)\mathbb{P}(b)$, where $\mathbb{P}(a|b)$ is a PDF and $\mathbb{P}(b)$ is a PMF.

The robot receives from observed objects both geometric and semantic measurements. Let z_k denote a measurement received at time k from the object. This measurement is split into geometric z_k^g and semantic z_k^s measurements; All those measurements are aggregated to a set $z_k \triangleq \{z_k^g, z_k^s\}$. The robot action at time k is denoted a_k, and finally we denote the measurement history as $\mathcal{H}_k \triangleq \{z_{1:k}, a_{0,k-1}\}$. The measurements at different time steps are assumed independent; in addition, the semantic and geometric measurements are assumed independent as well.

We utilize a known Gaussian motion model with constant parameters, denoted $\mathcal{M}_k$, and defined as:

$$\mathcal{M}_k \triangleq \mathbb{P}(x_k|x_{k-1}, a_{k-1}), \tag{1}$$

and a known geometric model $\mathbb{P}(z_k^g|x^o, x_k)$. In addition, we use an externally trained viewpoint-dependent classifier and uncertainty model $\mathbb{P}(z_{k,n}^s|c_n, x^o, x_k)$ that will be discussed in Section 3.1. Denote the corresponding measurement likelihood term,

$$\mathcal{L}_k \triangleq \mathbb{P}(z_k^g|x^o, x_k) \cdot \mathbb{P}(z_k^s|c, x^o, x_k), \tag{2}$$

where, both geometric and classifier models are considered Gaussian as well. Using $\mathcal{M}_k$ and $\mathcal{L}_k$, the posterior $\mathcal{X}_k$ can be computed recursively via:

$$\mathbb{P}(\mathcal{X}_k|\mathcal{H}_k) \propto \mathbb{P}(\mathcal{X}_{k-1}|\mathcal{H}_{k-1}) \cdot \mathcal{M}_k \cdot \mathcal{L}_k. \tag{3}$$

The likelihood terms and the motion model can be aggregated in a form of a factor-graph [31] for efficient computation of the parameters of $\mathbb{P}(X_k|\mathcal{H}_k)$.

2.2. Distribution over class probability vector

During inference, the robot receives a raw image in which observed objects are segmented. In standard (deep-learning) approaches a classifier is learned beforehand and used to classify the objects within each segment (e.g. bounding box) by producing an output of a class probability vector. Given fixed classifier weights w, we denote a probability vector from a classifier at time k as

$$\gamma_k \triangleq \mathbb{P}(c|I_k, w), \tag{4}$$

where I_k is the raw image of the object. Denote $\gamma_{k,w}$ as the probability vector given a specific w. In practice, the image fed into the classifier is a cropped image of an object via a bounding box. Note that $\gamma_k \triangleq [\gamma_k^1, ..., \gamma_k^m] \in \mathbb{R}^m$ is a probability vector, thus it must satisfy the following conditions:

- All its elements must sum to 1, i.e. $\sum_{i=1}^m \gamma_k^i = 1$.
- Each element is bounded between 0 and 1, i.e. $0 \leq \gamma_k^i \leq 1, \ \forall i = 1, ..., m$.

In contrast to this standard approach, we reason about classifier epistemic uncertainty. Denote D as the classifier's training set. Training a neural network is inherently a stochastic process, therefore w are random variables (as described in e.g. [2]) conditioned on D such that $w \sim \mathbb{P}(w|D)$, thus making γ_k a random variable. Unfortunately $\mathbb{P}(w|D)$ is not given explicitly, thus we create a set W of sampled w realizations to produce a point cloud of γ_k vectors per object and time step, such that the distribution over γ_k is described with the delta Dirac function $\delta(\cdot)$:

$$\gamma_k \sim \mathbb{P}(\gamma_k|I_k, D) = \int_w \delta(\gamma_k = \mathbb{P}(c|I_k, w))\mathbb{P}(w|D)dw, \tag{5}$$

which is approximated via sampling as:

$$\mathbb{P}(\gamma_k|I_k, D) \approx \frac{1}{|W|}\sum_w \delta(\gamma_k = \mathbb{P}(c|I_k, w)). \tag{6}$$

Thus, for each time step, we get a point cloud $\{\gamma_k\}$ per object where its "spread" describes the epistemic model uncertainty of the classifier. See Fig. 1, where an object is observed from multiple viewpoints, and the classifier outputs a cloud of γ's for each viewpoint. For example, the cloud $\{\gamma_k\}$ obtained by observing the object from the bottom right corner is widely spread, representing high epistemic uncertainty. Contrast with the upper-right viewpoint where the spread is tight, representing low epistemic uncertainty. In this paper, the semantic measurements are those point clouds within the $m-1$ simplex, such that $z_k^s = \{\gamma_k\}$. The set of sampled w can be created by, for example, MC-dropout [2], or Bootstrapping [3].

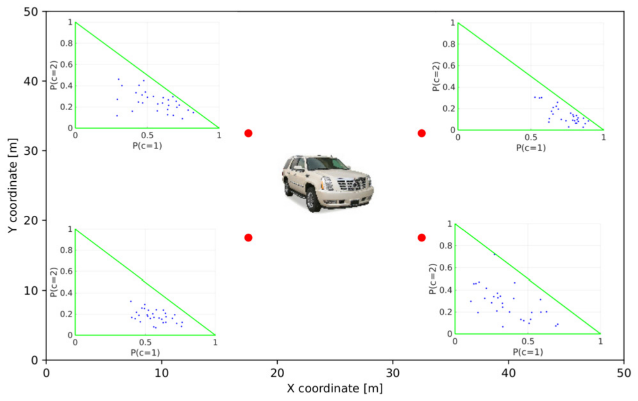


Fig. 1. Illustration of viewpoint dependency for classification scores and epistemic uncertainty. The figure presents simplex graphs for different viewpoints, where $m = 3$. The individual class probability scores are shown as blue points in the simplex, whose borders are light green. The large red points represent possible viewpoints observing the SUV in the middle. (For interpretation of the colors in the figures, the reader is referred to the web version of this article.)

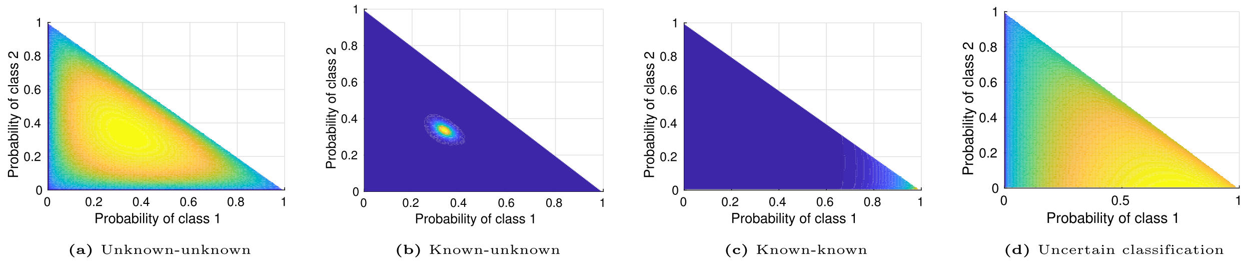


Fig. 2. The 4 archetypes of $b[\lambda_k]$, shown in a 3 dimensional Dirichlet simplex example, where blue to yellow correspond to low to high probability respectively. **(a)** is an out-of-distribution setting, where the classifier does not identify the object and the epistemic uncertainty is high. **(b)** is high data uncertainty setting, which is close to D, but the class is identifiable in the training set itself. In **(c)** the classifier recognizes the object of a certain class with certainty, a scenario we aim for. In **(d)** the classifier gives preference to one of the classes, but with a high degree of uncertainty.

2.3. Distribution over posterior class probability vector

Eventually, the posterior over a *sequence* of γ vectors can be inferred. This posterior considers both the epistemic uncertainty from multiple observations of an object, and localization uncertainty induced by coupling between relative poses and class probabilities. The posterior is defined as follows:

$$\lambda_k \triangleq \mathbb{P}(c|\gamma_{1:k}, z^g_{1:k}), \tag{7}$$

where λ_k is deterministically determined by both a sequence $\gamma_{1:k}$ and the geometric measurement history. Similar to γ_k, $\lambda_k \triangleq [\lambda^1_k, ..., \lambda^m_k] \in \mathbb{R}^m$ is a probability vector, with each element corresponding to its class c_i $\forall i = [1, m]$:

$$\lambda_k \triangleq \left[\mathbb{P}(c = 1|\gamma_{1:k}, z^g_{1:k}), \cdots, \mathbb{P}(c = m|\gamma_{1:k}, z^g_{1:k})\right]^T. \tag{8}$$

For a specific $\gamma_{1:k,w}$ sequence which is created by a specific w, the notation $\lambda_{k,w}$ is used. Because $\gamma_{1:k}$ is a random variable (as w is a random variable), so is λ_k. The key concept is that λ_k, which is a vector of class probabilities, is a random variable by itself. In other words, the belief over λ_k is a PDF over a PMF. As such, define the belief over λ_k as:

$$b[\lambda_k] \triangleq \mathbb{P}(\lambda_k|I_{1:k}, \mathcal{H}^g_k, D). \tag{9}$$

The belief $b[\lambda_k]$ encompasses the posterior classification probability vector via $\mathbb{E}(\lambda_k)$. Both $\mathbb{E}(\lambda_k)$ and the covariance $\Sigma(\lambda_k)$ are dependent on epistemic and localization uncertainty. The history $\mathcal{H}^g_k \triangleq \{a_{0:k-1}, Z^g_{1:k}\}$ includes all geometric measurements and actions up until time k. The belief $b[\lambda_k]$ representation is more expressive than a single λ_k, and reflects four possible archetypes, as seen in Fig. 2 ([16]). Fig. 2a presents an out-of-distribution case where the inputs to the classifier are totally alien, therefore the output is completely unpredictable. In Fig. 2c the classifier can safely identify the object with a high degree of certainty, i.e. the input is close to the training set. Intuitively, this is the case we aim for, and generally has the highest reward. In Fig. 2b the classifier certainly cannot disambiguate between different classes because of high data uncertainty, i.e. the classifier “knows” that it does not know. This can result from ambiguity in the training set between different classes when objects from different classes look identical from certain viewpoints. Finally, in Fig. 2d the classifier can vaguely infer the object class, but it is still far from the training set (e.g. a car of an unusual shape that there are no similar images in the training set), therefore with a large degree of uncertainty.

As will be discussed, the belief $b[\lambda_k]$ can be used within belief space planning (BSP), e.g., going to relative poses where the epistemic uncertainty is the smallest to safely classify objects, or vice-versa going to relative poses with high epistemic uncertainty to learn a model online potentially. Using $b[\lambda_k]$ within BSP is a key contribution of this paper.

More generally, λ_k is coupled with object and camera poses $\mathcal{X}_k$. A joint belief over λ_k and $\mathcal{X}_k$ can be maintained, denoted as:

$$b[\lambda_k, \mathcal{X}_k] \triangleq \mathbb{P}(\lambda_k, \mathcal{X}_k | I_{1:k}, \mathcal{H}_k^g, D). \tag{10}$$

$\mathbb{E}(\lambda_k)$ may be required to, for example, compute a reward function that depends on $\mathbb{E}(\lambda_k)$. Consider that $\mathbb{P}(c = i|\lambda_k, \mathcal{X}_k) = \lambda_k^i$, then for every object class $c = i$:

$$\mathbb{P}(c = i|I_{1:k}, \mathcal{H}_k^g, D) = \int\limits_{\lambda_k, \mathcal{X}_k} \lambda_k^i \cdot b[\lambda_k, \mathcal{X}_k] d\lambda_k d\mathcal{X}_k = \mathbb{E}(\lambda_k^i). \tag{11}$$

In [8] we presented an approach to maintain $b[\lambda_k]$ in a setting with a single object, without considering the coupling between $\mathcal{X}_k$ and λ_k, and the approach was limited to inference. On the other hand, here we account for the coupling between λ_k and $\mathcal{X}_k$, present an active approach, and expand to a multi-object setting. First, we consider the formulation for a single object. In Sec. 3 and Sec. 4 we extend the formulation to the multiple object case, with each method having its specific notations.

2.4. Belief space planning (BSP)

Given a general current belief b_k, one can reason about the best future action from a set of actions to maximize (or minimize) an object function. With b_k and a set of future actions $a_{k:k+L}$, the objective function is commonly defined as the expected cumulative reward,

$$J(b_k, a_{k:k+L}) = \mathbb{E}_{\mathcal{Z}_{k+1:k+L}}(\sum_{i=0}^{L} r(b_{k+i}(\mathcal{Z}_{k+1:k+i})), a_{k+i}), \tag{12}$$

where $r(\cdot)$ is a belief-dependent reward function, and L is the planning horizon. This formulation can be extended to policies as well.

The above equation can also be written in a recursive form as in,

$$J(b_k, a_{k:k+L}) = \int\limits_{\mathcal{Z}_{k+1}} \mathbb{P}(\mathcal{Z}_{k+1}|\mathcal{H}_k, a_k) \cdot \cdot J(b_{k+1}, a_{k+1:k+L}) d\mathcal{Z}_{k+1} + r(b_k, a_k), \tag{13}$$

where $b_{k+1} = b_{k+1}(\mathcal{Z}_{k+1})$. The term $\mathbb{P}(\mathcal{Z}_{k+1}|\mathcal{H}_k, a_k)$ is the measurement likelihood of future measurement history thus far and a_k, and is essential for BSP. In practice, the integral in Eq. (13) is intractable, thus approximated via sampling:

$$J(b_k, a_{k:k+L}) \approx \frac{1}{N_z} \sum_{\mathcal{Z}_{k+1}} J(b_{k+1}(\mathcal{Z}_{k+1}), a_{k+1:k+L}), \tag{14}$$

where N_z is the number of Z_{k+1} samples, and $Z_{k+1} \sim \mathbb{P}(Z_{k+1}|\mathcal{H}_k, a_k)$.

The optimal action sequence $a^*_{k:k+L}$ maximizes the objective function:

$$a^*_{k:k+L} = \underset{a_{k:k+L}}{\arg\max} \left(J(b_k, a_{k:k+L}) \right). \tag{15}$$

To evaluate $a^*_{k:k+L}$, one must consider all possible sequences (possibly via search algorithms) and select the one that produces the highest objective function. Specifically, we consider $b_k = b[\lambda_k, \mathcal{X}_k]$ for BSP, and discuss planning using various reward functions while focusing on classifier epistemic uncertainty reward function, namely the entropy of $b[\lambda_{k+i}]$ for a future time $k+i$. Yet, first, we must address the corresponding inference problem.

2.5. Problem formulation

Given geometric measurement history $\mathcal{H}_k^g$, an image sequence $I_{1:k}$, actions $a_{0:k-1}$, an epistemic-uncertainty-aware classifier trained on training dataset D with a set W of weight realizations $w \in W$, the problems of inference and planning are defined as follows:

1. *Inference:* Infer the posterior joint belief $b[\lambda_k, \mathcal{X}_k]$, as defined in Eq. (10).
2. *Planning:* Given $b[\lambda_k, \mathcal{X}_k]$, find the future action sequence $a^*_{k:k+L}$ that maximizes $J(b[\lambda_k, \mathcal{X}_k], a_{k:k+L})$ with the reward function $r(b[\lambda_k, \mathcal{X}_k])$.

We address the inference and planning problems in Sections 3 and 4, respectively.

2.6. Approach overview

Two approaches are presented for solving each of the problems presented in Sec. 2.5. The first approach is Multi-Hybrid (MH), a particle-based approach where multiple hybrid beliefs are maintained simultaneously. The second approach is Joint Lambda Pose (JLP), where a single continuous belief is maintained, and the posterior class probabilities are states within this belief.

The approach sections are divided into inference and planning; starting with inference, we introduce the viewpoint-dependent classifier uncertainty model, which predicts the distribution of the classifier output and is used by both methods for inference and planning. In particular, the classifier uncertainty model generates predicted measurements during planning. Then, we introduce MH and JLP for inference.

Section 4 addresses the planning problem by delving into the specifics of both MH and JLP of generating measurements for multiple objects. Afterward, we discuss reward functions and specifically expand upon information-theoretic reward for $b[\lambda]$.

3. Epistemic uncertainty aware inference

3.1. Viewpoint dependent classifier uncertainty model

We use a classifier uncertainty model that accounts for the coupling between localization and classification, and epistemic model uncertainty. As an example, Fig. 1 illustrates that $\{\gamma_k\}$ measurements vary across different viewpoints, with some containing high epistemic uncertainty and some low. The model learns to predict these measurements, and subsequently the viewpoints containing high epistemic uncertainty. In contrast, previous works that used a viewpoint-dependent classifier model (e.g. [5,6,18,22]) did not consider epistemic uncertainty while learning the model.

The conditions for γ_k being a probability vector must be considered when one must sample from the classifier model, thus unlike previous works [5,6], we cannot use a Gaussian distributed classifier model. One possible solution is to consider the classifier model as Dirichlet distributed (see [8]), but that model cannot be incorporated into a Gaussian optimization framework (e.g. iSAM2 [32]) with unknown poses which are coupled with classification results. Instead, we consider the following solution: we use a logit transformation for γ to a vector $l\gamma \in \mathbb{R}^{m-1}$ space, such that the support of each element $(-\infty, \infty)$:

$$l\gamma \triangleq \left[\log\left(\frac{\gamma^1}{\gamma^m}\right), \log\left(\frac{\gamma^2}{\gamma^m}\right), \ldots, \log\left(\frac{\gamma^{m-1}}{\gamma^m}\right)\right]^T. \tag{16}$$

Then, $l\gamma$ can be assumed Gaussian such that:

$$\mathbb{P}(l\gamma_k|c, x^o, x_k) = \mathcal{N}(h_c(x^o, x_k), \Sigma_c(x^o, x_k)), \tag{17}$$

and as a consequence γ_k is distributed Logistical Gaussian with parameters $\{h_c, \Sigma_c\}$. The probability density function (PDF) of γ_k is as follows:

$$\mathbb{P}(\gamma_k|c, x^o, x_k) = \frac{1}{\sqrt{|2\pi\,\Sigma_c|}} \cdot \frac{1}{\prod_{i=1}^m \gamma_k^i} \cdot e^{\left(-\frac{1}{2}||l\gamma_k - h_c||^2_{\Sigma_c}\right)}. \tag{18}$$

In practice, a classifier provides a cloud $\{\gamma_k\}$, and each $\gamma_k \in \{\gamma_k\}$ is transformed to $l\gamma_k$. There are m such models, one for each class. The training set consists of tuples of relative pose and $l\gamma$ point clouds such that $D_{cm} \triangleq \{x^{rel}, \{l\gamma\}\}$ for each class, where $x^{rel} \triangleq x^o \ominus x$ is the relative pose between object and robot. With the notation of x^{rel}, $h_c(x^o, x_k)$ becomes $h_c(x^{rel})$, and similarly Σ_c in Eq. (17). The expectation (classification scores) and covariance (epistemic uncertainty) is extracted from $\{l\gamma\}$ and fitted as known points either in the model using e.g. Gaussian Processes or deep-learning-based approaches. Real-life applications may require creating D_{cm} from multiple instances of the same objects, e.g. for class "car" multiple types of cars may be used. Fig. 3 illustrates the training data shown in black dots versus the trained model shown in blue. The model attempts to "predict" the epistemic uncertainty based on a given training set.

3.2. Multi-Hybrid inference

We present the Multi-Hybrid (MH) inference approach to maintaining the belief from $b[\lambda_k, \mathcal{X}_k]$ (10). With this method, we maintain $b[\lambda_k, \mathcal{X}_k]$ indirectly via a set of hybrid beliefs, each for a realization of classifier weights. The posterior class

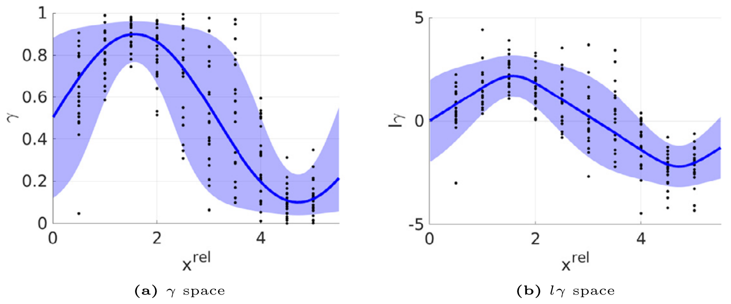


Fig. 3. A simplified illustration of the classifier uncertainty model is used in the paper. **(a)** and **(b)** represent γ and $l\gamma$ space respectively. The black dots represent the corresponding $\gamma(x^{rel}, w) \in \{\gamma\}(x^{rel})$ and $l\gamma(x^{rel}, w) \in \{l\gamma\}(x^{rel})$. The expectation and covariance are learned in **(b)** and interpolated for new queries of x^{rel}, potentially returning to **(a)** via the inverse logit transformation. The expectation is represented in dark blue, while the one-sigma covariance is represented in light blue.

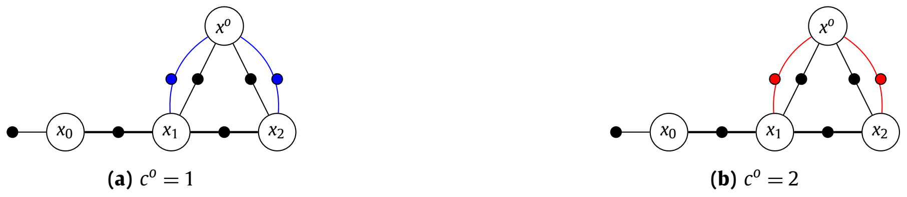


Fig. 4. Factor graphs for a toy scenario where the camera observes an object for a specific w, there are $|W|$ such factor graph pairs. The object has two candidate classes. Each dot and line represents a separate factor. The black factors between the camera and object represent the geometric model, while the colored factors represent the classifier models, $c = 1$ and $c = 2$ by blue and red respectively.

probabilities from each hybrid belief together represent the posterior classifier epistemic uncertainty. This section presents the important details of MH, with Appendix A.1 and Appendix A.2 containing the full derivations.

3.2.1. Single object

The belief $b[\lambda_k, \mathcal{X}_k]$, as defined by Eq. (10), is conditioned both on $\mathcal{H}_k^g$ and $I_{1:k}$, and D. As discussed in Sec. 2.2, an epistemic uncertainty aware classifier provides a cloud $\{l\gamma_{1:k}\}$ as semantic measurements. In MH, we maintain $b[\lambda_k, \mathcal{X}_k]$ by splitting it into components by marginalizing over c and $w \in W$, where W is a predetermined discrete set of w that are used throughout the entire scenario. Each λ_k component of weight w is denoted $\lambda_{k,w}$. Thereafter, $b[\lambda_k, \mathcal{X}_k]$ is maintained via:

$$b[\lambda_k, \mathcal{X}_k] \approx \frac{1}{|W|} \sum_c \sum_w hb_w[\mathcal{X}_k, c] \cdot \delta(\lambda_k - \lambda_{k,w}), \tag{19}$$

where δ denotes the delta Dirac function, and $hb_w[\mathcal{X}_k, c]$ is denoted as the hybrid belief of robot and object poses $\mathcal{X}_k$ for a specific weight w:

$$hb_w[\mathcal{X}_k, c] \triangleq \mathbb{P}(\mathcal{X}_k | w, c, I_{1:k}, \mathcal{H}_k^g, D) \cdot \mathbb{P}(c | w, I_{1:k}, \mathcal{H}_k^g, D). \tag{20}$$

The above hybrid belief is maintained as in [5], and $|W|$ hybrid beliefs must be maintained. The full derivation of Eq. (19), as well as the marginals over λ_k and $\mathcal{X}_k$, can be found in Appendix A.1. See Fig. 4 for a simple example of a graph representation for a single object with two candidate classes.

