

Point-based Value Iteration for Neuro-Symbolic POMDPs

Rui Yan^a, Gabriel Santos^a, Gethin Norman^b, David Parker^a,
Marta Kwiatkowska^a

^a*Department of Computer Science, University of Oxford, Oxford, OX1 3QD, UK*

^b*School of Computing Science, University of Glasgow, Glasgow, G12 8QQ, UK*

Abstract

Neuro-symbolic artificial intelligence is an emerging area that combines traditional symbolic techniques with neural networks. In this paper, we consider its application to sequential decision making under uncertainty. We introduce neuro-symbolic partially observable Markov decision processes (NS-POMDPs), which model an agent that perceives a continuous-state environment using a neural network and makes decisions symbolically, and study the problem of optimising discounted cumulative rewards. This requires functions over continuous-state beliefs, for which we propose a novel piecewise linear and convex representation (P-PWLC) in terms of polyhedra covering the continuous-state space and value vectors, and extend Bellman backups to this representation. We prove the convexity and continuity of value functions and present two value iteration algorithms that ensure finite representability by exploiting the underlying structure of the continuous-state model and the neural perception mechanism. The first is a classical (exact) value iteration algorithm extending α -functions of Porta *et al* (2006) to the P-PWLC representation for continuous-state spaces. The second is a point-based (approximate) method called NS-HSVI, which uses the P-PWLC representation and belief-value induced functions to approximate value functions from below and above for two types of beliefs, particle-based and region-based. Using a prototype implementation, we show the practical applicability of our ap-

Email addresses: rui.yan@cs.ox.ac.uk (Rui Yan), gabriel.santos@cs.ox.ac.uk (Gabriel Santos), gethin.norman@glasgow.ac.uk (Gethin Norman), david.parker@cs.ox.ac.uk (David Parker), marta.kwiatkowska@cs.ox.ac.uk (Marta Kwiatkowska)

proach on two case studies that employ (trained) ReLU neural networks as perception functions, dynamic car parking and an aircraft collision avoidance system, by synthesising (approximately) optimal strategies. An experimental comparison with the finite-state POMDP solver SARSOP demonstrates that NS-HSVI is more robust to particle disturbances.

Keywords:

Neuro-symbolic systems, continuous-state POMDPs, point-based value iteration, heuristic search value iteration

1. Introduction

An emerging trend in artificial intelligence is to integrate traditional symbolic techniques with data-driven components in sequential decision making and optimal control. Application domains include mobile robotics [1], visual reasoning [2], autonomous driving [3] and aircraft control [4]. In real-world autonomous navigation systems, agents rely on unreliable sensors to perceive the environment, typically represented using continuous-state spaces, and planning and control must deal with environmental uncertainty. Neural networks (NNs) have proven effective in these complex settings at providing fast data-driven perception mechanisms capable of performing tasks such as object detection or localisation, and are increasingly often deployed in conjunction with conventional controllers. Because of the potential applicability in safety-critical domains, there is growing interest in methodologies for automated optimal policy synthesis for such *neuro-symbolic systems*, which are currently lacking.

Partially observable Markov decision processes (POMDPs) provide a convenient mathematical framework to plan under uncertainty. Solving POMDPs in a scalable and efficient manner is already challenging for finite-state models [5, 6], but significant progress has been made, e.g., through point-based methods [7], which extend the classic value iteration algorithm for MDPs by applying it to a selected set of *belief states* of the POMDP. Typically, a belief state is a distribution over the states of the model representing an agent’s knowledge about the current state. Since the resulting belief MDP is infinite-state, conventional value iteration cannot be directly applied and instead point-based methods rely on a so-called α -*vector* parameterisation, a linear function characterised by its values in the vertices of the belief simplex, which is finitely representable since the value function is piecewise linear and

convex.

Compared to finite-state POMDPs, solving continuous-state POMDPs suffers from additional challenges due to the uncountably infinite underlying state space. The common approach to discretise or approximate the continuous components with a grid and use methods for finite-state POMDPs may compromise accuracy and lead to an exponential growth in the number of states. Refinement of the discretization to improve accuracy further increases computational complexity. Therefore, an important research direction is to instead consider POMDP solution techniques that operate directly in continuous domains.