3.2.2. Multiple objects

Let O_k denote the set of objects the robot sees up to time k. We denote variables corresponding to object o with a superscript $\square^o$. Each object is segmented from the image with its classifier outputs $\{\gamma_{k,w}^o\}_{w \in W}$. The set of all those clouds for all objects in O_k is denoted as $\{\Gamma_k\}$ with Γ defined as a realization of γ measurements, one per each observation. For a specific $w \in W$, we define the realization of γ measurements as $\Gamma_{k,w} \triangleq \{\gamma_{k,w}^o\}_{o \in O_k}$, thus $\{\Gamma_k\} \triangleq \{\Gamma_{k,w}\}_{w \in W}$. We define $l\Gamma_k$ as the logit transformation of all $\gamma_k \in \Gamma_k$ as in Eq. (16). The set of all geometric measurements at time k is denoted Z_k^g, the history $\mathcal{H}_k^g \triangleq \{a_{0:k-1}, Z_{1:k}^g\}$ includes all geometric measurements and actions up until time k, and subsequently $\mathcal{H}_k \triangleq \{I_{1:k}, \mathcal{H}_k^g\}$ includes all measurement and action history up to time k.

We define the joint posterior class probability vector as:

$$\Lambda_k \triangleq \mathbb{P}(C | l\Gamma_{1:k}, \mathcal{H}_k^g). \tag{21}$$

Thus, similarly to Sec. 3.2.1, we can write $b[\Lambda_k, \mathcal{X}_k]$ as:

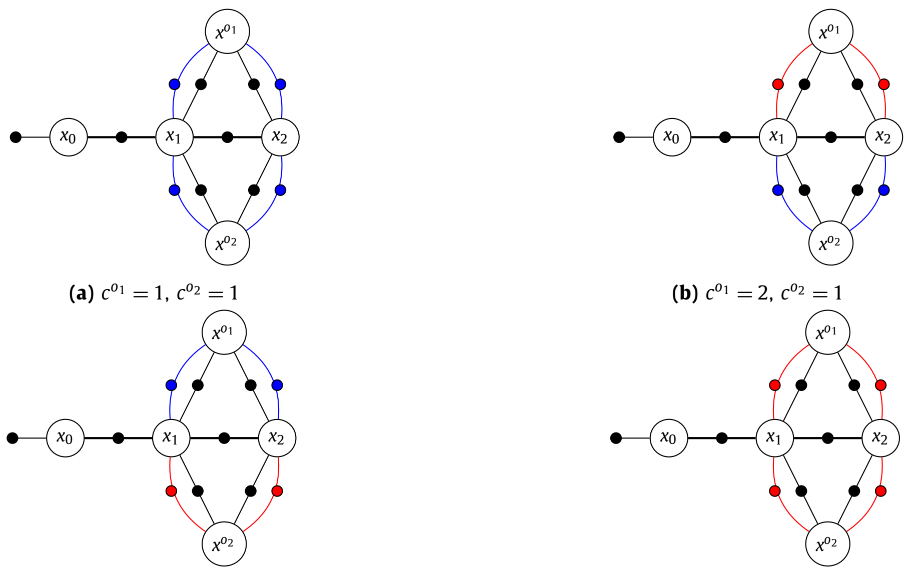

x^{o_1}
x_0
x_1
x_2
x^{o_2}
(a) $c^{o_1}=1$, $c^{o_2}=1$
(b) $c^{o_1}=2$, $c^{o_2}=1$
(c) $c^{o_1}=1$, $c^{o_2}=2$
(d) $c^{o_1}=2$, $c^{o_2}=2$

Fig. 5. Factor graphs used by MH with the same scenario as in Fig. 6b for a single w. Each dot and line represents a separate factor. The black factors between the camera and object represent the geometric model, while the colored factors represent the classifier models, $c=1$ and $c=2$ by blue and red respectively.

$$b[\Lambda_k,\mathcal{X}_k]\approx\frac{1}{|W|}\sum_w\sum_C hb_w[\mathcal{X}_k,C]\cdot\delta(\Lambda_k-\Lambda_{k,w}),\tag{22}$$

where $hb_w[\mathcal{X}_k,C]$ is now defined as:

$$hb_w[\mathcal{X}_k,C]\triangleq\mathbb{P}(\mathcal{X}_k|w,C,\mathcal{H}_k,D)\cdot\mathbb{P}(C|w,\mathcal{H}_k,D),\tag{23}$$

with the full derivation of Eq. (22) with the marginals for $\mathcal{X}_k$ and Λ_k presented in Appendix A.2.

In general, all $\mathcal{X}_k$ and C are coupled, and subsequently so do $\mathcal{X}_k$ and Λ_k. There are two possible sources of coupling: class priors that depend on other objects' classes (e.g. a computer mouse may be expected to appear next to a monitor), and the coupling between poses and classes induced by the viewpoint-dependent classifier uncertainty model (17).

While accurate, due to the combinatorial nature of C which considers all possible class realizations, the need to simultaneously maintain $|W|$ hybrid beliefs, the worst-case complexity of MH scales poorly with the number of objects and candidate classes. Fig. 5 illustrates the above for a simple example with two classes and two objects. In practice, pruning class realizations with low probability can reduce the computational complexity to manageable levels. Incremental inference approaches for hybrid beliefs, in line with [33], could further reduce computational complexity. These, however, are outside the scope of this paper. As an alternative, in the next section, we propose the JLP algorithm, which is by far computationally more efficient than MH.

3.3. Joint Lambda Pose inference

We present Joint Lambda Pose (JLP), an alternative, computationally lighter approach. Its accuracy depends on the conditions discussed below. Similarly to MH, we first consider the single object case and then extend JLP to the multiple object case. To the best of our knowledge, no approaches combine Gaussian distributed variables with random variables within a simplex besides sampling-based methods. Thus, $b[\lambda_k,\mathcal{X}_k]$ cannot be maintained as a single continuous belief, e.g. MH requires maintaining multiple hybrid beliefs, as discussed in Sec. 3.2. Instead, define $l\lambda_k$ as the logit transformation of λ_k, and the corresponding belief is maintained (considering a single object, for now):

$$b[l\lambda_k,\mathcal{X}_k]\triangleq\mathbb{P}(l\lambda_k,\mathcal{X}_k|I_{1:k},\mathcal{H}_k^g,D).\tag{24}$$

For a general λ_k, each λ_k^c can be updated using Bayes rule:

$$\lambda_k^c=\eta\cdot\lambda_{k-1}^c\cdot\mathbb{P}(l\gamma_k|c,x_k^{rel}).\tag{25}$$

When λ_k is cast into logit space, the above equation transforms into the following sum, written in a vector form:

$$l\lambda_k=l\lambda_{k-1}+l\mathcal{L}_k^s,\tag{26}$$

where for each element in λ_k, the normalizer η gets canceled as it is identical for all elements of λ_k, and $l\mathcal{L}^s_k$ is defined as:

$$l\mathcal{L}^s_k \triangleq \left[\log\left(\frac{\mathbb{P}(l\gamma_k|c=1, x^{rel}_k)}{\mathbb{P}(l\gamma_k|c=m, x^{rel}_k)}\right), ..., \log\left(\frac{\mathbb{P}(l\gamma_k|c=m-1, x^{rel}_k)}{\mathbb{P}(l\gamma_k|c=m, x^{rel}_k)}\right)\right]^T. \tag{27}$$

To recursively update a Gaussian $l\lambda_k$ in closed form from a Gaussian $l\lambda_{k-1}$, $l\mathcal{L}^s_k$ needs to be Gaussian as well.

The condition under which $l\mathcal{L}^s_k$ is accurately Gaussian distributed is described in the following Lemma:

Lemma 1. *Given m Gaussian distributed viewpoint-dependent classifier uncertainty models $\mathbb{P}(l\gamma_k|c, x^{rel}_k)$ as in Eq. (17), if the model covariance $\Sigma_{c=i}(x^{rel}_k) \equiv \Sigma_{c=j}(x^{rel}_k)$ $\forall i, j \in [1, m]$, then $l\mathcal{L}^s_k$ is Gaussian distributed.*

The proof is given in Appendix A.4. With the above Lemma, the following section introduces the JLP factor, which constructs $b[l\lambda_k, \mathcal{X}_k]$.

3.3.1. Joint Lambda Pose (JLP) factor

Assume the conditions in Lemma 1 are satisfied. Considering Eq. (A.24) for all $i \in [1, m-1]$, we can describe $l\mathcal{L}^s_k$ as follows:

$$l\mathcal{L}^s_k = \Phi l\gamma_k - \frac{1}{2}\phi, \tag{28}$$

where the matrix $\Phi \in \mathbb{R}^{(m-1)\times(m-1)}$ and the vector $\phi \in \mathbb{R}^{m-1}$ depend on the individual classifier models (17) and x^{rel}_k. Using Eq. (A.24), the matrix Φ is defined as

$$\Phi \triangleq \begin{bmatrix} h^T_{c=1}\Sigma^{-1}_{c=1} - h^T_{c=m}\Sigma^{-1}_{c=m} \\ \vdots \\ h^T_{c=m-1}\Sigma^{-1}_{c=m-1} - h^T_{c=m}\Sigma^{-1}_{c=m} \end{bmatrix}, \tag{29}$$

and ϕ is defined as

$$\phi \triangleq \begin{bmatrix} h^T_{c=1}\Sigma^{-1}_{c=1}h_{c=1} - h^T_{c=m}\Sigma^{-1}_{c=m}h_{c=m} \\ \vdots \\ h^T_{c=m-1}\Sigma^{-1}_{c=m-1}h_{c=m-1} - h^T_{c=m}\Sigma^{-1}_{c=m}h_{c=m} \end{bmatrix}, \tag{30}$$

where $h_{c=i}$ and $\Sigma_{c=i}$ are the expectation and covariance matrix of the model for class i, $\forall i = [1:m]$. If the conditions of Lemma 1 are satisfied, we can substitute $l\mathcal{L}^s_k$ in Eq. (26) with the expression in Eq. (28):

$$l\lambda_k = l\lambda_{k-1} + \Phi l\gamma_k - \frac{1}{2}\phi. \tag{31}$$

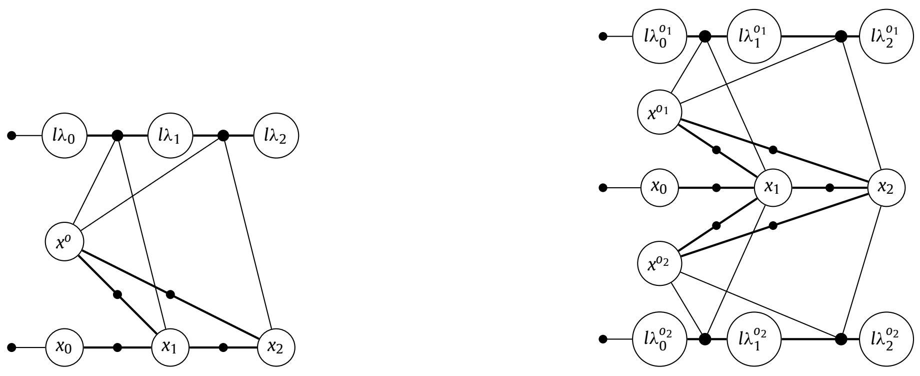
As $l\gamma_k$ is assumed Gaussian, its distribution is defined by expectation $\mathbb{E}(l\gamma_k)$ and covariance $\Sigma(l\gamma_k)$. Assuming a non-singular matrix Φ, we define the JLP factor as:

$$\mathbb{P}(l\lambda_k|l\lambda_{k-1}, I_k, D, x^{rel}_k) \triangleq \mathcal{N}\left(l\lambda_{k-1} + \Phi\mathbb{E}(l\gamma_k) - \frac{1}{2}\phi, \Phi\Sigma(l\gamma_k)\Phi^T\right). \tag{32}$$

As mentioned before, we utilize a classifier that outputs a set $\{\gamma_k\}$ instead of a single γ_k. Each $\gamma_k \in \{\gamma_k\}$ is then transformed via the logit transformation (16) to $l\lambda_k$, thus the entire set $\{\gamma_k\}$ is transformed to $\{l\gamma_k\}$. From there $\mathbb{E}(l\gamma_k)$ and $\Sigma(l\gamma_k)$ are inferred, and the JLP factor can be written as $\mathbb{P}(l\lambda_k|l\lambda_{k-1}, \{l\gamma_k\}, x^{rel}_k)$. As in Sec. 3.2.1, $\{l\gamma_k\}$ represents the classifier's epistemic uncertainty.

The factor (32) is a four-variable factor of $l\lambda_k$, $l\lambda_{k-1}$, x^o, and x, with the latter two used to compute x^{rel} via $x^{rel} \triangleq x^o \ominus x$. The factor can be inserted into a graph structure that can be optimized using standard SLAM methods, where $l\lambda_k$ for different k are separate variable nodes. This factor enables us to maintain $b[l\lambda_k, \mathcal{X}_k]$ using a single continuous belief, as discussed in the next section, and in turn, be faster computationally than MH.

The term $\Phi\Sigma(l\gamma_k)\Phi^T$ is positive definite when Φ is not singular, but in practice, we cannot guarantee this condition. If there is some x^{rel}_k for classes $c = i$ and $c = j$ where $h_{c=i}(x^{rel}) = h_{c=j}(x^{rel})$ and $\Sigma_{c=i}(x^{rel}) = \Sigma_{c=j}(x^{rel})$, then at that point Φ is singular. At that certain x^{rel}_k, the two classes cannot be discerned given the classifier models. While such a situation is possible, the corresponding $l\lambda$ elements become unconstrained, which poses an implementation problem for SLAM optimization algorithms such as iSAM2, which we use in this paper. To keep $\Phi\Sigma(l\gamma_k)\Phi^T$ non-singular, we add to it an identity matrix multiplied by a small positive constant $\epsilon \cdot I^{(m-1)\times(m-1)}$.



(a) A single object. **(b)** Two objects.

Fig. 6. Example of a factor graph for a scenario until $k = 2$, where at each time step an object **(a)** or objects **(b)** are observed. There are priors for x_0, and $l\lambda_0$ for every object. Between the camera poses are motion factors, connecting camera and object poses are geometric measurement factors, and between $l\lambda$'s at different time steps the 4 variable factors are connecting.

3.3.2. *Recursive update formulation*

A smoothing formulation may be considered where the joint belief $b[l\lambda_{1:k}, \mathcal{X}_k]$ is maintained. Using Bayes and chain rules, $b[l\lambda_{1:k}, \mathcal{X}_k]$ can then be updated as:

$$b[l\lambda_{1:k}, \mathcal{X}_k] = \eta \cdot \mathbb{P}(l\lambda_k | l\lambda_{k-1}, I_k, D, x_k^{rel}) \cdot \mathcal{M}_k \cdot \mathbb{P}(z_k^g | x_k^{rel}) \cdot b[l\lambda_{1:k-1}, \mathcal{X}_{k-1}], \tag{33}$$

where η is a normalization constant. In practice, previous $l\lambda_{1:k-1}$ are typically not required for classification inference and planning, so we can consider the belief $b[\lambda_k, \mathcal{X}_k]$, without maintaining a large number of states per object. To update this belief recursively, it is expressed as a function of the prior $b[l\lambda_{k-1}, \mathcal{X}_{k-1}]$. To do so, we marginalize over $l\lambda_{k-1}$ and use the Bayes rule:

$$b[l\lambda_k, \mathcal{X}_k] \propto \int_{l\lambda_{k-1}} \mathbb{P}(l\lambda_k | l\lambda_{k-1}, I_k, D, x_k^{rel}) \cdot \mathcal{M}_k \cdot \mathbb{P}(z_k^g | x_k^{rel}) \cdot b[l\lambda_{k-1}, \mathcal{X}_{k-1}] dl\lambda_{k-1}. \tag{34}$$

Fig. 6a presents a simple example to illustrate the factor graph structure using JLP. In this figure, we present a scenario with two time steps in which the robot observes a single object.

The extension to multiple objects within the JLP framework is straight-forward. Each object o has its own set of $l\lambda_k^o$ nodes, as seen in the example in Fig. 6b. The set of all $l\lambda_k^o$ for objects observed thus far is denoted as $l\bar{\lambda}_k \triangleq \{l\lambda_k^o\}_{o\in O_{1:k}}$. In contrast with Λ_k, which is a single probability vector over class realization with m^{N_k} categories, $l\bar{\lambda}_k$ is a set of vectors with $m-1$ elements each, being the logit transformation of a probability vector of an object, to a total of $(m-1)\cdot N_k$ elements for $l\bar{\lambda}_k$. As such, the joint belief is updated similarly to the single object case:

$$b[l\bar{\lambda}_k, \mathcal{X}_k] = \eta \int_{l\bar{\lambda}_{k-1}} \prod_{o\in O_k} \mathbb{P}(l\lambda_k^o | l\lambda_{k-1}^o, \{l\gamma_k^o\}, x_k^{rel}) \cdot \mathcal{M}_k \mathbb{P}(Z_k^g | \mathcal{X}_k) \cdot b[l\bar{\lambda}_{k-1}, \mathcal{X}_{k-1}] dl\bar{\lambda}_{k-1}, \tag{35}$$

with η being a normalization constant that does not participate in inference.

At worst, MH's time computational complexity scales exponentially with the number of objects observed, whereas JLP's scales polynomially. Therefore, JLP is significantly more computationally efficient. Appendix A.5 further discusses the computational complexity of JLP compared to MH, and in particular, the difference between maintaining $b[\Lambda_k, \mathcal{X}_k]$ in MH and $b[l\bar{\lambda}_k, \mathcal{X}_k]$.

4. Epistemic uncertainty aware semantic belief space planning

This section presents a framework for epistemic uncertainty-aware semantic BSP (EUS-BSP). Our framework incorporates reasoning about future posterior epistemic uncertainty within BSP. Moreover, future semantic and geometric measurements are generated using the coupling between λ and $\mathcal{X}$. Importantly, maintaining the corresponding future posterior belief $b[\lambda, \mathcal{X}]$ within BSP allows us to utilize a variety of reward functions, and in particular, information-theoretic rewards over epistemic uncertainty. As such, EUS-BSP provides critical capabilities for reliable autonomous semantic perception in uncertain environments. Each inference approach developed in Section 3 has its BSP counterpart. As discussed in detail below, they are not compatible with each other, i.e. MH, planning must be used with inference, and the same for JLP.

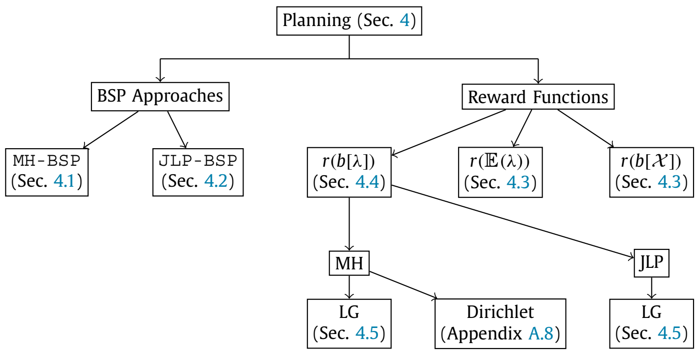


Fig. 7. A diagram of aspects considered in Sec. 4. *Dir* stands for Dirichlet distribution.

This section is structured as follows; first, we discuss MH-BSP in Sec. 4.1 and JLP-BSP in Sec. 4.2. Afterward, we address possible reward functions, first mentioning rewards in the form of $r(b[\mathcal{X}])$ and $r(\mathbb{E}(\lambda))$ in Sec. 4.3, both indirectly involving reasoning about epistemic uncertainty. Furthermore, we discuss an epistemic uncertainty information-theoretic reward $r(b[\lambda])$ in Sec. 4.4, specifically the negative of differential entropy $-H(\lambda)$. The computation of $-H(\lambda)$ is discussed for LG in Sec. 4.5, which is relevant for both MH and JLP. In Appendix A.8 we discuss the alternative of Dirichlet distributed λ for MH, and in Appendix A.9 we discuss the difference between LG and Dirichlet. Fig. 7 presents a diagram of all aspects considered in this section.

4.1. *Multi-Hybrid planning (MH-BSP)*

In this section and thereafter we assume the set of possible candidate actions $a_{k,k+L}$ is discrete and known. Our approach may use any action generation method, e.g. Probabilistic Road-Map (RPM) [34] or fixed motion primitives. Considering multiple objects, future Z^g and $\{l\Gamma\}$ must be generated, s.t. the objective function is as follows:

$$J(b[\Lambda_k, \mathcal{X}_k], a_{k:k+L}) = \mathbb{E}_{\{l\Gamma_{k+1:k+L}\}, Z^g_{k+1:k+L}}(\sum_{i=1}^{l} r(b[\Lambda_{k+i}, \mathcal{X}_{k+i}], a_{k+i})), \tag{36}$$

where $b[\Lambda_k, \mathcal{X}_k]$ is obtained by MH from Sec. 3.2.2, and

$$b[\Lambda_{k+i}, \mathcal{X}_{k+i}] = \mathbb{P}(\Lambda_{k+i}, \mathcal{X}_{k+i} | \{l\Gamma_{k+1:k+i}\}, Z^g_{k+1:k+i}, I_{1:k}, \mathcal{H}^g_k, D). \tag{37}$$

As each $l\Gamma_{k+i,w}$ consists of separate $l\gamma^o_{k+i,w}$, and similarly Z^g_{k+i} consists of $z^{g,o}_{k+i}$, MH-BSP first predicts which objects will be observed at time $k+i$. This can be done using an object observation model (see e.g. [5]) and sampled robot and object poses (either by sampling all objects or using a heuristic, see e.g. [11]). These objects are included in the predicted O_{k+i} set, and form $\mathcal{X}^{inv}_{k+1} \triangleq x_{k+1} \cup \{x^o\}_{o \in O_{k+1}}$.

To present the generative model, first consider the generation of $\{l\Gamma_{k+1}\}$ and Z^g_{k+1} from $b[\Lambda_k, \mathcal{X}_k]$ conditioned on action a_k. We follow the formulation in Eq. (13), and expand upon $\mathbb{P}(\mathcal{Z}_{k+1}|\mathcal{H}_k, a_k)$ for the specific case of MH-BSP. As such, the following marginalization scheme describes the sampling hierarchy:

$$\mathbb{P}(\{l\Gamma_{k+1}\}, Z^g_{k+1}|\mathcal{H}_k, a_k) = \sum_C \int_{\Lambda_k, \mathcal{X}_{k+1}} \mathcal{L}_{k+1} \cdot \mathbb{P}(C|\Lambda_k) \cdot \mathcal{M}_{k+1} \cdot b[\Lambda_k, \mathcal{X}_k] d\Lambda_k d\mathcal{X}_{k+1}, \tag{38}$$

which induces the following sampling hierarchy, for every object $o \in O_{k+1}$:

$$w \sim \mathbb{P}(w|D) \tag{39}$$

$$\mathcal{X}_{k+1} \sim \mathcal{M}_{k+1}\mathbb{P}(\mathcal{X}_k|\Lambda_{k,w}, l\gamma_{1:k,w}, \mathcal{H}^g_k) \tag{40}$$

$$C \sim Cat(\Lambda_{k,w}) \tag{41}$$

$$z^{g,o}_{k+1} \sim \mathbb{P}(z^{g,o}_{k+1}|x^{rel,o}_{k+1}) \tag{42}$$

$$\{l\gamma^o_{k+1}\} \sim \mathbb{P}(l\gamma^o_{k+1}|c^o, x^{rel,o}_{k+1}), \tag{43}$$

where $x^{rel,o}_{k+1} \triangleq x^o \ominus x_{k+1}$, and is determined by $\mathcal{X}^{inv}_{k+1}$. Recall that O_{k+1} must be determined by sampling $\mathcal{X}_{k+1}$. First, w is sampled uniformly from $|W|$. From there, as $b[\Lambda_k]$ is already represented by a set of samples $\{\Lambda_{k,w}\}$, sampling w chooses $\Lambda_{k,w}$ as well. Next, definition (23) for $hb_w[\mathcal{X}_k, C]$, and that $\Lambda_{k,w}$ is chosen:

Algorithm 1 MH-BSP Measurement Generation.

```
Input: Belief $b[\Lambda_k, \mathcal{X}_k]$, action $a_k$
 1: $b[\Lambda_k, \mathcal{X}_{k+1}] \leftarrow \mathcal{M}_k \cdot b[\Lambda_k, \mathcal{X}_k]$
 2: $\Lambda_k, \mathcal{X}_{k+1} \leftarrow$ Sample($b[\Lambda_k, \mathcal{X}_{k+1}]$)
 3: $O_{k+1} \leftarrow$ PredictObs($\mathcal{X}_{k+1}$)
 4: $\mathcal{X}^{inv}_{k+1} \leftarrow O_{k+1}, \mathcal{X}_{k+1}$
 5: $C \leftarrow$ Sample($Cat(\Lambda_k)$)
 6: $Z^g_{k+1} \leftarrow \emptyset$
 7: $\{l\Gamma_{k+1}\} \leftarrow \emptyset$
 8: for $o \in O_{k+1}$ do
 9:     $x^{rel,o}_{k+1} \leftarrow x^o, x_{k+1} \in \mathcal{X}^{inv}_{k+1}$
10:     $c^o \leftarrow C$
11:     $z^{g,o}_{k+1} \leftarrow$ Sample($\mathbb{P}(z^{g,o}_{k+1}|x^{rel,o}_{k+1})$)
12:     $Z^g_{k+1} \leftarrow Z^g_{k+1} \cup z^g_{k+1}$
13:     $\{l\gamma^o_{k+1}\} \leftarrow \emptyset$
14:     for $w \in W$ do
15:         $\{l\gamma^o_{k+1}\} \leftarrow \{l\gamma^o_{k+1}\} \cup$ Sample($\mathbb{P}(l\gamma^o_{k+1}|c^o, x^{rel,o}_{k+1})$)
16:     end for
17:     $\{l\Gamma_{k+1}\} \leftarrow \{l\Gamma_{k+1}\} \cup \{l\gamma^o_{k+1}\}$
18: end for
19: return $Z^g_{k+1}, \{l\Gamma_{k+1}\}$
```

Algorithm 2 MH-BSP Objective Function.

```
Input: MH Belief $b[\Lambda_k, \mathcal{X}_k]$, a set of actions $a_{k:k+L}$
 1: $J \leftarrow 0$
 2: for number of samples $N_s$ do
 3:     $Z^g_{k+1}, \{l\Gamma_{k+1}\} \leftarrow$
            MH Measurement Generation($b[\Lambda_k, \mathcal{X}_k], a_k$)(Alg. 1)
 4:     $b[\Lambda_{k+1}, \mathcal{X}_{k+1}] \leftarrow$ UpdateMH($b[\Lambda_k, \mathcal{X}_k], \{l\Gamma_{k+1}\}, Z^g_{k+1}, a_k$)
 5:     $r(b[\Lambda_{k+1}, \mathcal{X}_{k+1}]) \leftarrow$ Reward($b[\Lambda_{k+1}, \mathcal{X}_{k+1}]$
 6:     $J \leftarrow J + r(b[\Lambda_{k+1}, \mathcal{X}_{k+1}])/N_s$
 7:     if $L \neq 0$ then
 8:         $J \leftarrow J$
            +MH Objective Function($b[\Lambda_{k+1}, \mathcal{X}_{k+1}], a_{k+1:k+L})/N_s$
 9:     end if
10: end for
11: return $J$
```

$$\mathbb{P}(\mathcal{X}_k|\Lambda_{k,w}, l\gamma_{1:k,w}, \mathcal{H}^g_k) = \sum_C hb_w[\mathcal{X}_k, C]. \tag{44}$$

Then, $\mathbb{P}(\mathcal{X}_k|\Lambda_{k,w}, l\gamma_{1:k,w}, \mathcal{H}^g_k)$ is propagated via:

$$\mathbb{P}(\mathcal{X}_{k+1}|\Lambda_{k,w}, l\gamma_{1:k,w}, \mathcal{H}^g_k, a_k) = \sum_C \mathcal{M}_{k+1} hb_w[\mathcal{X}_k, C], \tag{45}$$

and $\mathcal{X}_{k+1}$ is sampled, from there we determine O_{k+1}.

For each object $o \in O_{k+1}$, MH-BSP determines the appropriate $x^{rel,o}_{k+1}$, and generates its own geometric measurement $z^{g,o}_{k+1}$. Next, class realization C is sampled; as the action a_k alone does not change Λ from time k to $k+1$ without measurements, $\Lambda_{k,w}$ is used to sample C. As such, C is a categorical random variable with the probability vector $\Lambda_{k,w}$ as its parameters. Finally, with $c \in C$ and $x^{rel,o}_{k+1}$, MH-BSP samples a set of $|W|$ vectors $l\gamma^o_{k+1}$.

Often planning algorithms use Maximum Likelihood (ML) estimation to reduce computational effort compared to sampling. In our case, taking the ML estimation of C can be problematic because it only considers the most likely class realization, ignoring all possible others.

For the following time steps, MH-BSP uses the generated $\{l\Gamma_{k+1}\}$ and Z^g_{k+1} to infer $b[\Lambda_{k+1}, \mathcal{X}_{k+1}]$ using MH inference from Sec. 3.2.2. Now using action a_{k+1}, MH-BSP generates $\{l\Gamma_{k+2}\}$ and Z^g_{k+2}, then $b[\Lambda_{k+2}, \mathcal{X}_{k+2}]$, and continues generating measurements and inferring corresponding belief until the end of the planning horizon. Algorithm 1 presents the MH-BSP measurement generation algorithm, where the function *PredictObs* predicts which objects are observed given sampled camera and object poses.

Algorithm 2 summarizes MH-BSP objective function computation, where the function *UpdateHB* is the hybrid belief update approach presented in Sec. 3.2.2, and *InferDist* infers $b[\Lambda_{k+1}, \mathcal{X}_{k+1}]$ from measurement generated in Algorithm 1. Algorithm 2 recursively calls itself until the action set only includes one action, allowing non-myopic planning.

4.2. Joint Lambda Pose planning (JLP-BSP)

This section presents JLP-BSP, an epistemic uncertainty aware semantic BSP framework that leverages JLP from Section 3.3 as the inference engine. If the assumption in Lemma 1 is exactly or approximately satisfied, we can utilize JLP for planning. As in inference, JLP-BSP is computationally faster than MH-BSP with a tradeoff of reduced accuracy.

Similarly to MH-BSP, the generation of new measurements is discussed. As described in Sec. 4.1, MH-BSP uses the classifier uncertainty model (17) parameters $h_i(x^{rel}_{k+1})$ and $\Sigma_i(x^{rel}_{k+1})$ to generate $\{l\gamma_{k+1}\}$ given class $c=i$ and x^{rel}_{k+1}. On the other hand, JLP-BSP does not require generating $\{l\gamma_{k+1}\}$ and elegantly uses $h_i(x^{rel}_{k+1})$ and $\Sigma_i(x^{rel}_{k+1})$ as generated measurements. With this, the objective function takes the following form:

$$J(b[l\bar{\lambda}_k,\mathcal{X}_k],a_{k:k+L})=\mathbb{E}_{\mathbb{E}(l\Gamma_{k+1:k+L}),\Sigma(l\Gamma_{k+1:k+L}),Z^g_{k+1:k+L}}\left(\sum_{i=1}^{L}r(b[l\bar{\lambda}_k,\mathcal{X}_{k+i}],a_{k+i})\right),\tag{46}$$

where,

$$b[l\bar{\lambda}_{k+i},\mathcal{X}_{k+i}]=\mathbb{P}(l\bar{\lambda}_{k+i},\mathcal{X}_{k+i}|\mathbb{E}(l\Gamma_{k+1:k+L}),\Sigma(l\Gamma_{k+1:k+L}),Z^g_{k+1:k+i},I_{1:k},\mathcal{H}^g_k,D),\tag{47}$$

with $\mathbb{E}(l\Gamma_k)\triangleq\{\mathbb{E}(l\gamma^o_k)\}_{o\in O_k}$ and similarly $\Sigma(l\Gamma_k)\triangleq\{\Sigma(l\gamma^o_k)\}_{o\in O_k}$

As in Sec. 4.1, we consider measurement generation for time $k+1$ from time k. Likewise, we follow the formulation in Eq. (13), and expand upon $\mathbb{P}(\mathcal{Z}_{k+1}|\mathcal{H}_k,a_k)$ for the specific case of JLP-BSP. This time, we present a sampling hierarchy that is described by the following marginalization scheme:

$$\begin{aligned}&\mathbb{P}(\mathbb{E}(l\Gamma_{k+1}),\Sigma(l\Gamma_{k+1}),Z^g_{k+1}|\mathcal{H}_k,a_k)\\&\quad=\int_{\mathcal{X}_{k+1},l\bar{\lambda}_k}\prod_{o\in O_{k+1}}\mathbb{P}(\mathbb{E}(l\gamma^o_{k+1}),\Sigma(l\gamma^o_{k+1})|l\lambda^o_k,\mathcal{H}_k,a_k)\cdot\mathbb{P}(z^{g,o}_{k+1}|\mathcal{X}_{k+1})\cdot b[l\bar{\lambda}_k,\mathcal{X}_{k+1}]d\mathcal{X}_{k+1}.\end{aligned}\tag{48}$$

Using the above equation, the generative model used to generate measurements for every $o\in O_{k+1}$ is written. First, JLP-BSP determines the set O_{k+1}, and samples the hypothesized object class c^o from $l\lambda^o_k\in l\bar{\lambda}_k$. It is done by sampling $\mathcal{X}_{k+1}$ and $l\bar{\lambda}_k$ using $b[l\bar{\lambda}_k,\mathcal{X}_k]$ as follows:

$$l\bar{\lambda}_k,\mathcal{X}_{k+1}\sim\mathcal{M}_{k+1}\cdot b[l\bar{\lambda}_k,\mathcal{X}_k].\tag{49}$$

Similar to MH-BSP, $l\bar{\lambda}_k$ stays the same conditioned on a_k, thus not propagated. Then JLP-BSP determines O_{k+1}, $\mathcal{X}^{inv}_{k+1}$ and $x^{rel,o}_{k+1}$ per object as in Sec. 4.1. Subsequently, for $o\in O_{k+1}$, JLP-BSP samples c^o and afterwards generates the measurements:

$$c^o\sim Cat(\lambda^o_k)\tag{50}$$

$$z^g_{k+1}\sim\mathbb{P}(z^g_k|x^{rel,o}_{k+1})\tag{51}$$

$$\mathbb{E}(l\gamma^o_{k+1})=h_c(x^{rel,o}_{k+1})\tag{52}$$

$$\Sigma(l\gamma^o_{k+1})=\Sigma_c(x^{rel,o}_{k+1}),\tag{53}$$

where $h_c(x^{rel,o}_{k+1})$ and $\Sigma_c(x^{rel,o}_{k+1})$ are the Gaussian parameters of $\mathbb{P}(l\gamma^o_{k+1}|c,x^{rel,o}_{k+1})$, as in Eq. (17).

Algorithm 3 presents the JLP measurement generation algorithm, where $\mathbb{E}(l\Gamma_k)\triangleq\{\mathbb{E}(l\gamma^o_k)\}_{o\in O_k}$, and similarly $\Sigma(l\Gamma_k)\triangleq\{\Sigma(l\gamma^o_k)\}_{o\in O_k}$. Algorithm 4 presents the objective function computation approach. *UpdateJLP* refers to updating $b[l\bar{\lambda}_k,\mathcal{X}_k]$ as in Sec. 3.3 given generated measurements. Algorithm 4 calls itself recursively until there is only one action left in the set. The algorithm is similar to Algorithm 2, except for the measurement generation and update functions specific to JLP.

4.3. Reward functions over $b[\lambda,\mathcal{X}]$

Predicting future $b[\lambda_{k+i},\mathcal{X}_{k+i}]$ at a future time $k+i$ allows us to consider multiple reward functions, all captured by the general formulation $r(b[\lambda,\mathcal{X}])$. To the best of our knowledge, we are the first to consider reasoning about *future posterior epistemic uncertainty* within a BSP setting. For rewards based on the poses $r(\mathcal{X})$ e.g. distance-to-goal, or rewards based on the belief over the poses $r(b[\mathcal{X}])$ e.g. information-theoretic costs, we can compute the marginal $b[\mathcal{X}_{k+i}]$ as for MH, or by marginalizing out $l\lambda_{k+i}$ from $b[\lambda_{k+i},\mathcal{X}_{k+i}]$ for JLP.

In addition, a reward over the posterior class probability $r(\mathbb{P}(c|\mathcal{H}))$ may also be considered, which can be extracted by computing $\mathbb{E}(\lambda_{k+i})$ from the marginal $b[\lambda_{k+i}]$:

Algorithm 3 JLP-BSP Measurement Generation.

```
Input: Belief $b[l\bar{\lambda}_k, \mathcal{X}_k]$, action $a_k$
 1: $b[l\bar{\lambda}_k, \mathcal{X}_{k+1}] \leftarrow \mathcal{M}_k \cdot b[l\bar{\lambda}_k, \mathcal{X}_k]$
 2: $l\bar{\lambda}_k, \mathcal{X}_{k+1} \leftarrow$ Sample$(b[\Lambda_k, \mathcal{X}_{k+1}])$
 3: $O_{k+1} \leftarrow$ PredictObs$(\mathcal{X}_{k+1})$
 4: $\mathcal{X}^{inv}_{k+1} \leftarrow O_{k+1}, \mathcal{X}_{k+1}$
 5: $Z^g_{k+1} \leftarrow \emptyset$
 6: $\mathbb{E}(l\Gamma_{k+1}) \leftarrow \emptyset$
 7: $\Sigma(l\Gamma_{k+1}) \leftarrow \emptyset$
 8: for $o \in O_{k+1}$ do
 9:     $x^{rel}_{k+1} \leftarrow x^o, x_{k+1} \in \mathcal{X}^{inv}_{k+1}$
10:     $c^o \leftarrow$ Sample$(Cat(\lambda^o_k))$
11:     $z^{g,o}_{k+1} \leftarrow$ Sample$(\mathbb{P}(z^{g,o}_{k+1} | x^{rel}_{k+1}))$
12:     $Z^g_{k+1} \leftarrow Z^g_{k+1} \cup z^g_{k+1}$
13:     $\mathbb{E}(l\gamma^o_{k+1}) \leftarrow h_c(x^{rel}_{k+1})$
14:     $\mathbb{E}(l\Gamma_{k+1}) \leftarrow \mathbb{E}(l\Gamma_{k+1}) \cup \mathbb{E}(l\gamma^o_{k+1})$
15:     $\Sigma(l\gamma^o_{k+1}) \leftarrow \Sigma_c(x^{rel}_{k+1})$
16:     $\Sigma(l\Gamma_{k+1}) \leftarrow \Sigma(l\Gamma_{k+1}) \cup \Sigma(l\gamma^o_{k+1})$
17: end for
18: return $Z^g_{k+1}, \mathbb{E}(l\Gamma_{k+1}), \Sigma(l\Gamma_{k+1})$
```

Algorithm 4 JLP-BSP Objective Function.

```
Input: JLP Belief $b[l\bar{\lambda}_k, \mathcal{X}_k]$, a set of actions $a_{k:k+l}$
 1: $J \leftarrow 0$
 2: for number of samples $N_s$ do
 3:     $Z^g_{k+1}, \mathbb{E}(l\Gamma_{k+1}), \Sigma(l\Gamma_{k+1}) \leftarrow$
            JLP Measurement Generation$(b[l\bar{\lambda}_k, \mathcal{X}_k], a_{k:k+l})$(Alg. 3)
 4:     $b[l\bar{\lambda}_{k+1}, \mathcal{X}_{k+1}] \leftarrow$ UpdateJLP$(b[l\bar{\lambda}_k, \mathcal{X}_k]),$
            $Z^g_{k+1}, \mathbb{E}(l\Gamma_{k+1}), \Sigma(l\Gamma_{k+1}))$
 5:     $r(b[l\bar{\lambda}_{k+1}, \mathcal{X}_{k+1}]) \leftarrow$ Reward$(b[l\bar{\lambda}_{k+1}, \mathcal{X}_{k+1}]$
 6:     $J \leftarrow J + r(b[l\bar{\lambda}_{k+1}, \mathcal{X}_{k+1}])/N_s$
 7:     if $l \neq 0$ then
 8:         $J \leftarrow J$
            $+$JLP Objective Function$(b[l\bar{\lambda}_{k+1}, \mathcal{X}_{k+1}], a_{k+1:k+l})/N_s$
 9:     end if
10: end for
11: return $J$
```

$$\mathbb{P}(c \mid \{l\gamma_{k+1:k+i}\}, z^g_{k+1,k+i}, I_{1:k}, \mathcal{H}^g_k, D) = \int_{\lambda_{k+i}} \mathbb{P}(c|\lambda_{k+i}) \cdot b[\lambda_{k+i}] d\lambda_{k+i} = \mathbb{E}(\lambda_{k+i}), \tag{54}$$

therefore we can write $r(\mathbb{P}(c|\mathcal{H}))$ as $r(\mathbb{E}(\lambda))$. An example of such reward is the negative of Shannon Entropy, such that $r(\mathbb{E}(\lambda)) = \sum_c \lambda^c \log(\lambda^c)$. This reward favors class probability vectors when one of the candidates has a probability close to one, and others close to zero. Crucially, as $b[\Lambda_{k+i}, \mathcal{X}_{k+i}]$ for MH-BSP and $b[l\bar{\lambda}_{k+i}, \mathcal{X}_{k+i}]$ for JLP-BSP both reason about epistemic uncertainty, it affects implicitly every reward. Thus, we account for future posterior epistemic uncertainty indirectly in all the cases discussed in this section.

4.4. Information-theoretic reward over $b[\lambda]$

In Sec. 4.3 we discussed reward functions in the form of $r(b[\mathcal{X}])$ and $r(\mathbb{P}(c|\mathcal{H}))$. But crucially, maintaining $b[\lambda_{k+i}, \mathcal{X}_{k+i}]$ opens the possibility of planning directly over $b[\lambda]$. We consider info-theoretical rewards over λ in the form of $r(b[\lambda])$. Specifically, we consider the differential entropy of λ_{k+i}, denoted $H(\lambda_{k+i})$, and is defined as:

$$H(\lambda_{k+1}) \triangleq - \int_{\lambda_{k+1}} b[\lambda_{k+1}] \cdot \log b[\lambda_{k+1}] d\lambda_{k+1}. \tag{55}$$

The reward considered is the minus of the entropy, i.e. $r(b[\lambda]) = -H(\lambda)$, which, as we will see in Sec. 4.5 (and Appendix A.8), is dependent both on $\mathbb{E}(\lambda)$ and the epistemic model uncertainty.

A possible alternative is a reward of the following general form for λ (see e.g. [27]):

$$r(b[\lambda]) = \omega_1 \cdot f_1(\mathbb{E}(\lambda)) + \omega_2 \cdot f_2(\Sigma(\lambda)), \tag{56}$$

where ω_1 and ω_2 are hyperparameters, and f_1 and f_2 are general functions. Here λ can be interchangeable with its logit transformation $l\lambda$. This reward requires the tuning of ω_1 and ω_2 manually, as opposed to using $r(b[\lambda]) = -H(\lambda)$, which does not require parameter tuning at all. In particular, as seen in Sec. 4.5, $H(\lambda)$ addresses both $\mathbb{E}(\lambda)$ and $\Sigma(\lambda)$ simultaneously. $H(\lambda)$ diminishes (i.e., $r(b[\lambda])$ grows) when $\mathbb{E}(\lambda)$ is closer to the simplex corners, i.e., when one category has its probability close to 1 and the rest close to 0. Also, $H(\lambda)$ diminishes the smaller $\Sigma(\lambda)$ becomes, which corresponds to smaller epistemic uncertainty.

However, computing $H(\lambda_{k+i})$ requires the PDF value of $b[\lambda]$, according to Eq. (55), thus necessitates modeling the distribution of λ_{k+i}. This distribution can be either parametric e.g. Dirichlet or LG, which we will discuss here, or non-parametric such as Kernel Density Estimation (KDE). `MH` provides $\{\lambda\}$, therefore any distribution that supports probability vectors can be chosen. On the other hand, `JLP` limits λ to be LG distributed per definition. Sec. 4.5 details using the Logistical Gaussian distribution for $b[\lambda_k]$ in the context of computing entropy. To simplify notations, all of the variables in the next section are considered at the same time step, the time index is dropped. In addition, we use the single-object notation, i.e. λ and c.

4.5. Logistic Gaussian for $b[\lambda]$

$b[\lambda_k]$ can be modeled as Logistic Gaussian (LG) distributed, which is supported by both `MH` and `JLP`, as illustrated in Fig. 7. This distribution (with PDF as in Eq. (18)) supports probability vectors with conditions presented in Sec. 2.2 for γ, thus samples from LG are probability vectors. This distribution does not have an analytical expression for expectation and covariance, and must be computed numerically or approximated, e.g. via bounds, as discussed later.

To compute the parameters from a point cloud of probability vectors, e.g. $\{\lambda\}$, the logit transformation is applied on $\lambda \in \{\lambda\}$ to get $\{l\lambda\}$. Then, as $l\gamma$ is modeled Gaussian the LG parameters $\mathbb{E}(l\lambda)$ and $\Sigma(l\lambda)$ are inferred. In addition to expectation and covariance, the LG distribution does not have a closed-form solution for its differential entropy. However, the LG variable is a transformation of a Gaussian variable with a known expression for entropy. As such, the entropy is expressed using the following lemma.

Lemma 2. *Let $\lambda = [\lambda^1, ..., \lambda^m]^T$ be Logistical-Gaussian distributed, and $l\lambda$ its logit transformation as in Eq. (16), thus $l\lambda$ is Gaussian with parameters $\mathbb{E}(l\lambda)$ and $\Sigma(l\lambda)$. As such, the differential entropy $H(\lambda)$ is described by:*

$$H(\lambda) = H(l\lambda) + \sum_{i=1}^{m-1} \mathbb{E}[l\lambda^i] - \int_{l\lambda} \log\left(1 + \sum_{i=1}^{m-1} e^{l\lambda^i}\right) \mathbb{P}(l\lambda) dl\lambda. \tag{57}$$

The complete proof is shown at Appendix A.6.

As $l\lambda$ is Gaussian, $H(l\lambda) = 0.5 \cdot \log(2\pi e|Cov(l\lambda)|)$. The integral in Eq. (57) does not have an analytical solution to the best of our knowledge. One approach is to compute the entropy numerically from $\{\lambda\}$ that is already available, but it is computationally expensive to do so for a large number of candidate classes. Another option is to compute bounds for the entropy, presented in the following lemma.

Lemma 3. *Let $\lambda = [\lambda^1, ..., \lambda^m]^T$ be Logistical-Gaussian distributed, and $l\lambda$ its logit transformation as in Eq. (16), thus $l\lambda$ is Gaussian with parameters $\mathbb{E}(l\lambda)$ and $\Sigma(l\lambda)$. As such, an upper bound for $H(\lambda)$ is given by:*

$$H(\lambda) \leq H(l\lambda) + \sum_{i=1}^{m-1} \mathbb{E}(l\lambda^i) - m \cdot \max_i\{0, \mathbb{E}(l\lambda^i)\}, \tag{58}$$

and similarly, a lower bound is given by:

$$H(\lambda) \geq H(l\lambda) + \sum_{i=1}^{m-1} \mathbb{E}(l\lambda^i) - m \cdot \max_i\{0, \mathbb{E}(l\lambda^i)\} - m\log m - \sqrt{\frac{\sigma_{ii}^{max}}{2\pi}}, \tag{59}$$

where $\sigma_{ii}^{max} \triangleq \max_i \Sigma_{ii}(l\lambda)$ is the largest value element in the covariance of $l\lambda$.

The complete proof is shown at Appendix A.7.

One can observe from the upper bound that $H(l\lambda)$ is necessarily larger than $H(\lambda)$ as $l\gamma$ is not subjected to the probability vector constraints, thus $\mathbb{E}(l\gamma^i)$ can be negative for every i and $\sum_{i=1}^{m-1} \mathbb{E}(l\gamma^i) - m \cdot \max_i\{0, \mathbb{E}(l\gamma^i)\}$ is necessarily non-positive.

Fig. 8 presents the entropy values of $b[\lambda]$ as a function of its LG parameters $\mathbb{E}(l\lambda)$ and $Var(l\lambda)$ in two candidate classes case. As a probability vector of two candidate classes has a single degree of freedom, two parameters can fully describe the distribution. The figure shows that the farther $\mathbb{E}[l\lambda]$ is from zero, i.e. the closer $\mathbb{E}[\lambda^1]$ to either one or zero, the smaller the entropy gets generally. The effect is more pronounced when $Var(l\lambda)$ is small. If we minimize entropy during planning, the robot will aim to reach regions where $\mathbb{E}[\lambda]$ is close to the edges of the simplex, with smaller posterior epistemic uncertainty.

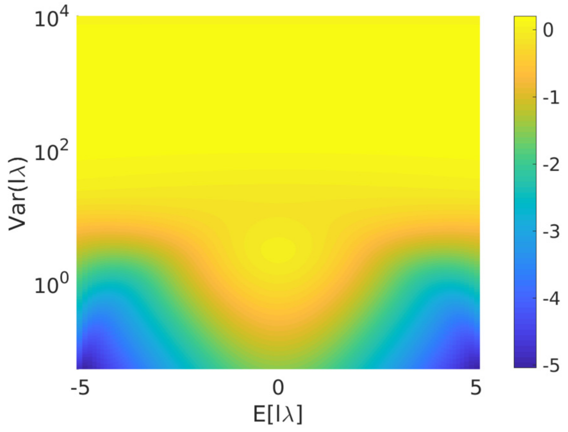


Fig. 8. Entropy of a one dimensional Logistical Gaussian that corresponds to two dimensional probability vector γ. The x and y axes represent $\mathbb{E}(l\gamma)$ and $Var(l\gamma)$ respectively. Blue to yellow colors correspond to low to high entropy.

The scenarios presented in Fig. 2 correspond to the following cases in Fig. 8:

- The unknown-unknown case (Fig. 2a) corresponds to $\mathbb{E}(l\lambda)$ close to 0, and large $Var(l\lambda)$, i.e. the upper central part of Fig. 8.
- The known-unknown case (Fig. 2b) corresponds to $\mathbb{E}(l\lambda)$ close to 0, and small $Var(l\lambda)$, i.e. the lower central part of Fig. 8.
- The known-known case (Fig. 2c) corresponds to $\mathbb{E}(l\lambda)$ with large absolute value, and small $Var(l\lambda)$, i.e. the lower areas at the sides.
- The uncertain classification case (Fig. 2d) corresponds to $\mathbb{E}(l\lambda)$ with large absolute value, and large $Var(l\lambda)$, i.e. the upper areas at the sides.

5. Evaluation

We evaluate our approaches for semantic SLAM inference and planning in simulation (Sec. 5.2) and an experiment (Sec. 5.3) over the Active Vision Dataset scenario Home-3-01 [30], with viewpoint dependent classifier uncertainty models trained using the BigBIRD dataset [35]. We considered environments with multiple spatially scattered objects, and the robot's task is to accurately classify them while localizing. Our implementation uses the GTSAM library [31] with a Python wrapper. The hardware used is an Intel i7-7700 processor running at 2.8 GHz and 16 GB RAM, with GeForce GTX 1050 Ti with 4 GB RAM.

5.1. Compared approaches and metrics

We consider three approaches for inference and planning: `MH` and `JLP` methods with the corresponding `MH-BSP` and `JLP-BSP`, and an approach that does not consider model uncertainty, denoted as Without Epistemic Uncertainty (`WEU`). In this approach, we maintain a single hybrid belief and use it for inference and planning, similar to approaches presented in [5,23].

Our approach is evaluated for classification accuracy using the Mean Square Detection Error metric (MSDE, also used by Teacy et al. [18] and Feldman & Indelman [20]). Given $b[\lambda_k]$, MSDE is defined as follows:

$$MSDE \triangleq \frac{1}{m}\sum_{i=1}^{m}\left(\lambda_{gt}^{i} - \mathbb{E}(\lambda_k^i)\right)^2, \tag{60}$$

where λ_{gt}^i is the ground truth probability of the object being of class $c = i$ and is equal to 1 if the object is class i and 0 otherwise. For a completely incorrect classification $MSDE \leq 1$, ideal classification $MSDE = 0$, and for classification results where all class probabilities are equal, $MSDE = \frac{m-1}{m^2}$.

5.2. Simulation

5.2.1. Simulation setting

We consider a closed set setting and assume $m = 2$ candidate classes. The camera senses objects up to 10 meters distance, with an opening angle of 120°. In the main paper, a model that satisfies Lemma 1 is chosen. In Appendix A.10 results are presented for a model that does not satisfy Lemma 1. The baseline MSDE score for $m = 2$ where all class probabilities

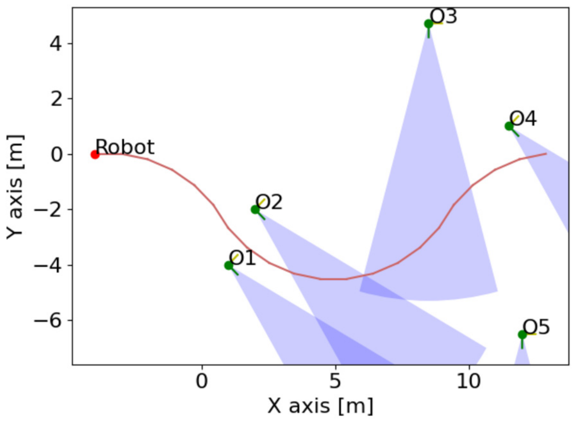


Fig. 9. The ground truth of the scenario in Sec. 5.2.2. The red dot represents the robot's starting point, with the red curve being the path. The green dots represent the objects' location with the corresponding object labels. The green line represents the object orientation, with the yellow line presenting 90° of that orientation. The blue cones represent the observation viewpoints in which the classifier identifies the object class well with low uncertainty, i.e. case 1.

are equal is $MSDE = 0.25$. We corrupt the simulated measurements, both geometric and semantic, with Gaussian noise. In the passive case, the robot's motion is deterministic. In the active case, while the motion primitives used are fixed, the motion is stochastic with a small Gaussian noise.

5.2.2. Inference: single run

The setting for this comparison is an environment with 5 objects. These objects are placed within the environment, presented in Fig. 9 along with the ground truth trajectory. The robot passes through an area where the objects have classification scores with high epistemic uncertainty. Normally, with methods that do not consider epistemic uncertainty, classification results will have a high chance of being incorrect, but our approach provides more accurate results as it considers epistemic uncertainty. Denote ψ as the relative orientation between the object's orientation (chosen during the classifier uncertainty model training) and the camera's pose. We simulate a classifier model that considers the following cases:

1. The classifier differentiates well between classes with low epistemic uncertainty, $\psi = 0°$.
2. The classifier does not differentiate well between the two classes, $\psi = 90°, 270°$.
3. The classifier differentiates between classes well, but with high epistemic uncertainty, $\psi = 180°$.

As such, $\psi = 0°$ is the relative orientation where the best classification with the lowest uncertainty is expected (corresponding to the blue cone in Fig. 9 that represents this relative orientation), and $\psi = 180°$ is the relative orientation that most prone to classification errors when not considering epistemic uncertainty. Considering the ground truth trajectory for the scenario, objects 1 and 2 represent case 3. As such, we expect our approaches to infer the correct class within a large number of steps because of the uncertainty. Object 3 represents case 1, and when it is observed the classification will be accurate on the first view. Objects 4 and 5 represent case 2, where classification is difficult as the model does not differentiate well between the classes of those objects. The object ground truth classes are $c = 1$ for objects 1, 2, and 5, and $c = 2$ for objects 3 and 4.

A visualization of the models presented can be seen in Fig. 10.

We consider noisy geometric measurements of relative pose (both for motion and geometric models), and cloud point semantic measurements, i.e. the classifier gives $\{\gamma\}$ per each object, sampled from the classifier uncertainty model. The classifier uncertainty model uses the following expectation function (see Eq. (17)):

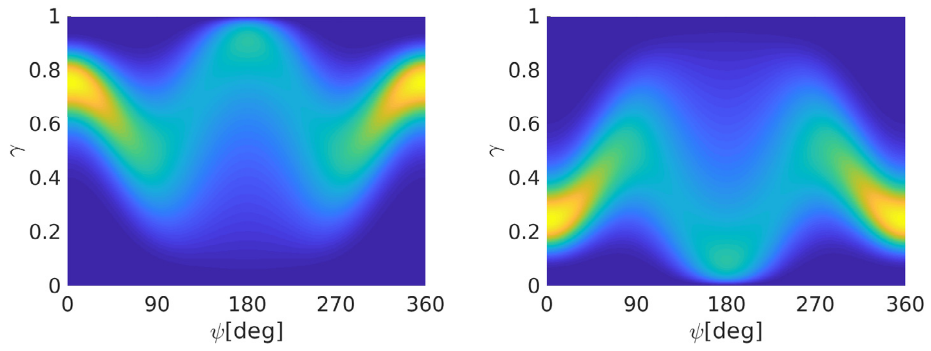
$$\begin{aligned} h_{c=1}(x^{rel}) &= \frac{1}{2}\cos(2\cdot\psi) + \frac{1}{2} \\ h_{c=2}(x^{rel}) &= -\frac{1}{2}\cos(2\cdot\psi) - \frac{1}{2}, \end{aligned} \tag{61}$$

and the following function for root-information:

$$R_{c=1}(x^{rel}) = R_{c=2}(x^{rel}) = 1.4 + 0.6\cdot\cos(\psi). \tag{62}$$

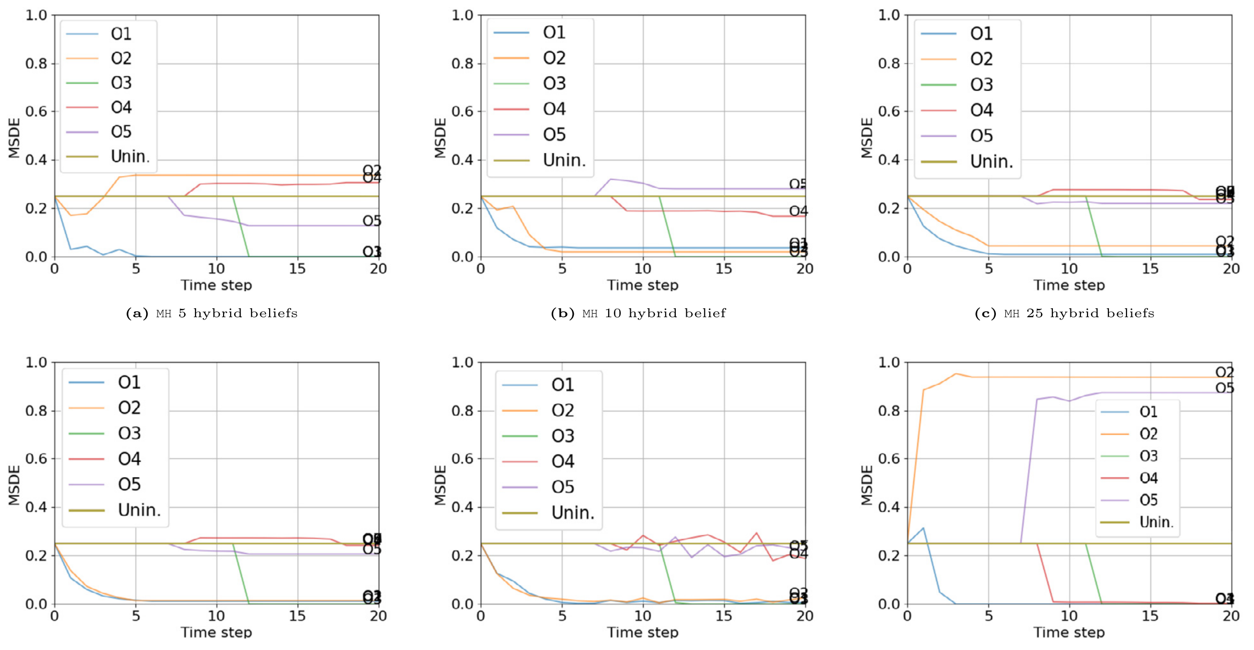
Subsequently, the covariance parameter from Eq. (17) in the two class case is computed as follows:

$$\Sigma_c(x^{rel}) = \sqrt{\frac{1}{R_c(x^{rel})}}. \tag{63}$$



(a) $\mathbb{P}(\gamma^{c=1}|c=1,\psi)$ (b) $\mathbb{P}(\gamma^{c=1}|c=2,\psi)$

Fig. 10. A visualization of the classifier uncertainty model used in Sec. 5.2.2. We present the value of $\mathbb{P}(\gamma^{c=1}|c, x^{rel}) \equiv \mathbb{P}(\gamma^{c=1}|c,\psi)$ as a function of relative orientation ψ and $\gamma^{c=1}$ value, for classes $c=1$ and $c=2$ in **(a)** and **(b)** respectively. Blue and yellow colors correspond to low and high PDF values respectively.



(a) MH 5 hybrid beliefs (b) MH 10 hybrid belief (c) MH 25 hybrid beliefs (d) MH 100 hybrid beliefs (e) JLP (f) WEU

Fig. 11. **(a)**, **(b)**, **(c)**, and **(d)** show MSDE results per time step for MH per object, each in a different color, for 5, 10, 25, and 100 respectively. **(e)** shows MSDE results for JLP. **(f)** shows MSDE results for WEU. In the legend, Unin. means uninformative classification, i.e. the case where $\mathbb{P}(c_1) = \mathbb{P}(c_2) = 0.5$.

With the covariance parameter being equal, the presented model satisfies the assumption of Lemma 1, allowing us to use the JLP approach.

Fig. 11 presents MSDE results for each object separately. We perform inference with MH with a different number of hybrid beliefs and compare it to JLP and WEU. With MH and JLP, the class of objects 1 and 2 is inferred using multiple observations, eventually inferring the correct class. The class of object 3, once seen, is quickly and accurately inferred. The class of objects 4 and 5 remain ambiguous (MSDE of approximately 0.25) because they are observed from viewpoints that correspond to case 2. In general, MH in Fig. 11a-11d tends to present smoother results the more hybrid beliefs are used and also compared to JLP in Fig. 11e where for each time step the entropy must be computed numerically from new λ_k samples. WEU in Fig. 11f shows that objects can be classified incorrectly if not considering epistemic uncertainty, such as object 4, as shown in the figure.

In summary, Fig. 12a presents average MSDE results for all the objects combined, showing that epistemic uncertainty aware approaches outperform WEU, while MH with 10 beliefs and JLP perform similarly. Fig. 12b presents a computation time comparison between WEU, JLP, and MH for different numbers of hybrid beliefs. From this figure, we can see that JLP is comparable to WEU, with MH being significantly more computationally intensive as the number of simultaneous beliefs increases.

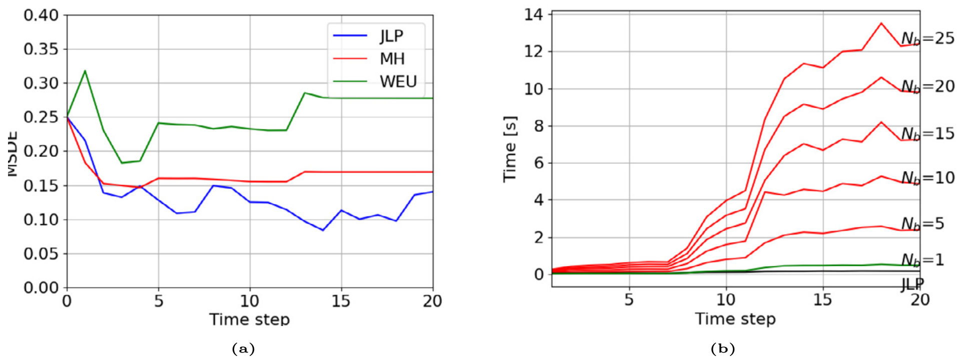


Fig. 12. (**a**) compares MSDE to time step between MH with 100 beliefs in red, JLP in blue, and without uncertainty in green. (**b**) compares run-time per inference step between realizations of MH with different number of hybrid beliefs in red, JLP in black, and WEU in green.

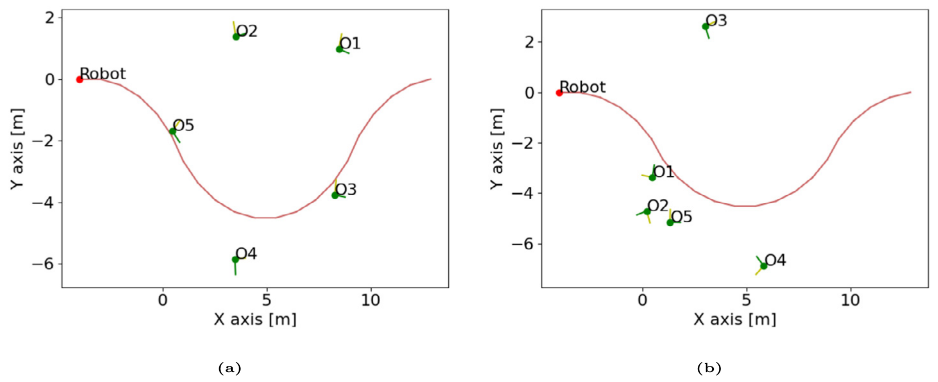


Fig. 13. Examples of a sampled environment in which the inference is performed. The red trajectory is the robot's path. The green dots denote the objects, numbered O1 to O5. The green and yellow lines represent their orientation, 0° and 90° respectively.

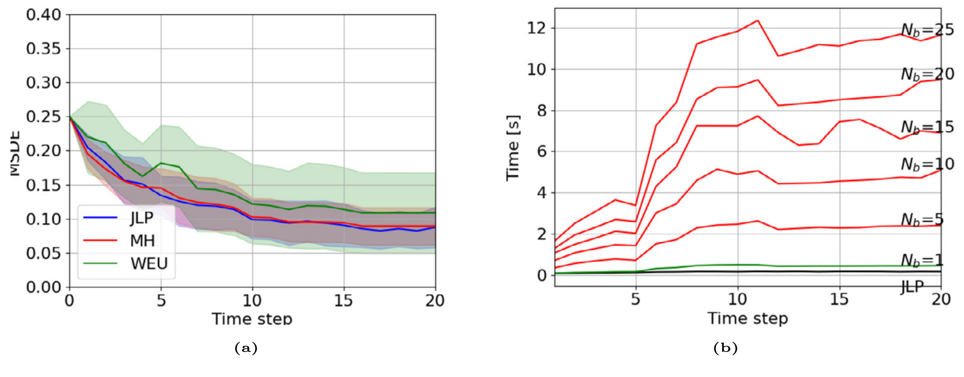
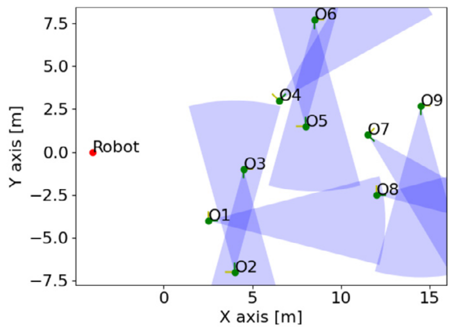


Fig. 14. (**a**) compares MSDE to time step between MH in red, JLP in blue, and without uncertainty in green. The transparent colors correspond to the respective plot in one σ value. (**b**) compares run-time per inference step between realizations of MH with different number of hybrid beliefs in red, JLP in black, and WEU in green.

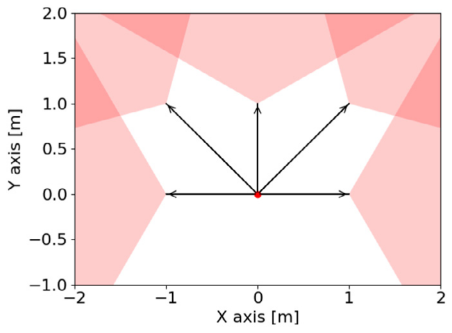
5.2.3. *Inference: statistical study*

We perform a Monte-Carlo study to compare between MH, JLP, and WEU. We run the simulation 10 times and present results for MSDE and computational time. The setting for this comparison is an environment with 5 objects with randomized poses, with examples presented in Fig. 13. Otherwise, the same setting and classifier uncertainty model are used as in Sec. 5.2.2.

We present MSDE statistical results in Fig. 14a, with one σ uncertainty. MH was performed with 100 beliefs. While MH and JLP perform similarly, both outperform the approach that does not consider epistemic uncertainty, especially in cases where the camera goes through areas that correspond to $\psi = 180°$. Fig. 14b presents run-time results for the algorithms. Expectedly, as the number of simultaneous beliefs increases for MH, the algorithm runs slower. JLP is comparable to main-



(a)



(b)

Fig. 15. **(a)** is the ground truth of the scenario in Sec. 5.2.4. The red dot represents the robot's starting point. The green dots represent the objects' location with the corresponding object labels. The green line represents the object orientation, with the yellow line presenting 90° of that orientation. The blue cones represent the observation angles in which the objects are classified most accurately with the lowest epistemic uncertainty, with 3 overlapping areas as low epistemic uncertainty areas. **(b)** presents the five motion primitives in the scenario. The red dot represents the origin point, the black arrows the possible actions, and the red cones represent the field of view after the action.

taining a single hybrid belief in this case, demonstrating that it is more practical when the conditions of Lemma 1 are satisfied. Appendix A.10 presents results for the case where the conditions of Lemma 1 are not satisfied.

5.2.4. Planning: single run

We simulate a planning scenario of 9 objects placed such that there are 3 zones of low uncertainty with high expected classification scores, as shown in Fig. 15a. We compare two reward functions for planning. R_1 is the negative of the entropy of λ as defined in Eq. (55), while R_2 is the entropy of $\mathbb{E}(\lambda)$ as defined in Sec. 4.3. Reward function R_1 is modified to include a cap of $R_{max} = 5$ per object to encourage exploration and classification of all objects in the scene. We modify R_1 and R_2 to include all objects by summing the entropy of each marginal λ_{k+1} per object. For a future belief of all $b[\lambda^o_{k+1}]$ per object o in the set of 9 objects:

$$\begin{aligned} R_1 &= \sum_o \min(-H(\lambda^o_{k+1}), R_{max}) \\ R_2 &= -\sum_o H(\mathbb{E}(\lambda^o_{k+1})) \end{aligned} \tag{64}$$

For both reward functions, we use `MH-BSP` and `JLP-BSP`. We use only R_2 for `WEU` as R_1 is not applicable because it does not consider epistemic uncertainty, while R_2 can use the posterior class probability as $\mathbb{E}(\lambda^o_{k+1})$. Optimally, the robot would plan to go through the high separation low uncertainty zone. We have five possible motion primitives, as represented in Fig. 15b with a vision cone of 120° emanating from the camera. The trajectory length is 20 time steps, at each time step trajectories are randomly sampled for a horizon $L = 10$. At each action sample, we consider the Maximum Likelihood Estimation of the propagated pose for measurement generation. When the horizon is reached, we perform the action at the current time that leads to the best reward. We perform re-planning after every action step, and the entire belief $b[\lambda, \mathcal{X}]$ is updated for each sampled action and estimated observation. While more efficient planning schemes exist (e.g. PFT-DPW [36]), considering state-of-the-art POMDP algorithms is beyond the scope of the paper. The setting for the classifier model, viewing radius and angle, motion, and geometric noise are the same as in the inference simulation. `MH-BSP` uses 10 hybrid beliefs. Optimally, the robot would plan to go through all three zones to achieve accurate classification of all objects.

Fig. 16 presents the trajectories created for all the methods. The ones that plan over R_1 create trajectories that pass closer to the overlapping low uncertainty areas from Fig. 15a, resulting in more accurate classification than planning over R_2 for all methods, especially `WEU`.

Fig. 17 presents a comparison for $H(\lambda_k)$ at the inference phase, when comparing planning over R_1, and R_2 for `MH-BSP` in Fig. 17a and `JLP-BSP` in Fig. 17b. In both figures planning over R_1 yields lower entropy, correlating to lower epistemic uncertainty. The effect is more noticeable for `MH-BSP` than `JLP-BSP`.

Fig. 18 presents MSDE results for all the methods, split into results for `MH-BSP` in Fig. 18a and for `JLP-BSP` in Fig. 18b, both showing a comparison to `WEU` in the green plot. Evidently, planning over R_1 slightly outperforms planning over R_2, with `WEU` lagging far behind.

Fig. 19 presents a bar graph with an error representation of the classification results at time $k = 20$ for all objects. Generally, planning over R_1 tends to have a more accurate classification than planning over R_2 with lower uncertainty. On the other hand, `WEU` tends to go towards extremes of class probabilities 0 or 1, whether it is the correct class.

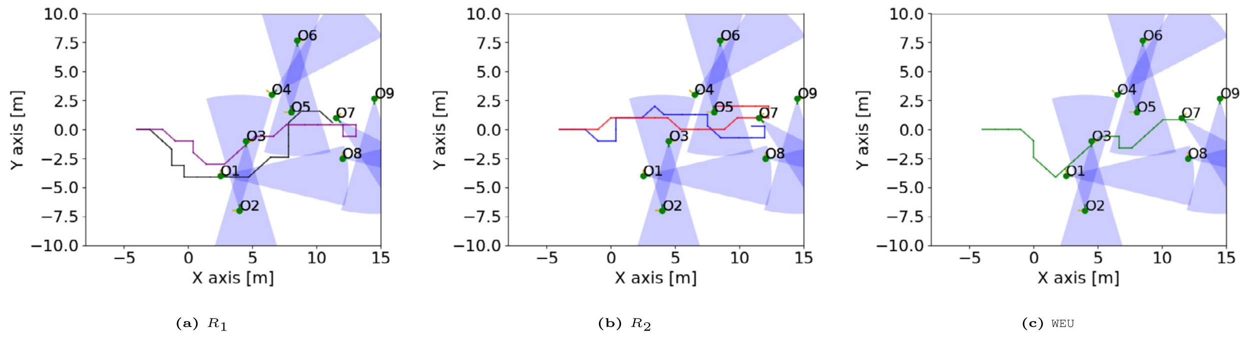


Fig. 16. This figure presents the ground truth of planned trajectories. **(a)** for planning over R_1 for MH-BSP (purple) and JLP-BSP (black). **(b)** for planning over R_2 for MH-BSP (red) and JLP-BSP (blue). **(c)** for WEU (green). All are for the multiple object scenario. The object is shown in a green dot, with the green line representing the object orientation, with the yellow line presenting 90° of that orientation. The blue cones represent the areas where observations have the lowest epistemic uncertainty.

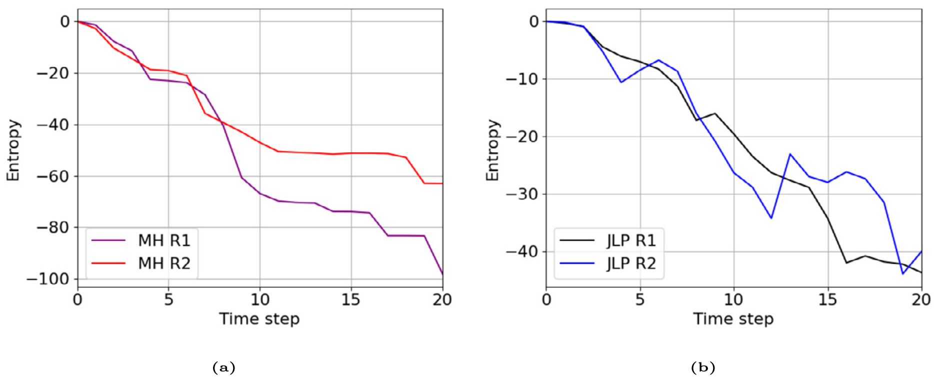


Fig. 17. $H(\lambda_{20})$ values for MH-BSP **(a)** and JLP-BSP **(b)** as a function of the time step. In **(a)**, the purple and red plots represent R_1 and R_2 respectively, and similarly in **(b)**, the black and blue plots represent R_1 and R_2 respectively.

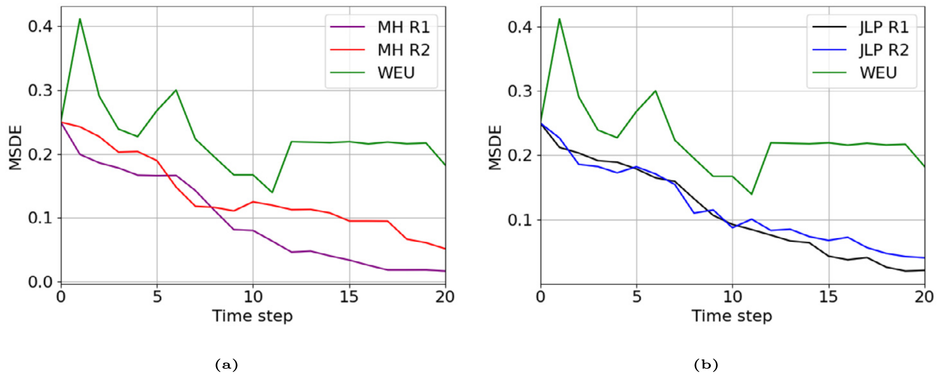


Fig. 18. Single-run study for multiple object scenario study for MSDE comparing planning over R_1 and R_2 for MH-BSP **(a)** and JLP-BSP **(b)**, and WEU.

Fig. 20 presents the computational time per step for all our approaches using R_2 reward function. Here MH-BSP uses 10 hybrid beliefs. This figure shows that JLP-BSP is slightly faster than WEU while also reasoning about posterior epistemic uncertainty, because the number of states in JLP-BSP scales linearly with the number of objects and candidate classes, as opposed to exponentially with WEU and MH-BSP. On the other hand, JLP-BSP is significantly more computationally efficient than MH-BSP.

See Appendix A.10.1 for additional results for planning around a single object.

5.2.5. Planning: statistical study

For the statistical study, we randomly corrupt geometric and semantic measurements with noise. We use the scenario from Sec. 5.2.4, using R_1, and R_2 with JLP-BSP, and compare it to WEU. We perform 10 iterations, each with a planning horizon $L = 10$, and present results for entropy and MSDE. Each run was performed up to 20 time steps.

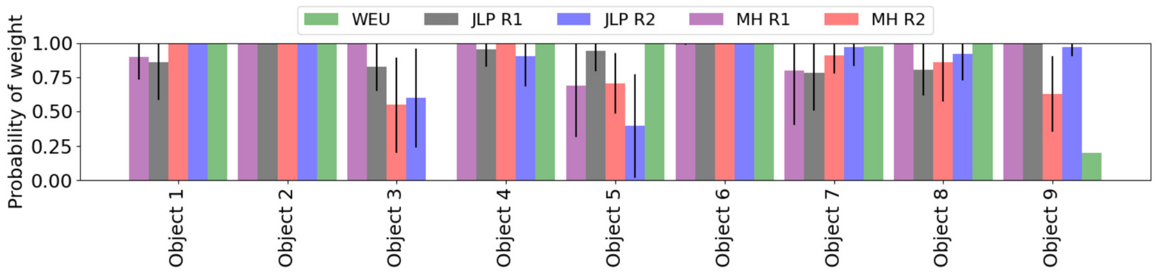


Fig. 19. Probability of the objects being ground truth class for our methods at time $k = 20$ for all objects. We compare planning over R_1 and R_2, `JLP-BSP`, `MH-BSP`, and `WEU`. Purple and red for R_1 and R_2 respectively using `MH`, black and blue for using for R_1 and R_2 respectively using `JLP-BSP`, and green for `WEU`. The one σ deviation is represented via the black line at each relevant bar, and represents the posterior model uncertainty.

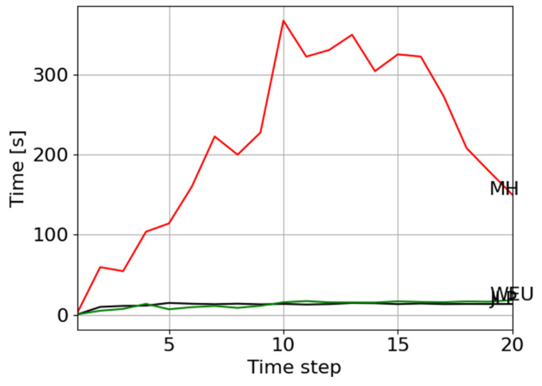


Fig. 20. This figure compares run-time per inference step between realizations of `MH-BSP` with 5 hybrid beliefs in red, `JLP-BSP` in black, and `WEU` in green.

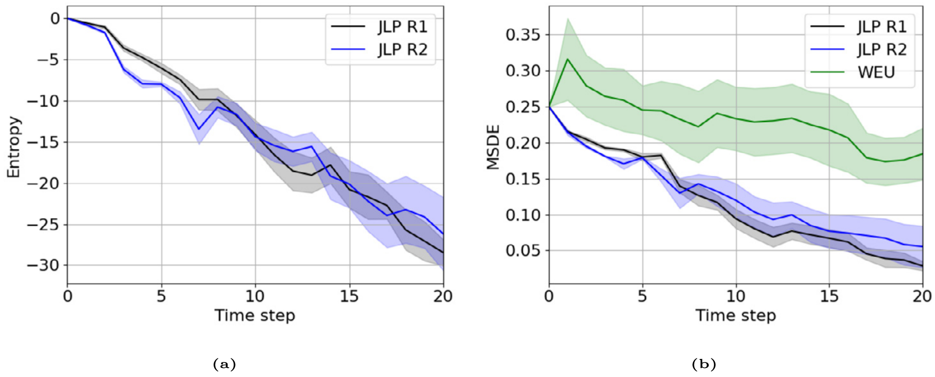


Fig. 21. Statistical study for the scenario in Sec. 5.2.5. **(a)** presents a comparison for the sum of entropy over all objects between trajectories for R_1 and R_2 as a function of the time step for `MH-BSP`. **(b)** presents an MSDE comparison between `MH-BSP`, `JLP-BSP`, and `WEU` as a function of time step. In both, the line represents the statistical expectation, while the colored area represents a one σ deviation.

Fig. 21 presents the statistical results for the sum of the entropy in Fig. 21a, and the MSDE results in Fig. 21b, with the colored areas representing one σ deviation. All in all, planning over R_1 performs better over planning over R_2 for `JLP-BSP`, with lower entropy and MSDE. In addition, MSDE results compared to `WEU` are vastly superior for epistemic-uncertainty-aware methods.

5.3. Experiment

5.3.1. Setup

For the experiment, we consider a myopic planning scenario in a semantic SLAM setting, using Active Vision Dataset (AVD) [30] Home 005 with example images presented in Fig. 22. In this scenario, the objects are grouped into two groups, one on a table near the window back-lit by sunlight as seen in Fig. 22a, and another on the kitchen counter seen in Fig. 22b. We perform planning for a 20 time step trajectory, at each step performing myopic planning. In this dataset, the candidate

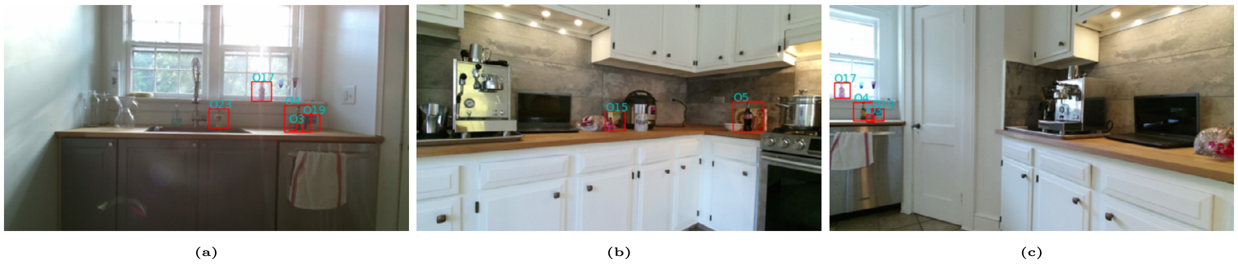


Fig. 22. Example images of the Active Vision Dataset, home 005. The red boxes represent the bounding boxes for the objects, and the notation Ox represents the x-th object.

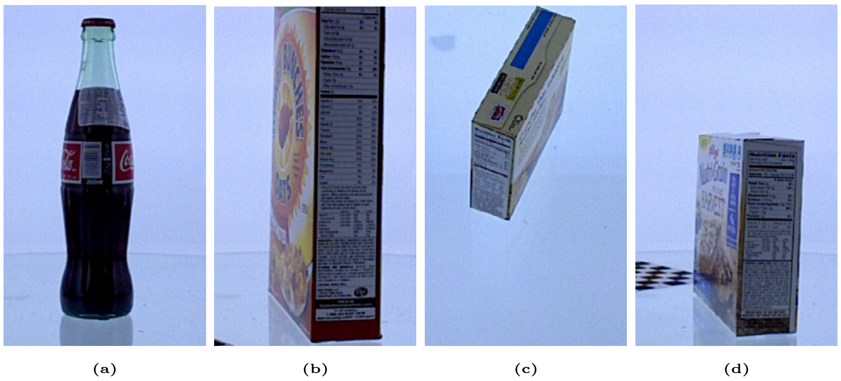


Fig. 23. Example images of the BigBIRD dataset for training the classifier models. **(a)** is an example for the "pop bottle" class, while the rest are examples of "packet".

actions are given for each image a-priori. We aim to compare JLP-BSP and WEU for classification accuracy using MSDE (60), differential entropy representing epistemic uncertainty, and computational time. The reward functions R_1 and R_2 are identical to those presented in Eq. (64).

Five candidate classes are considered: "Packet", "Book Jacket", "Pop Bottle", "Digital Clock", and "Soap Dispenser". For each class, we trained classifier uncertainty models using images from BigBIRD dataset [35], with example images presented in Fig. 23. For classification, we used VGG convolutional neural network [37] with dropout activated during test time. The R_1 upper limit R_{max} per object is 500, as the increasing number of objects increases the scale of R_1 values. Recall Lemma 2, the entropy depends on the covariance of $l\gamma$ via $H(l\lambda)$.

The classifier models were trained via PyTorch on fully connected networks with five hidden layers. Recall Eq. (17), we train $h_c(x^{rel})$ and $\Sigma_c(x^{rel})$ from a dataset $D_c = \{x^{rel}, \{l\gamma\}\}$ per object, where $x^{rel} = [\psi, \theta]$ is parametrized by relative yaw angle ψ and relative pitch angle θ. h_c and Σ_c are represented by separate neural networks, up to a total of $2m$ networks. As seen in Sec. 3.3, all $\Sigma_{c=i} \equiv \Sigma_{c=j}$ for $i, j = 1, ..., m-1$ for the JLP factor to be Gaussian. This constraint limits the expressibility of Σ_c, thus not accurately representing the epistemic uncertainty from certain viewpoints of objects. As such, instead of enforcing a hard constraint on all Σ_c, we train the classifier uncertainty model with a loss function that imposes a penalty if Σ_c for different c are not similar, enforcing a soft constraint.

The loss function L_h for the h_c network is mean square error (MSE):

$$L_h(h_c, \{l\gamma\}) = MSE(h_c, \{l\gamma\}) = \sum_{i=1}^{m} \left(h_c^i - \mathbb{E}(l\gamma^i)\right)^2, \tag{65}$$

where h_c^i is the i-th element of h_c. The loss function L_Σ for the Σ_c uses MSE over the covariance matrix elements and adds a Frobenius norm term that acts as the soft constraint that makes the values of Σ_c closer:

$$L_\Sigma(h_c, \Sigma_c, \{l\gamma\}) = MSE(\Sigma_c, \Sigma(l\gamma)) + \kappa \cdot F_N(h_c, \Sigma_c) \tag{66}$$

where the MSE for the above loss function is defined:

$$MSE(\Sigma_c, \Sigma(l\gamma)) = \frac{1}{(m-1)^2} \sum_{i=1}^{m} \sum_{j=1}^{m} ([\Sigma_c]_{ij} - [\Sigma(l\gamma)]_{ij})^2, \tag{67}$$

$F_N(\cdot)$ is the Frobenius Norm, defined:

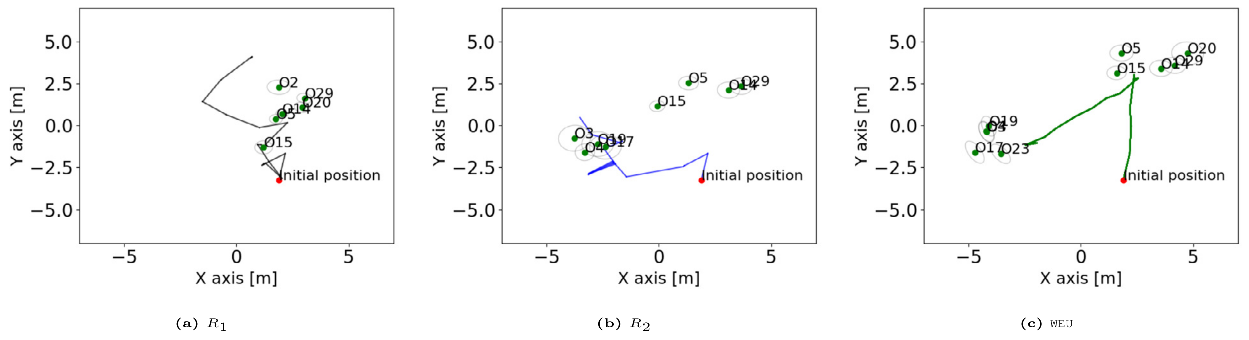


Fig. 24. This figure presents the ground truth of planned trajectories with object pose estimations. **(a)** for planning over R_1 for JLP-BSP in black. **(b)** for planning over R_2 for JLP-BSP in blue. **(c)** for WEU (green). All for the AVD scenario. The object estimation is shown in a green dot with corresponding estimation covariance of 3σ in gray. The red dots represent the starting position of each trajectory.

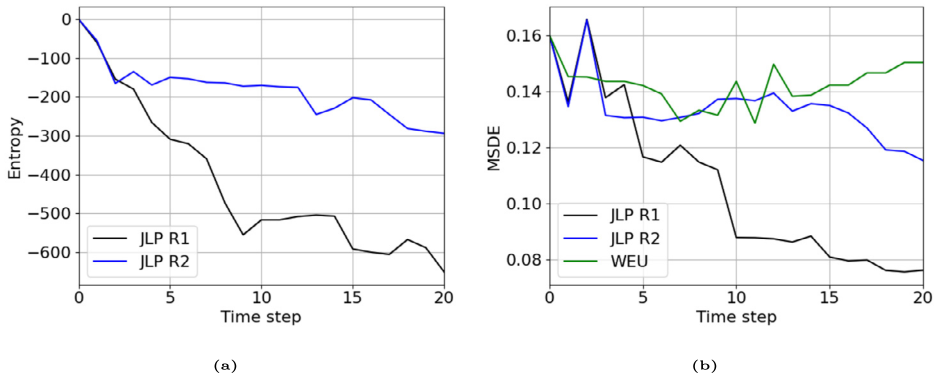


Fig. 25. Experimental results for the scenario in Sec. 5.3.1. **(a)** presents a comparison for the sum of entropy over all objects between trajectories for R_1 and R_2 as a function of the time step for JLP-BSP. **(b)** presents an MSDE comparison between JLP-BSP, and WEU as a function of time step.

$$F_N(\Sigma_c) = Tr\left((\Sigma_{c=i}^{-1} - \Sigma_{c=m}^{-1}) \cdot (\Sigma_{c=i}^{-1} - \Sigma_{c=m}^{-1})^T\right), \tag{68}$$

and κ is a positive constant. In our case, $\kappa = 0.005$.

5.3.2. Results

Fig. 24 presents the paths created by the planning session. The path for planning over R_1 focuses on the object group on the kitchen counter, while the others focus more on the object on the table by the window. This can be explained by poorer visibility of the objects near the window, induced by the sunlight, therefore inducing higher epistemic uncertainty than the objects on the counter. As we consider a SLAM setting without prior knowledge of how many objects are in the scene, the robot corresponding to the black trajectory did not observe the objects in the bottom-left cluster. Therefore it did not take into account these objects during inference and planning.

The results of those trajectories chosen can be seen in Fig. 25, where the entropy and MSDE results are presented. In Fig. 25a the lower epistemic uncertainty for planning with R_2 can be evident. In addition, the MSDE comparison in Fig. 25b significantly favors planning over R_1 over R_2 and especially compared to WEU, with epistemic-uncertainty-aware planning outperforms both.

Fig. 26 shows the class probability of the ground truth class for all the objects for time-step $k = 20$. While both JLP-BSP with R_2 and WEU observe an object more as the group near the window contains more objects, the objects that JLP-BSP with R_1 observes are classified more accurately.

Fig. 27 presents a computational time comparison between JLP-BSP and WEU. The figure shows a significant advantage for JLP-BSP over WEU, as the number of candidate classes is 5, instead of 2 in the simulation. WEU computational time per step drops with time steps as some class realizations are pruned. As evident from the figure, JLP-BSP offers computational efficiency greater than WEU, while also opening access to model uncertainty, both for inference and planning.

6. Discussion and limitations

This paper extends the classical formulation of Gaussian passive and active SLAM to include object classification scores that also reason about the classifier's epistemic uncertainty (e.g., via MC-Dropout). The first presented approach, MH, as discussed in Sec. 3.2, is practically intractable without aggressive pruning (i.e., using a high pruning threshold) as the number

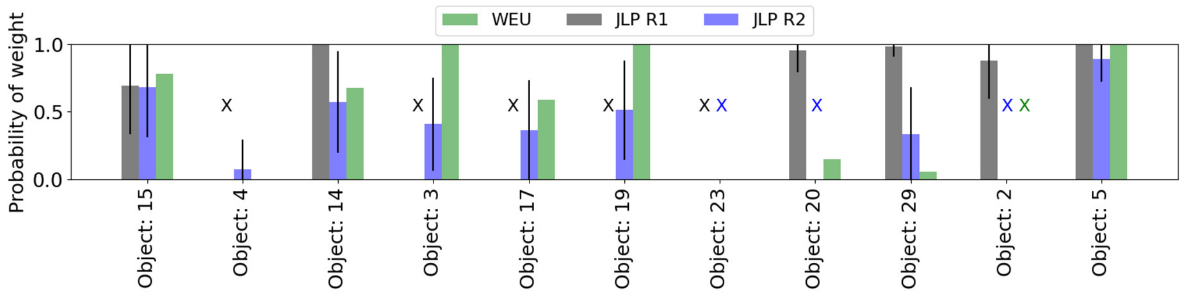


Fig. 26. Probability of the objects being ground truth class for our methods at time $k = 20$ for all objects. We compare planning over R_1 and R_2, JLP-BSP, and WEU. Black and blue for using for R_1 and R_2 respectively using JLP-BSP, and green for WEU. The one σ deviation is represented via the black line at each relevant bar and represents the posterior model uncertainty. The colored X marks represent that object was not observed by the corresponding method.

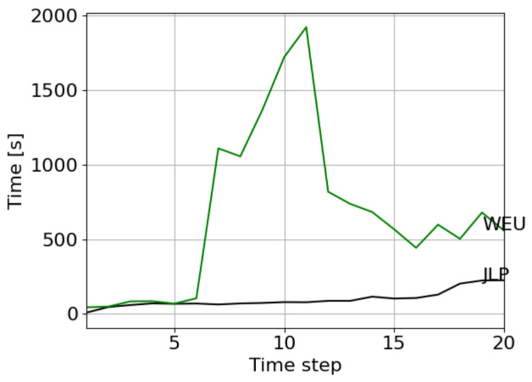


Fig. 27. This figure compares run-time per inference step between realizations of JLP-BSP in black, and WEU in green for the AVD scenario.

of hybrid beliefs grows exponentially with the number of objects. Also, each hybrid belief is computationally intensive, especially for cases where m is large (e.g., 1000 candidate class on ImageNet [38]). Aggressive pruning may induce impossible to recover from incorrect classification results.

On the other hand, JLP maintains a single continuous belief that does not require pruning, making it significantly more computationally feasible than MH. That being said, JLP is constrained by the dimension of each $l\lambda$: each object contributes $m - 1$ variables to be optimized; therefore large m also induces significant computational effort for the optimization, as $(m - 1)^2$ terms are added to the Jacobian per object, which in general are not sparse. Moreover, to the best of the author's knowledge, there are no convergence guarantees for JLP. The computational effort may be reduced by, for example, sparsification of the Jacobian during optimization.

7. Conclusions

We presented a unified semantic SLAM framework for inference and BSP that maintains a joint belief over robot and objects' poses and posterior class probability, addressing viewpoint-based classification aliasing and reasoning about the epistemic uncertainty of the classifier. In particular, two approaches were introduced; firstly, we introduced MH which simultaneously maintains multiple hybrid beliefs over poses and object classes, with semantic class probability vector measurements varying with different predetermined weights. Secondly, we introduced JLP, a more computationally efficient alternative that uses the novel JLP factor. Furthermore, we introduced MH-BSP and JLP-BSP as the formulation of both approaches to a BSP framework. We introduced a novel information-theoretic reward to plan over future posterior epistemic uncertainty, improving classification performance over methods and reward functions that do not consider epistemic uncertainty. Both approaches leverage the coupling between relative poses and object classes via a viewpoint-dependent classifier uncertainty model, allowing us to predict future epistemic uncertainty for planning. In simulation and experiment, we showed that reasoning about epistemic uncertainty improves classification performance in inference and planning.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Appendix A

A.1. Multi-Hybrid: derivation of Eq. (19) and marginals for single object

We marginalize $b[\lambda_k, \mathcal{X}_k]$ over w:

$$b[\lambda_k, \mathcal{X}_k] = \int\limits_w \mathbb{P}(\mathcal{X}_k, \lambda_k | I_{1:k}, \mathcal{H}_k^g, w) \cdot \mathbb{P}(w|D) dw \approx \frac{1}{|W|} \sum_w \mathbb{P}(\mathcal{X}_k, \lambda_k | I_{1:k}, \mathcal{H}_k^g, w). \tag{A.1}$$

Then, using chain rule yields

$$b[\lambda_k, \mathcal{X}_k] \approx \frac{1}{|W|} \sum_w \mathbb{P}(\mathcal{X}_k | \lambda_k, w, I_{1:k}, \mathcal{H}_k^g) \cdot \mathbb{P}(\lambda_k | w, I_{1:k}, \mathcal{H}_k^g). \tag{A.2}$$

Each term on the right-hand side of the above is addressed separately. $\mathbb{P}(\mathcal{X}_k | \lambda_k, w, I_{1:k}, \mathcal{H}_k^g)$ is marginalized over c and using chain-rule can be split into the following distributions:

$$\mathbb{P}(\mathcal{X}_k | \lambda_k, w, I_{1:k}, \mathcal{H}_k^g) = \sum_c \mathbb{P}(\mathcal{X}_k | c, \lambda_k, w, I_{1:k}, \mathcal{H}_k^g) \cdot \mathbb{P}(c | \lambda_k, w, I_{1:k}, \mathcal{H}_k^g) \tag{A.3}$$

$\mathcal{X}_k$ is conditioned on c, thus λ_k can be omitted. For c, given the posterior probability vector λ_k, the rest can be omitted, and $\mathbb{P}(c|\lambda_k) = \lambda_k^c$ where λ_k^c is the element c of λ_k.

λ_k is a function of w, $I_{1:k}$, and $\mathcal{H}_k^g$; therefore, $\mathbb{P}(\lambda_k | w, I_{1:k}, \mathcal{H}_k^g)$ is a Dirac function $\delta(\cdot)$, such that

$$\mathbb{P}(\lambda_k | w, I_{1:k}, \mathcal{H}_k^g) = \delta(\lambda_k - \lambda_{k,w}). \tag{A.4}$$

As such, Eq. (A.2) is rewritten as:

$$b[\lambda_k, \mathcal{X}_k] \approx \frac{1}{|W|} \sum_c \sum_w \underbrace{\mathbb{P}(\mathcal{X}_k | c, l\gamma_{1:k,w}, \mathcal{H}_k^g)}_{b_w^c[\mathcal{X}_k]} \cdot \lambda_k^c \cdot \delta(\lambda_k - \lambda_{k,w}), \tag{A.5}$$

where $b_w^c[\mathcal{X}_k]$ is the continuous belief conditioned on w and c. Each $w \in W$ is constant throughout the scenario with the reasoning of keeping the number of particles constant, thus avoiding managing an exponentially increasing number of components (such as in [8]). This can be achieved by, e.g., training multiple models on the same dataset via bootstrapping, or re-using classifier weight sets created by MC-dropout. That way, Eq. (A.5) shows that maintaining $b[\lambda_k, \mathcal{X}_k]$ is equivalent to maintaining $b_w^c[\mathcal{X}_k]$ and λ_k^c for all $w \in W$ and c.

Each class probability $\lambda_{k,w}^c$ within the particle $\lambda_{k,w}$ is updated using Bayes rule as follows:

$$\lambda_{k,w}^c = \eta \cdot \lambda_{k-1,w}^c \cdot \mathbb{P}(z_k^g, l\gamma_{k,w} | c), \tag{A.6}$$

where η is a normalizing constant such that $\sum_c \lambda_k^c = 1$, and does not affect inference. As the classifier model is viewpoint dependent (Eq. (17)), we must marginalize $\mathbb{P}(z_k^g, l\gamma_{k,w}|c)$ over x^o and x_k to fully utilize our models as follows:

$$\lambda_{k,w}^c \propto \lambda_{k-1,w}^c \int\limits_{x^o, x_k} \mathcal{L}_k \cdot b_w^{c-}[x^o, x_k] dx^o dx_k, \tag{A.7}$$

where $b_w^{c-}[x^o, x_k]$ is the propagated conditional continuous belief, constructed as follows, all the other variables are marginalized out from $\mathcal{X}_k$ beside x^o and x_k:

$$b_w^{c-}[x^o, x_k] \triangleq \int\limits_{\mathcal{X}_k / x^o, x_k} \mathcal{M}_k \cdot b_w^c[\mathcal{X}_{k-1}] d(\mathcal{X}_k / x^o, x_k). \tag{A.8}$$

$b_w^c[\mathcal{X}_k]$ from (A.5) is incrementally updated using standard SLAM state-of-the-art approaches (e.g. iSAM2 [32]):

$$b_w^c[\mathcal{X}_k] \propto b_w^c[\mathcal{X}_{k-1}] \cdot \mathcal{M}_k \cdot \mathcal{L}_k. \tag{A.9}$$

Essentially, for every w, we maintain a hybrid belief over robot and object poses, and classes, defined as:

$$hb_w[\mathcal{X}_x, c] \triangleq b_w^c[\mathcal{X}_k] \cdot \lambda_{k,w}^c, \tag{A.10}$$

and using the above definition, and considering that the Dirac function only "blocks" all λ_k except for $\lambda_{k,w}$, we can rewrite Eq. (A.5) in terms of $hb_w[\mathcal{X}_x, c]$:

$$b[\lambda_k, \mathcal{X}_k] \approx \frac{1}{|W|} \sum_c \sum_w hb_w[\mathcal{X}_k, c] \cdot \delta(\lambda_k - \lambda_{k,w}). \tag{A.11}$$

Practically, for every $w \in W$, MH maintains $b^c_w[\mathcal{X}_k]$ with the accompanying $\lambda^c_{k,w}$ for every object class realization, overall maintaining $|W|$ hybrid beliefs $hb_w[\mathcal{X}_k, c]$ in parallel.

Further, one may require to infer the marginals $b[\lambda_k]$ or $\mathbb{P}(\mathcal{X}_k | I_{1:k}, \mathcal{H}^g_k, D)$, e.g. to compute an appropriate reward function, as we shall see in Section 4. We can describe $b[\lambda_k]$ in term of $\lambda_{k,w}$ particles by marginalizing $b[\lambda_k]$ over w:

$$\begin{aligned} b[\lambda_k] &\approx \frac{1}{|W|} \sum_w \mathbb{P}(\lambda_k | w, I_{1:k}, \mathcal{H}^g_k) \\ &= \frac{1}{|W|} \sum_w \mathbb{P}(\lambda_k | l\gamma_{1:k,w}, \mathcal{H}^g_k) \\ &= \frac{1}{|W|} \sum_w \delta(\lambda_k - \lambda_{k,w}). \end{aligned} \tag{A.12}$$

On the other hand, to compute $\mathbb{P}(\mathcal{X}_k | I_{1:k}, \mathcal{H}^g_k, D)$, we marginalize over w and c,

$$\mathbb{P}(\mathcal{X}_k | I_{1:k}, \mathcal{H}^g_k, D) \approx \frac{1}{|W|} \sum_w \sum_c hb_w[\mathcal{X}_k, c]. \tag{A.13}$$

A.2. Multi-Hybrid: derivation of Eq. (22) and marginals for multiple objects

Define the joint posterior class probability vector as:

$$\Lambda_k \triangleq \mathbb{P}(C | l\Gamma_{1:k}, \mathcal{H}^g_k), \tag{A.14}$$

where $C \triangleq \{c^o\}_{o \in O_{1:k}}$ is the class realization of all objects observed up to time k, with c^o being the o-th object class. In addition, we include in $\mathcal{X}_k$ the poses of all the objects, such that $\mathcal{X}_k \triangleq x_{0:k} \cup \{x^o\}_{o \in O_{1:k}}$. Subsequently, the belief over Λ_k and $\mathcal{X}_k$ is:

$$b[\Lambda_k, \mathcal{X}_k] \triangleq \mathbb{P}(\Lambda_k, \mathcal{X}_k | I_{1:k}, \mathcal{H}^g_k, D). \tag{A.15}$$

Observe that Λ_k is still a probability vector but with m^{N_k} possible categories. To illustrate this, consider an example with two objects and three candidate classes, i.e. $O_{1:k} = \{o, o'\}$ and $m = 3$. Then each category contains a class hypothesis for all object classes, e.g. $c^o = 1$, $c^{o'} = 3$. As such, there are 9 possible class realizations and therefore Λ_k has 9 categories whose probabilities should sum to one. That way, the number of categories in Λ_k grows exponentially with the number of objects, potentially to intractable levels. Fortunately, this can be mitigated by pruning components with low probability, as was done in [5,6].

For every $w \in W$, updating Λ_k is largely similar to updating λ_k in Appendix A.1, except for a few differences. The likelihood terms include all objects O_k observed at time k, and the conditional probability over the poses is conditioned on class realization C instead of the class of a single object,

$$\Lambda^C_{k,w} \propto \Lambda^C_{k-1,w} \int_{x^{inv}_k} \mathcal{L}_k \cdot b^{C-}_w[\mathcal{X}^{inv}_k] dx^{inv}_k, \tag{A.16}$$

where $\Lambda^C_{k,w}$ denotes the posterior probability of class realization C at time k for weight realization w, and $\mathcal{X}^{inv}_k$ represents the last robot pose and all poses of objects observed at time k, i.e. $\mathcal{X}^{inv}_k \triangleq x_k \cup \{x^o\}_{o \in O_k}$. Likelihood $\mathcal{L}_k$ now encompasses all the measurement likelihoods of all objects as follows:

$$\mathcal{L}_k \triangleq \prod_{o \in O_k} \mathbb{P}(l\gamma^o_k | C, x^o, x_k) \cdot \mathbb{P}(z^g_k | x^o, x_k). \tag{A.17}$$

The belief $b^{C-}_w[\mathcal{X}^{inv}_k]$ is the propagated belief conditioned on C, and marginalized over the uninvolved variables such that $\mathcal{X}_k = \mathcal{X}^{inv}_k \cup \mathcal{X}^{\neg inv}_k$:

$$b^{C-}_w[\mathcal{X}^{inv}_k] = \int_{\mathcal{X}^{\neg inv}_k} \mathcal{M}_k \cdot b^C_w[\mathcal{X}_{k-1}] d\mathcal{X}^{\neg inv}_k. \tag{A.18}$$

Similarly to Sec. 3.2.1 we can rewrite $b[\Lambda_k, \mathcal{X}_k]$ as:

$$b[\Lambda_k, \mathcal{X}_k] \approx \frac{1}{|W|} \sum_w \sum_C hb_w[\mathcal{X}_k, C] \cdot \delta(\Lambda_k - \Lambda_{k,w}), \tag{A.19}$$

where $hb_w[\mathcal{X}_k, C]$ is the hybrid belief conditioned on w:

$$hb_w[\mathcal{X}_k, C] \triangleq b_w^C[\mathcal{X}_k] \cdot \Lambda^C_{k,w}, \tag{A.20}$$

and $b_w^C[\mathcal{X}_k] \triangleq \mathbb{P}(\mathcal{X}_k | c, l\gamma_{1:k,w}, \mathcal{H}_k^g)$. Similarly to Eq. (19), maintaining $b[\Lambda_k, \mathcal{X}_k]$ is equivalent to maintaining $hb_w[\mathcal{X}_k, C]$ for all $w \in W$ and C. In case $b[\Lambda_k]$ is required, for the multi-object case Eq. (A.12) becomes:

$$b[\Lambda_k] \approx \frac{1}{|W|} \sum_w \delta(\Lambda_k = \Lambda_{k,w}); \tag{A.21}$$

Similarly. In case $\mathbb{P}(\mathcal{X}_k | I_{1:k}, \mathcal{H}_k^g, D)$ is required, for the multi-object case Eq. (A.13) becomes:

$$\mathbb{P}(\mathcal{X}_k | I_{1:k}, \mathcal{H}_k^g, D) \approx \frac{1}{|W|} \cdot \sum_w \sum_C hb_w[\mathcal{X}_k, C]. \tag{A.22}$$

A.3. Multi-Hybrid: computation complexity analysis

With m candidate classes and n objects, the number of possible class realizations per $\Lambda_{k,w}$ is m^n, as C considers all possible class realizations of all objects observed thus far, making C combinatorial in nature. Inference over continuous states (k camera poses and n object poses), i.e. the conditional belief $\mathbb{P}(\mathcal{X}_k | C, l\gamma_{1:k}, \mathcal{H}_k^g)$, can be efficiently done using, e.g. the state of the art iSAM2 approach [32], with a computational complexity of $O((k+n)^{1.5})$, and at worst $O((k+n)^3)$ for loop closures. To compute an individual λ_k particle, we must account for the attached $\mathbb{P}(\mathcal{X}_k | C, l\gamma_{1:k}, \mathcal{H}_k^g)$, and thus the worst-case computational complexity is $O(m^n(k+n)^3)$. Eventually MH maintains $|W|$ particles, and therefore the overall time computational complexity for $b[\lambda_k]$ inference is $O(|W| m^n (k+n)^3)$ at worst without pruning.

The computational complexity can be further reduced by pruning e.g. low probability classes for individual particles, but as with any pruning, this can induce a problem where a certain realization gets “locked” in either probability 0 or 1, rendering the possibility of probability changing for the said realization impossible.

For memory complexity, we require the latest robot pose, the n poses, and m sized class probability vector for every object per λ particle. All in all, we must maintain $O(nm^n)$ random variables in memory per particle, and $O(|W| nm^n)$ variables for $b[\lambda]$. This also can be reduced by e.g. pruning low probability class realizations or incremental inference methods.

A.4. Proof of Lemma 1

In this proof, the time index k and occasionally x_k^{rel} from h_c and Σ_c are omitted to reduce clutter. We prove by construction by writing the PDF of the classifier uncertainty model for the i-th element of $l\mathcal{L}^s$. The model for class i has an expectation $h_{c=i}(x^{rel})$ and a covariance matrix $\Sigma_{c=i}(x^{rel})$. Thus:

$$\begin{aligned} l\mathcal{L}_i^s &= \log \left(\frac{(2\pi)^{\frac{m-1}{2}} \sqrt{|\Sigma_{c=m}|} e^{-\frac{1}{2}||l\gamma - h_{c=i}||^2_{\Sigma_{c=i}}}}{(2\pi)^{\frac{m-1}{2}} \sqrt{|\Sigma_{c=i}|} e^{-\frac{1}{2}||l\gamma - h_{c=m}||^2_{\Sigma_{c=m}}}} \right) \\ &= -\frac{1}{2}\log(|\Sigma_{c=i}|) + \frac{1}{2}\log(|\Sigma_{c=m}|) \\ &- \frac{1}{2}||l\gamma - h_{c=i}||^2_{\Sigma_{c=i}} + \frac{1}{2}||l\gamma - h_{c=m}||^2_{\Sigma_{c=m}}. \end{aligned} \tag{A.23}$$

Now, applying the condition $\Sigma_{c=i}(x^{rel}) \equiv \Sigma_{c=j}(x^{rel})$, and denoting both as Σ_c we get the following expression:

$$l\mathcal{L}_i^s = l\gamma^T \Sigma_c^{-1} h_{c=i} - \frac{1}{2} h_{c=i}^T \Sigma_c^{-1} h_{c=i} - l\gamma^T \Sigma_c^{-1} h_{c=m} + \frac{1}{2} h_{c=m}^T \Sigma_c^{-1} h_{c=m}. \tag{A.24}$$

From the above equation, if $l\gamma$ is a multi-variate Gaussian random variable, then $l\mathcal{L}_i^s$ is a linear combination of Gaussian random variables, therefore Gaussian by itself. This is valid for every $i \in [1, m-1]$.

In general, the classifier model covariance functions may not be equivalent. Therefore, Eq. (A.23) includes a quadratic expression of $l\gamma$, making $l\mathcal{L}_i^s$ a mixture of Gaussian and Generalized Chi distributions. To counter this, the models' covariances must be “close” to each other to approximately describe $l\mathcal{L}_i^s$ as a Gaussian. If $l\mathcal{L}^s$ is assumed Gaussian via moment matching or other methods, it will only approximate the true distribution of $l\mathcal{L}^s$ with the accuracy dependent on the “distance” between $\Sigma_{c=i}(x^{rel})$ and $\Sigma_{c=j}(x^{rel})$ for all $i, j \in [1, m]$. This distance can be represented by, for example, Frobenius Norm.

Remark: The Frobenius norm can be inserted into the loss function while training the viewpoint-dependent classifier uncertainty models (17), thereby enforcing sufficiently close covariance functions between different models such that the approach presented in this section can be used. Our implementation utilizes this concept, as explained in Sec. 5.3.1.

A.5. Joint Lambda Pose computational complexity analysis and discussion

MH maintains Λ_k, a posterior joint probability vector for all class realizations, with m^{n_k} categories as seen in Sec. 3.2.2. Thus Λ_k grows exponentially with the number of objects observed, and considering that the inference is done for $|W|$ times, each w with its own Λ_k, MH is intractable unless pruning methods are applied. The overall time computational complexity for $b[\lambda_k]$ inference is $O(|W|m^n(k+n)^3)$ at worst without pruning, see Appendix A.3. for detailed analysis.

On the other hand, JLP maintains $\bar{l\lambda}_k$ which essentially maintains a separate $l\lambda_k$ per object, resulting in a size of $(m-1)\cdot n_k$, which grows linearly with the number of objects, and results in better scaling. Moreover, the inference is done only once, as the set W only plays a role in classifier output γ.

Consider again, for example, the scenario with two objects and three candidate classes, Λ_k is a probability vector with 9 categories. On the other hand, $\bar{l\lambda}_k = \{l\lambda_k^o, l\lambda_k^{o'}\}$ where $l\lambda_k^o$ and $l\lambda_k^{o'}$, each, is a vector with two elements as they are the logit transformation of λ_k^o and $\lambda_k^{o'}$, respectively, totaling in 4 elements for $\bar{l\lambda}_k$. JLP has a single continuous belief with at most k pose states, and n object pose states. In addition, each object has $l\lambda$ with $m-1$ variables. Consider pose states with d variables each; at worst the total number of variables in JLP is $dk+dn+n\cdot(m-1)$. In total, the computational time complexity is $O\left((dk+dn+nm)^3\right)$ at worst. Compared to MH, JLP scales significantly better with the number of objects.

We can compare Fig. 5 and Fig. 6b to illustrate the difference between MH and JLP inference. Those figures present a scenario with two objects and two candidate classes. Fig. 5 presents the factor graphs for a specific w. Therefore in this case we have to maintain $4|W|$ factor graphs. On the other hand, with JLP we have to maintain a single one that also contains additional $l\lambda$ nodes. As we can see, the advantage of the JLP approach compared to the MH approach is that it does not require maintaining multiple hybrid beliefs. It maintains a single continuous belief that encompasses all poses and object classes while reasoning about classifier epistemic uncertainty, allowing a rich, viewpoint-dependent representation of object class probabilities.

The main disadvantage of this approach is the requirement of Lemma 1 to hold for accuracy. While the requirement can be offset by enforcing additional constraints on the training of classifier uncertainty models, using the resulting models will render JLP as an approximation compared to MH. Another potential drawback is that JLP forces λ to be LG distributed, which results in slower entropy computation (see Appendix A.9 for details). Despite that, the advantage in computational efficiency of JLP is significant enough to offset slower entropy computation relative to MH, thus practically significantly more feasible.

A.6. Proof of Lemma 2

The reverse logit transformation from $l\lambda$ to λ is given by:

$$\lambda = \left[\frac{e^{l\lambda^1}}{1+\sum_{i=1}^{m-1} e^{l\lambda^i}}, \ldots, \frac{e^{l\lambda^{m-1}}}{1+\sum_{i=1}^{m-1} e^{l\lambda^i}}, \frac{e^1}{1+\sum_{i=1}^{m-1} e^{l\lambda^i}}\right]^T. \tag{A.25}$$

Thus, λ is LG distributed, and the probability density function is given as:

$$\mathbb{P}(\lambda) = \frac{1}{\sqrt{2\pi|\Sigma|}} \cdot \frac{1}{\prod_{i=1}^m \lambda^i} \cdot e^{-\frac{1}{2}||l\lambda-\mu||_\Sigma^2}, \tag{A.26}$$

with $\mu \in \mathbb{R}^{m-1}$ and $\Sigma \in \mathbb{R}^{(m-1)\times(m-1)}$ being the LG parameters. The term $\frac{1}{\prod_{i=1}^m \lambda^i}$ is the determinant of the transformation Jacobian and is denoted as $|J(\lambda)|$. Thus we write $\mathbb{P}(\lambda)$ as:

$$\mathbb{P}(\lambda) = \mathbb{P}(l\lambda)|J(\lambda)|, \tag{A.27}$$

with $\mathbb{P}(l\lambda) \triangleq \mathbb{P}(l\lambda) = \mathbb{P}(l\lambda)(\mu, \Sigma)$, and write $H(\lambda)$ as:

$$H(\lambda) = -\int_\lambda \mathbb{P}(\lambda)\cdot \log \mathbb{P}(\lambda) d\lambda \tag{A.28}$$

Then, we transform the integral variable back to $l\lambda$, as we have a closed form expression for $H(l\lambda)$. As $|J(\lambda)|$ is the transformation Jacobian, $|J(\lambda)|d\lambda = dl\lambda$. From there we can write the integral in Eq. (A.28) as a function of $l\lambda$:

$$\begin{aligned} H(\lambda) =& -\int_{\lambda} \mathbb{P}(l\lambda)\cdot|J(\lambda)|\cdot\log(\mathbb{P}(l\lambda)\cdot|J(\lambda)|)d\lambda = \\ & -\int_{l\lambda} \mathbb{P}(l\lambda)\cdot\log(\mathbb{P}(l\lambda)\cdot|J(\lambda)|)dy = \\ & -\int_{l\lambda} \mathbb{P}(l\lambda)\cdot\log(\mathbb{P}(l\lambda))dl\lambda - \int_{l\lambda} \mathbb{P}(l\lambda)\cdot\log(|J(\lambda)|)dl\lambda = \\ & H(l\lambda) - \int_{l\lambda} \mathbb{P}(l\lambda)\cdot\log(|J(\lambda)|)dl\lambda. \end{aligned} \tag{A.29}$$

The term $\int_{l\lambda}\mathbb{P}(l\lambda)\cdot\log(|J(\lambda)|)dl\lambda$ is positive, as $\mathbb{P}(l\lambda)$ is always positive and $J(\lambda) > 1$, therefore $H(\lambda) < H(l\lambda)$. Next, we describe $\log(|J(\lambda)|)$ in as a function of $l\lambda$:

$$\begin{aligned} \log|J(\lambda)| =& \log\left(\frac{1}{\prod_{i=1}^{m}\lambda^i}\right) = -\sum_{i=1}^{m-1}\log\lambda^i - \log\lambda^m = \\ & -\sum_{i=1}^{m-1}\left[\log(e^{l\lambda^i}) + \log\left(1+\sum_{j=1}^{m-1}e^{l\lambda^j}\right)\right] \\ & -\log(1) + \log\left(1+\sum_{j=1}^{m-1}e^{l\lambda^j}\right) = \\ & -\sum_{i=1}^{m-1}l\lambda^i + m\cdot\log\left(1+\sum_{j=1}^{m-1}e^{l\lambda^j}\right). \end{aligned} \tag{A.30}$$

Now we plug the above expression for $\log|J(\lambda)|$ into Eq. (A.29) and express $H(\lambda)$ as a function of $l\lambda$:

$$\begin{aligned} H(\lambda) =& H(l\lambda) \\ & + \int_{l\lambda}\mathbb{P}(l\lambda)\cdot\left[\sum_{i=1}^{m-1}l\lambda^i - m\cdot\log\left(1+\sum_{j=1}^{m-1}e^{l\lambda^j}\right)\right]dl\lambda. \end{aligned} \tag{A.31}$$

As $\int_{l\lambda}\mathbb{P}(l\lambda)l\lambda^i dl\lambda = \mathbb{E}[l\lambda^i]$, we can simplify the above equation into the form shown in Lemma 2:

$$H(\lambda) = H(l\lambda) + \sum_{i=1}^{m-1}\mathbb{E}(l\lambda^i) - m\int_{l\lambda}\log\left(1+\sum_{j=1}^{m-1}e^{l\lambda^j}\right)\cdot\mathbb{P}(l\lambda)dl\lambda. \tag{A.32}$$

A.7. Proof of Lemma 3

A.7.1. Upper bound

Let us look at the integral in Eq. (A.32). The term $\log\left(1+\sum_{j=1}^{m-1}e^{l\lambda^j}\right)$ can be bounded from below by:

$$\log\left(1+\sum_{j=1}^{m-1}e^{l\lambda^j}\right) \geq \log(\max_i\{1, e^{l\lambda^i}\}) = \max_i\{0, l\lambda^i\}. \tag{A.33}$$

Substituting the above equation to Eq. (A.32) yields the following inequality:

$$H(\lambda) \leq H(l\lambda) + \sum_{i=1}^{m-1}\mathbb{E}(l\lambda^i) - m\int_{l\lambda}\max_i\{0, l\lambda^i\}\cdot\mathbb{P}(l\lambda)dl\lambda. \tag{A.34}$$

The integral term is similar to the expectation definition for $l\lambda_i$, except that it considers only positive $l\lambda_i$, making the resulting value from the integral larger than $\mathbb{E}(l\lambda_i)$. For the next step, we consider the case where there is at least a single $\mathbb{E}[l\lambda^i] \geq 0$, and the case where for all i, $\mathbb{E}[l\lambda^i] < 0$. Considering both cases we can write:

$$\begin{cases} \int_{l\lambda} \max_i\{0, l\lambda^i\} \cdot \mathbb{P}(l\lambda) dl\lambda \geq 0 & \mathbb{E}(l\lambda^i) < 0 : \forall i \\ \int_{l\lambda} \max_i\{0, l\lambda^i\} \cdot \mathbb{P}(l\lambda) dl\lambda \geq \max \mathbb{E}(l\lambda^i) & \exists \mathbb{E}(l\lambda^i) \geq 0. \end{cases}$$

Considering both cases:

$$\int_{l\lambda} \max_i\{0, l\lambda^i\} \cdot \mathbb{P}(l\lambda) dl\lambda \geq \max\{0, \mathbb{E} l\lambda^i\}. \tag{A.35}$$

Finally, we can substitute the above expression into Eq. (A.34) and get the expression in Lemma 3:

$$H(\lambda) \leq H(l\lambda) + \sum_{i=1}^{m-1} \mathbb{E}(l\lambda^i) - m \max\{0, \mathbb{E}(l\lambda^i)\}. \tag{A.36}$$

A.7.2. Lower bound

Let us look again at the integral in Eq. (A.32). This time, the term $\log\left(1 + \sum_{j=1}^{m-1} e^{l\lambda^j}\right)$ can be bounded from above by:

$$\log\left(1 + \sum_{j=1}^{m-1} e^{l\lambda^j}\right) \leq \log(\max_i\{m, me^{l\lambda^i}\}) = \max_i\{0, l\lambda^i\} + \log(m). \tag{A.37}$$

Now, we substitute the above inequality into Eq. (A.32), and we get the following expression:

$$H(\lambda) \leq H(l\lambda) + \sum_{i=1}^{m-1} \mathbb{E}(l\lambda^i) - m\log(m) - m \int_{l\lambda} \max_i\{0, l\lambda^i\} \cdot \mathbb{P}(l\lambda) dl\lambda. \tag{A.38}$$

This time we look for an upper bound for $\int_{l\lambda} \max_i\{0, l\lambda^i\} \cdot \mathbb{P}(l\lambda) dl\lambda$. Let us consider that:

$$\int_0^\infty \frac{l\lambda^i}{\sqrt{2\pi}} e^{-\frac{(l\lambda^i)^2}{2\Sigma_{ii}}} dl\lambda^i = \sqrt{\frac{\Sigma_{ii}}{2\pi}} \leq \sqrt{\frac{\Sigma_{ii}^{max}}{2\pi}} \tag{A.39}$$

where Σ_{ii} is the element (i, i) in the diagonal of matrix Σ, and Σ_{ii}^{max} is the largest element of Σ. Then we can bound $\int_{l\lambda} \max_i\{0, l\lambda^i\} \cdot \mathbb{P}(l\lambda) dl\lambda$ by:

$$\begin{cases} \int_{l\lambda} \max_i\{0, l\lambda^i\} \cdot \mathbb{P}(l\lambda) dl\lambda \leq \sqrt{\frac{\Sigma_{ii}^{max}}{2\pi}} & \mathbb{E}(l\lambda^i) < 0 : \forall i \\ \int_{l\lambda} \max_i\{0, l\lambda^i\} \cdot \mathbb{P}(l\lambda) dl\lambda \leq & \\ \leq \max \mathbb{E}(l\lambda^i) + \sqrt{\frac{\Sigma_{ii}^{max}}{2\pi}} & \exists \mathbb{E}(l\lambda^i) \geq 0. \end{cases}$$

From the above equation, we reach:

$$\int_{l\lambda} \max_i\{0, l\lambda^i\} \cdot \mathbb{P}(l\lambda) dl\lambda \leq \max_i\{0, \mathbb{E}(l\lambda^i)\} + \sqrt{\frac{\Sigma_{ii}^{max}}{2\pi}}, \tag{A.40}$$

and by substituting into Eq. (A.32), we reach the lower bound presented in Lemma 3:

$$\begin{aligned} H(\lambda) \geq & H(l\lambda) + \sum_{i=1}^{m-1} \mathbb{E}(l\gamma^i) \\ & - m \cdot \max_i\{0, \mathbb{E}(l\gamma^i)\} - m\log m - \sqrt{\frac{\sigma_{ii}^{max}}{2\pi}}. \end{aligned} \tag{A.41}$$

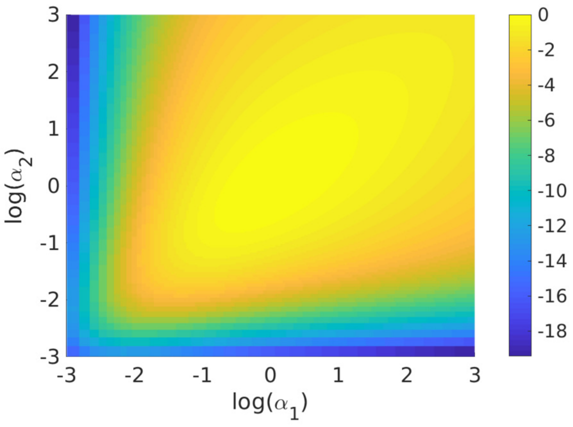


Fig. A.28. Entropy of a two-dimensional Dirichlet distribution as a function of the log of parameters. Blue to yellow colors correspond to low to high entropy values.

A.8. Dirichlet distribution for $b[\lambda]$

An alternative to LG for $b[\lambda]$ (Sec. 4.5) is the Dirichlet distribution, which is supported only by MH, as illustrated in Fig. 7. This distribution is a natural representation of distribution over probability vectors in which samples necessarily satisfy all the conditions of probability vectors presented in Sec. 2.2.

The Dirichlet distribution is parametrized by a parameter set $\alpha \triangleq \{\alpha_1, ..., \alpha_m\}$, and the PDF is:

$$Dir(\lambda; \alpha) = \frac{1}{B(\alpha)} \prod_{i=1}^{m} (\lambda^i)^{\alpha_i - 1}, \tag{A.42}$$

with λ^i being the i-th class probability. $B(\alpha)$ is a normalization constant defined as

$$B(\alpha) \triangleq \frac{\prod_{i=1}^{m} \Gamma(\alpha_i)}{\Gamma(\alpha_0)}, \tag{A.43}$$

where $\Gamma(\cdot)$ is the Gamma function and $\alpha_0 \triangleq \sum_{i=1}^{m} \alpha_i$ for shorthand.

Recall that in MH $b[\lambda]$ is maintained via maintaining each $\lambda_w \in \{\lambda_w\}_{w \in W}$ as in Eq. (A.6) and Eq. (A.16). Dirichlet's distribution parameters, given $\{\lambda\}$, can be estimated in an iterative manner as follows [39]:

$$\psi(\alpha_i^{\text{new}}) = \psi(\sum_{j=1}^{m} \alpha_j^{old}) + \log \hat{\lambda}^i, \tag{A.44}$$

where $\log \hat{\lambda}^i \triangleq \frac{1}{|W|} \log \lambda_w^i$, and $\psi(\cdot)$ is the digamma function. The following expression shows the entropy of the Dirichlet distribution given α parameters:

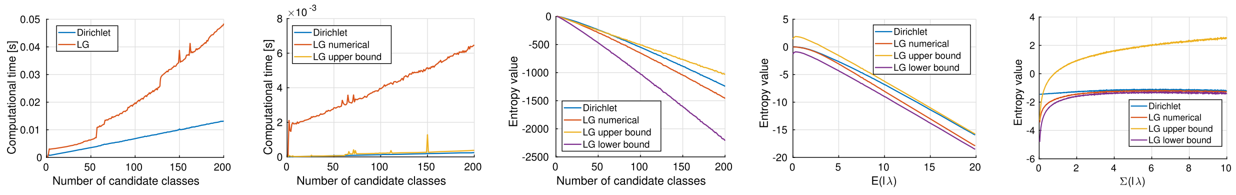
$$H(\lambda) = \log B(a) + (\alpha_0 - m)\psi(\alpha_0) - \sum_{i=1}^{m} (\alpha_i - 1)\psi(\alpha_i). \tag{A.45}$$

The term $B(\alpha)$ needs to be numerically computed. While it is not an analytical solution, the computation is significantly faster than computing differential entropy using samples.

This entropy takes the maximal value when $\alpha_i = 1$, $\forall i$, and at the "edges" of the distribution, where a single parameter is much larger than the others, the entropy is the lowest. If one of the parameters is zero, then $H(\lambda) = -\infty$, as $\psi(0) = -\infty$. This behavior of entropy can be observed in Fig. A.28 which shows an example of a two-dimensional distribution.

The scenarios presented in Fig. 2 correspond to the following cases in Fig. A.28:

- The unknown-unknown case (Fig. 2a) corresponds to α_1 and α_2 that are close to 1, i.e. the central part of Fig. A.28.
- The known-unknown case (Fig. 2b) corresponds to α's with large and similar values, i.e. the upper right part of Fig. A.28.
- The known-known case (Fig. 2c) corresponds to the case where one α is significantly larger than the other and larger than 1, i.e. left or bottom areas of Fig. A.28.
- The uncertain classification case (Fig. 2d) corresponds to the case where one α is not significantly larger than the other, i.e. the areas between the high and low entropy in Fig. A.28.



(a) Inference time (b) Entropy time (c) Entropy value (d) Entropy value (e) Entropy value

Fig. A.29. This figure presents a comparison between Dirichlet and Logistical Gaussian (denoted LG) in terms of computational time, entropy, and log-likelihood values. **(a)** presents a computational time comparison between Dirichlet and LG for inference as a function of probability vector dimension. Similarly, **(b)** presents a computational time for entropy computation, both for numerical and upper bound. **(c)** presents a value comparison between the different entropy computations. Note that in **(b)** and **(c)**, the plots are given the parameters calculated for **(a)**. **(d)** presents an entropy value comparison between distributions with different expectations with a fixed covariance. Similarly, **(e)** presents an entropy value comparison between distributions with different covariance values with a fixed expectation.

A.9. Comparison between Dirichlet and Logistic Gaussian

When considering the reward $H(\lambda)$ we have to consider two steps:

1. Computation of $b[\lambda]$ parameters. With MH we maintain separately $\lambda_w \in \{\lambda_w\}_{w\in W}$ and subsequently describe $b[\lambda]$ using $\{\{\lambda_w\}_{w\in W}\}$. To compute $H(\lambda)$, we must assume a distribution for $b[\lambda]$ and infer its parameters. JLP on the other hand limits $b[\lambda]$ to be LG distributed.
2. Calculation of $H(\lambda)$ to use as a reward function for planning, either via numerical computation or by using bounds.

This discussion is relevant for $H(\lambda)$ computation for MH, as in JLP λ is LG distributed by definition.

Parameter inference for Dirichlet is faster than for LG although the process is numeric. On the other hand, LG is more expressive. For m classes, Dirichlet distribution has m parameters, while LG has $m-1$ parameters for $\mathbb{E}(l\lambda)$, and $\frac{m\cdot(m-1)}{2}$ parameters for $\Sigma(l\lambda)$, totaling in $(1+\frac{m}{2})\cdot(m-1)$ parameters.

LG does not have an analytical solution for computing entropy, and its numeric computation is slower than Dirichlet's. On the other hand, computing bounds of entropy for LG is comparable in terms of computational effort to computing Dirichlet entropy. One must note that while Dirichlet is less computationally expensive than LG when MH with Dirichlet and JLP are compared, JLP is still computationally much more efficient.

Fig. A.29 compares the two distributions in terms of computational effort and value of the entropy. In all figures, the x-axis is the number of candidate classes, and for each, a dataset of 1000 class probability vectors was sampled. Fig. A.29a presents the measured time of parameter computation, clearly showing an advantage for Dirichlet distribution for high dimensional probability vectors despite the parameter computation process for Dirichlet distribution containing functions that must be numerically computed. Fig. A.29b presents the computational time of the entropy, for numerical computation for both LG and Dirichlet, and the bounds for LG (computation time is identical both for upper and lower bounds; thus only the upper bound is shown). Here entropy computation for Dirichlet holds a significant advantage over the numerical computation of entropy for LG, and the bound computation time is comparable to Dirichlet. Fig. A.29c presents entropy values for Dirichlet, numerical LG, lower and upper bounds for LG. The upper bound tends to be close to the numerical solution for fewer candidate classes. In addition, entropy for Dirichlet distribution tends to be higher. In Fig. A.29d the number of candidate classes is fixed to two, i.e. the dimension of $l\lambda$ is $\mathbb{R}^1$. The covariance $\Sigma(l\lambda)$ is fixed at 3, and $\mathbb{E}(l\lambda)$ goes from 0 to 20. In this figure, the entropy value monotonically decreases when increasing $\mathbb{E}(l\lambda)$. The lower bound is tighter between the two bounds as $\mathbb{E}(l\lambda)$ increases. In Fig. A.29e $\mathbb{E}(l\lambda)$ is fixed instead at $\mathbb{E}(l\lambda)=3$ and $\Sigma(l\lambda)$ varies between 0 and 10. We can see that the entropy value increases with the increase in $\Sigma(l\lambda)$, but the bigger effect is for LG compared to the Dirichlet distribution. The upper bound is tighter at lower $\mathbb{E}(l\lambda)$ values, while the lower bound is tighter for higher values.

One may ask: which distribution should be used? As mentioned previously, JLP is limited to LG. For MH, the trade-off is between distribution expressiveness and computational effort. While Logistical Gaussian is more expressive because of a larger number of parameters, the computation effort is significantly higher than for Dirichlet distribution. Also, Dirichlet distribution, unlike the Logistical Gaussian, can manage a very small number of probability vector samples.

A.10. Inference: Joint Lambda Pose assumption

One may consider the ramifications of using JLP with models that do not satisfy Lemma 1. The most straightforward result is that MH and JLP results do not coincide with each other, and that may result in either erroneous or overconfident classification (i.e. large values of $\mathbb{E}(l\lambda)$ and/or too small values of $\Sigma(l\lambda)$).

This time, the classifier model uses the following parameters: for expectation:

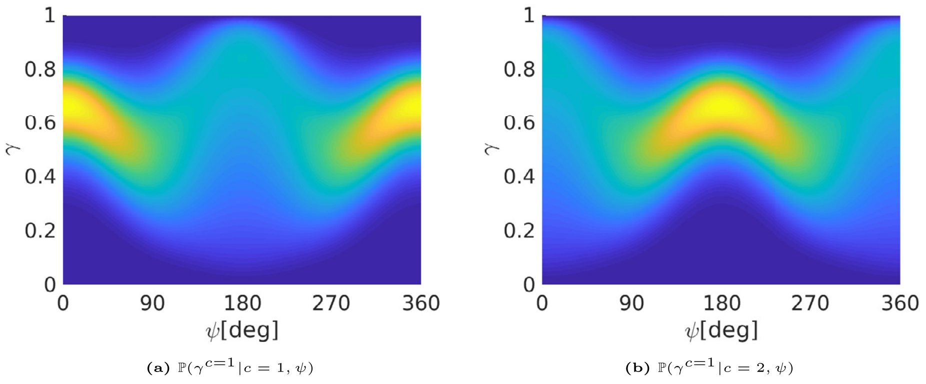


Fig. A.30. A visualization of the classifier uncertainty model used in Appendix A.10. We present the value of $\mathbb{P}(\gamma^{c=1}|c,\psi)$ as a function of relative orientation ψ and $\gamma^{c=1}$ value, for classes $c=1$ and $c=2$ in **(a)** and **(b)** respectively. Blue and yellow colors correspond to low and high PDF values respectively.

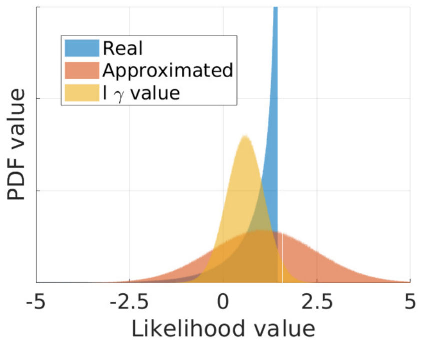


Fig. A.31. An approximate PDF value graph for JLP assumption where $\psi=0°$. The yellow area represents the distribution of $l\gamma_k$, the red area represents $\mathcal{L}_k^s$ PDF when JLP assumption is used, and the blue area represents the real PDF of $\mathcal{L}_k^s$.

$$\begin{aligned} h_{c=1}(x^{rel}) &= 0.3\cdot\cos(2\cdot\psi)+0.3 \\ h_{c=2}(x^{rel}) &= -0.3\cdot\cos(2\cdot\psi)-0.3, \end{aligned} \tag{A.46}$$

and the following parameters for root-information:

$$\begin{aligned} R_{c=1}(x^{rel}) &= 1.4+0.6\cdot\cos(\psi) \\ R_{c=2}(x^{rel}) &= 1.4-0.6\cdot\cos(\psi). \end{aligned} \tag{A.47}$$

The goal is creating opposing $\Sigma_c(\psi)$ for the two classes, such that Lemma 1 does not hold and may result in inaccurate classification while using JLP. Fig. A.30 presents a visualization of the model. Specifically, the problematic areas are around $\psi=0°$ where the models actually predict that when $\gamma^{c=1}>0.8$ the likelihood is actually higher for $c=2$, and vice-versa when $\psi=180°$ when $\gamma^{c=2}>0.8$ predicts a higher likelihood for $c=1$.

Fig. A.31 presents the PDF values of $l\gamma_k$, and $\mathcal{L}_k^s$ with and without JLP assumption at $\psi=0°$. Here $l\gamma_k\sim\mathcal{N}(0.6,0.25)$, coinciding with the classifier uncertainty model parameters of class $c=1$. In this figure, the difference between the approximation and real $\mathcal{L}_k^s$ is evident, with the approximated $\mathcal{L}_k^s$ risking a larger chance of incorrect classification as the area below 0 is larger than that for the real $\mathcal{L}_k^s$.

In this scenario, we use the same setting we used at Sec. 5.2.1, 5.2.2, and 5.2.3, and compare the MSDE scores in Fig. A.32. For Fig. A.32a and A.32b we use the scenario from Sec. 5.2.2. In Fig. A.32a MSDE results for MH with 100 hybrid beliefs are shown as the most accurate, where objects 1 and 2 have more accurate classification than the rest. Fig. A.32b presents MSDE results for JLP, where we see significantly less accurate results. Finally, we perform a statistical study with 5 random object locations of 10 runs as in Sec. 5.2.3, comparing between MH with 10 hybrid beliefs, JLP, and WEO, and see that statistically the difference between MH and JLP is not large even without Lemma 1 holding, as opposed to the specific run from Figs. A.32a and A.32b.

A.10.1. Planning: single object

We simulate a planning scenario of a single object using EUS-BSP. Relative to the object, there is an area with low epistemic uncertainty and high separation between classes, represented as a blue cone in Fig. A.33a. We use the same settings as in Sec. 5.2.4, with the same reward functions R_1 and R_2. The trajectory length is 20 time steps, at each time

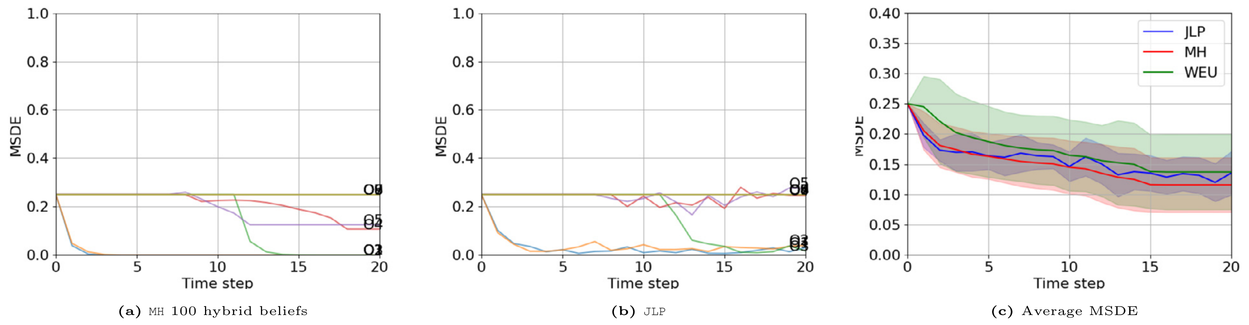


Fig. A.32. **(a)** shows MSDE results to time step per object for MH with 100 hybrid beliefs which we consider most accurate, while **(b)** shows JLP results for MSDE. **(c)** is a statistical study comparing average MSDE over all objects for WEU in green, JLP in blue, and MH in red. The line corresponds to MSDE expectation and the colored area to one σ range.

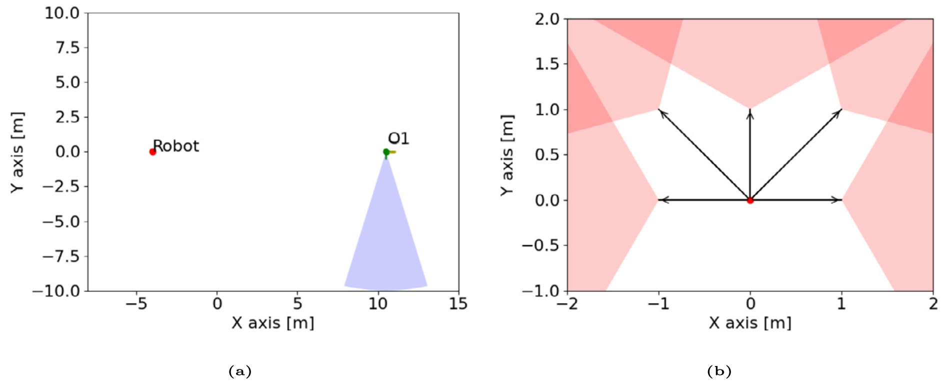


Fig. A.33. **(a)** is the ground truth of the scenario in Sec. 5.2.4. The red dot represents the robot's starting point. The green dot represents the object's location with its label. The green line represents the object orientation, with the yellow line presenting 90° of that orientation. The blue cones represent the observation angles in which the objects are classified most accurately with the lowest epistemic uncertainty. **(b)** presents the five motion primitives in the scenario. The red dot represents the origin point, the black arrows the possible actions, and the red cones represent the field of view after the action.

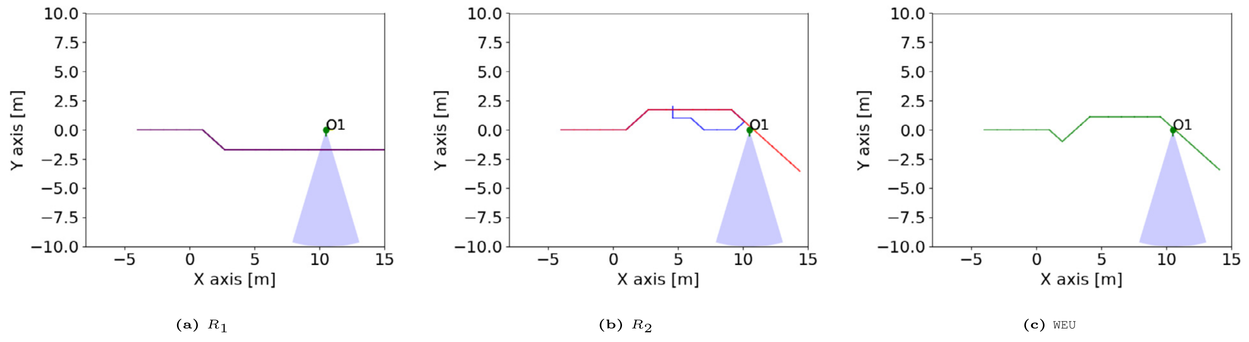


Fig. A.34. This figure presents the ground truth of planned trajectories. **(a)** for planning over R_1 for MH-BSP (purple) and JLP-BSP (black). **(b)** for planning over R_2 for MH-BSP (red) and JLP-BSP (blue). **(c)** for WEU (green). All are for the multiple object scenario. The object is shown in a green dot, with the green line representing the object orientation, with the yellow line presenting 90° of that orientation. The blue cones represent the areas where observations have the lowest epistemic uncertainty.

step, trajectories are randomly sampled for a horizon $L = 10$. This section presents results for a single object. We analyze the differences in performance and the trajectories over R_1 and R_2 for MH-BSP and JLP-BSP, and compare them to the baseline WEU.

Fig. A.34 presents the ground truth trajectories calculated by planning over R_1 and R_2 both for JLP-BSP and MH-BSP, and planning for R_2 for WEU. The epistemic-uncertainty-aware methods seek to pass near the blue-cone area for more accurate classification with lower epistemic uncertainty. Methods that plan over R_1 tend to pass through the cone.

The behavior presented in Fig. A.34 is reflected in Fig. A.35 where the values of $H(\lambda_k)$ are shown as a function of time during inference after the corresponding action has been performed. The values of $H(\lambda_k)$ correlate to the epistemic

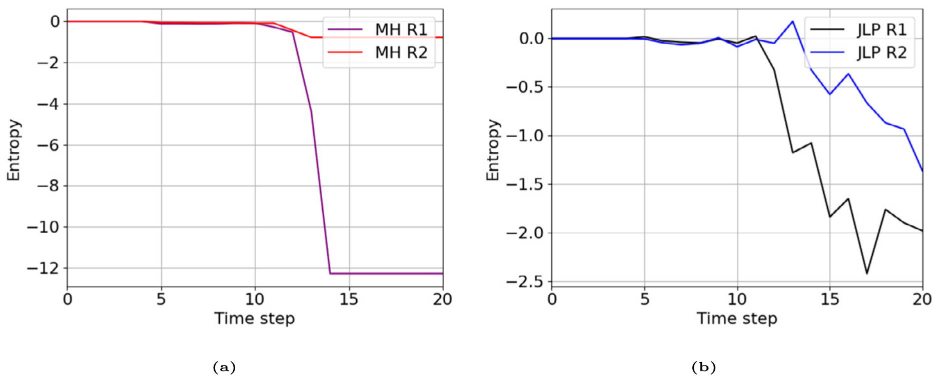


Fig. A.35. $H(\lambda_{20})$ values for MH **(a)** and JLP **(b)** as a function of the time step. In **(a)**, the purple and red plots represent R_1 and R_2 respectively, and similarly in **(b)**, the black and blue plots represent R_1 and R_2 respectively.

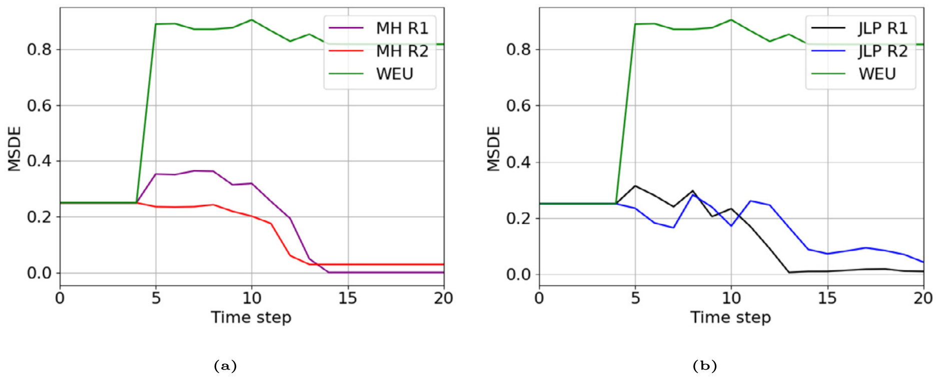


Fig. A.36. MSDE values as a function of time step for MH **(a)** and JLP **(b)**, compared to WEU. In **(a)**, the purple and red plots represent R_1 and R_2 respectively, and similarly in **(b)**, the black and blue plots represent R_1 and R_2 respectively. WEU is represented in both figure with a green plot.

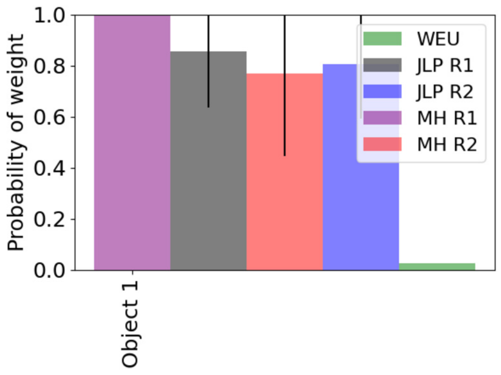


Fig. A.37. Probability of the object being class $c = 2$ (ground truth) for our methods at time $k = 20$. We compare planning over R_1 and R_2, JLP-BSP, MH-BSP, and WEU. Purple and red for R_1 and R_2 respectively using MH-BSP, black and blue for using for R_1 and R_2 respectively using JLP-BSP, and green for WEU. The one σ deviation is represented via the black line at each relevant bar and represents the posterior model uncertainty.

uncertainty. Evidently, planning over R_1 yields lower epistemic uncertainty for both MH-BSP (Fig. A.35a) and JLP-BSP (Fig. A.35b).

Fig. A.36 presents MSDE results for all the methods, split into results for MH in Fig. A.36a and for JLP in Fig. A.36b, both showing a comparison to WEU in the green plot. WEU performs significantly worse in this setting than all the other methods. When comparing planning over R_1 and R_2, the first presents better results for both JLP and MH.

Fig. A.37 presents the results at time $k = 20$ for all methods as a bar graph with error margins for the ground truth class. We can compare the entropy from Fig. A.34 and MSDE from Fig. A.36 with the bar graphs, with lower entropy values resulting in smaller posterior epistemic uncertainty. Similarly, lower MSDE values result in a more “certain” result in the bar graph, as we can see for methods that plan over R_1.

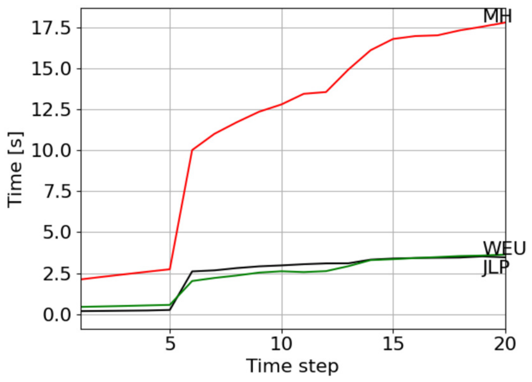


Fig. A.38. This figure compares run-time per inference step between realizations of MH with 5 hybrid beliefs in red, JLP in black, and WEU in green.

In Fig. A.38 we perform computation time comparisons between WEU, MH-BSP, and JLP-BSP. The significant advantage in computational time for JLP-BSP is evident against MH-BSP, and while WEU is lower still, JLP-BSP also opens the possibility of reasoning about epistemic uncertainty.

References

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I.D. Reid, J.J. Leonard, Simultaneous localization and mapping: present, future, and the robust-perception age, IEEE Trans. Robot. 32 (6) (2016) 1309–1332.
[2] Y. Gal, Z. Ghahramani, Dropout as a Bayesian approximation: representing model uncertainty in deep learning, in: Intl. Conf. on Machine Learning (ICML), 2016.
[3] G. Paass, Assessing and improving neural network predictions by the bootstrap algorithm, in: Advances in Neural Information Processing Systems (NIPS), 1993, pp. 196–203.
[4] D. Kopitkov, V. Indelman, Robot localization through information recovered from cnn classificators, in: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), IEEE, 2018.
[5] V. Tchuiev, Y. Feldman, V. Indelman, Data association aware semantic mapping and localization via a viewpoint-dependent classifier model, in: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), 2019.
[6] V. Tchuiev, V. Indelman, Semantic distributed multi-robot classification, localization, and mapping with a viewpoint dependent classifier model, IEEE Robot. Autom. Lett. 5 (3) (2020) 4649–4656.
[7] Y. Feldman, V. Indelman, Spatially-dependent Bayesian semantic perception under model and localization uncertainty, Auton. Robots (2020).
[8] V. Tchuiev, V. Indelman, Inference over distribution of posterior class probabilities for reliable Bayesian classification and object-level perception, IEEE Robot. Autom. Lett. 3 (4) (2018) 4329–4336.
[9] L.P. Kaelbling, M.L. Littman, A.R. Cassandra, Planning and acting in partially observable stochastic domains, Artif. Intell. 101 (1) (1998) 99–134.
[10] V. Indelman, L. Carlone, F. Dellaert, Planning in the continuous domain: a generalized belief space approach for autonomous navigation in unknown environments, Int. J. Robot. Res. 34 (7) (2015) 849–882.
[11] E.I. Farhi, V. Indelman, iX-BSP: belief space planning through incremental expectation, in: IEEE Intl. Conf. on Robotics and Automation (ICRA), 2019.
[12] D. Ha, J. Schmidhuber, World models, arXiv preprint, arXiv:1803.10122, 2018.
[13] A. Coates, A.Y. Ng, Multi-camera object detection for robotics, in: 2010 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2010, pp. 412–419.
[14] S. Omidshafiei, B.T. Lopez, J.P. How, J. Vian, Hierarchical bayesian noise inference for robust real-time probabilistic object classification, arXiv preprint, arXiv:1605.01042, 2016.
[15] J. Nitsch, M. Itkina, R. Senanayake, J. Nieto, M. Schmidt, R. Siegwart, M. Kochenderfer, C. Cadena, Out-of-distribution detection for automotive perception, arXiv preprint, arXiv:2011.01413, 2020.
[16] A. Malinin, M. Gales, Predictive uncertainty estimation via prior networks, in: Advances in Neural Information Processing Systems (NIPS), 2018, pp. 7047–7058.
[17] J. Velez, G. Hemann, A.S. Huang, I. Posner, N. Roy, Modelling observation correlations for active exploration and robust object detection, J. Artif. Intell. Res. (2012).
[18] W. Teacy, S.J. Julier, R. De Nardi, A. Rogers, N.R. Jennings, Observation modelling for vision-based target search by unmanned aerial vehicles, in: Intl. Conf. on Autonomous Agents and Multiagent Systems (AAMAS), 2015, pp. 1607–1614.
[19] A.V. Segal, I.D. Reid, Hybrid inference optimization for robust pose graph estimation, in: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), IEEE, 2014, pp. 2675–2682.
[20] Y. Feldman, V. Indelman, Bayesian viewpoint-dependent robust classification under model and localization uncertainty, in: IEEE Intl. Conf. on Robotics and Automation (ICRA), 2018.
[21] K. Ok, K. Liu, K. Frey, J. How, N. Roy, Robust object-based Slam for high-speed autonomous navigation, in: IEEE Intl. Conf. on Robotics and Automation (ICRA), 2019, pp. 669–675.
[22] N. Atanasov, B. Sankaran, J. Ny, G.J. Pappas, K. Daniilidis, Nonmyopic view planning for active object classification and pose estimation, IEEE Trans. Robot. 30 (2014) 1078–1090.
[23] T. Patten, W. Martens, R. Fitch, Monte Carlo planning for active object classification, Auton. Robots 42 (2) (2018) 391–421.
[24] L. Burks, I. Loefgren, N. Ahmed, Optimal continuous state pomdp planning with semantic observations: a variational approach, IEEE Trans. Robot. 35 (6) (2019) 1488–1507.
[25] R. Faddoul, W. Raphael, A.-H. Soubra, A. Chateauneuf, Partially observable Markov decision processes incorporating epistemic uncertainties, Eur. J. Oper. Res. 241 (2) (2015) 391–401.
[26] A. Hayashi, D. Ruiken, C. Goerick, T. Hasegawa, Online adaptation of uncertain models using neural network priors and partially observable planning, in: IEEE Intl. Conf. on Robotics and Automation (ICRA), IEEE, 2019, pp. 2440–2446.

[27] B. Lütjens, M. Everett, J.P. How, Safe reinforcement learning with model uncertainty estimates, arXiv preprint, arXiv:1810.08700, 2018.
[28] Y. Wang, X. Tao, X. Shen, J. Jia, Wide-context semantic image extrapolation, in: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 1399–1408.
[29] B. Mildenhall, P. Srinivasan, M. Tancik, J. Barron, R. Ramamoorthi, R. Ng, Nerf: representing scenes as neural radiance fields for view synthesis, in: European Conf. on Computer Vision (ECCV), Springer, 2020, pp. 405–421.
[30] P. Ammirato, P. Poirson, E. Park, J. Kosecka, A.C. Berg, A dataset for developing and benchmarking active vision, in: IEEE International Conference on Robotics and Automation (ICRA), 2017.
[31] F. Dellaert, Factor graphs and GTSAM: a hands-on introduction, Tech. Rep. GT-RIM-CP&R-2012-002, Georgia Institute of Technology, September 2012.
[32] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, F. Dellaert, iSAM2: incremental smoothing and mapping using the Bayes tree, Int. J. Robot. Res. 31 (2) (2012) 217–236.
[33] M. Hsiao, M. Kaess, MH-iSAM2: multi-hypothesis iSAM using Bayes tree and hypo-tree, in: IEEE Intl. Conf. on Robotics and Automation (ICRA), 2019.
[34] L. Kavraki, P. Svestka, J.-C. Latombe, M. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Trans. Robot. Autom. 12 (4) (1996) 566–580.
[35] A. Singh, J. Sha, K.S. Narayan, T. Achim, P. Abbeel, BigBIRD: a large-scale 3d database of object instances, in: 2014 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2014, pp. 509–516.
[36] Z. Sunberg, M. Kochenderfer, Online algorithms for pomdps with continuous state, action, and observation spaces, in: Proceedings of the International Conference on Automated Planning and Scheduling, vol. 28, 2018.
[37] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint, arXiv:1409.1556, 2014.
[38] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: a large-scale hierarchical image database, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, CVPR 2009, IEEE, 2009, pp. 248–255.
[39] T. Minka, Estimating a Dirichlet distribution, www.stat.cmu.edu/~minka/papers/dirichlet, 2003.