Additionally, belief spaces for continuous-state POMDPs have infinitely many dimensions, which further complicates the problem. Since functions over continuous spaces can have arbitrary forms not amenable to computation, a key challenge is finding an efficient representation of the value function that allows closed-form belief updates and Bellman backups for the underlying (parameterisable) transition and reward functions. This problem was addressed by Porta *et al* in [8], where it was proved that continuous-state POMDPs with discrete observations and actions have a piecewise linear and convex value function and admit a finite representation in terms of so-called α -functions, which generalise α -vectors by replacing weighted summation with integration. Working with a representation in terms of linear combinations of Gaussian mixtures, they derive point-based value iteration and implement it by randomly sampling belief points to approximate the value function.

In this paper, we address the problem of optimal policy synthesis for discounted cumulative rewards on a subclass of continuous-state POMDPs with discrete observations and actions, called *neuro-symbolic POMDPs* (NS-POMDPs), whose transition and reward functions are symbolic while observation functions are synthesised in a data-driven fashion, e.g., by means of NNs. The strengths of NNs include trainability from data and fast inference for complex scenarios (e.g., object detection and recognition), while symbolic approaches can provide high interpretability, provable correctness guarantees and ease of inserting human expert knowledge into the underlying systems [9]. Our model is expressive enough for realistic perception functions, while being sufficiently tractable to solve.

Working directly with continuous state spaces rather than a discretisation, we propose novel finite representations of the value function inspired by the α -functions of [8], prove convergence and continuity of the value func-

tion, and present two algorithms for this representation, the classical value iteration (VI), and a variant of the HSVI (Heuristic Search Value Iteration) algorithm [10]. Our first main contribution is demonstrating that, by exploiting the structure of NS-POMDPs, one can indeed find an α -function representation, namely *piecewise linear and convex representation under piecewise constant α -functions (P-PWLC)*, that has a simple parameterisation and is closed with respect to belief updates and the Bellman operator. More specifically, we show that value functions can be represented using pointwise maxima of piecewise constant α -functions (a finite set of polyhedra and a value vector), which can be obtained directly as the preimage of the (NN) perception function, in conjunction with mild assumptions that ensure closure with respect to the transition and reward functions of NS-POMDPs. In contrast to [8], where Gaussian mixtures are used to represent α -, transitions and reward functions, thus possibly requiring approximation of NS-POMDPs, our representation closely matches the structure of NS-POMDPs, even with NN perception functions.

Since α -functions for VI increase exponentially in the number of observations, our second main contribution is a continuous-state space variant of HSVI, called NS-HSVI, for scalable computation of approximate value function from below and above. Starting with the polyhedral preimage of the model’s NN perception function, NS-HSVI works by progressively subdividing the continuous state space during value backups to compute lower bounds, and is able to track the evolution of the system without requiring a priori knowledge about how to discretise the state space. We use a lower K -Lipschitz envelope of a convex hull to approximate an upper bound. We formulate two representations of the belief space, which have closed forms for the quantities of interest: *particle-based*, which relies on sampling of individual points, and *region-based*, which places a (uniform) distribution over a region of continuous space.

We develop a prototype implementation of the techniques and provide experimental results for strategy (policy) synthesis for particle- and region-based beliefs on two case studies: dynamic car parking and an aircraft collision avoidance system. We find that region-based values are more robust to disturbance than particle-based. We also compare our particle-based NS-HSVI to a finite-state POMDP approximation of an NS-POMDP model using SARSOP, and observe that our method consistently yields tighter lower bound values, at a higher computational cost due to expensive polyhedra computations, because the accuracy of SARSOP’s lower bound depends on

the length of the horizon considered when building the model.

Contributions. In summary, this paper makes the following contributions.

1. We propose neuro-symbolic POMDPs (NS-POMDPs), a subclass of continuous-state POMDPs with discrete observations and actions, whose observation functions are synthesised in a data-driven fashion.
2. We propose a novel piecewise constant α -function representation of the value function (as a pointwise maximum function over a set of piecewise constant α -functions defined over the continuous-state space). We show that this representation admits a finite polyhedral representation and is closed with respect to the Bellman operator.
3. We prove continuity and convexity of the value function for discounted cumulative rewards and derive a value iteration (VI) algorithm.
4. We present a new point-based method called NS-HSVI for approximating values of NS-POMDPs, proving that piecewise constant α -functions are a suitable representation for lower bound approximations of values. We develop two variants of the algorithm, one based on the popular particle-based beliefs and the other on novel region-based beliefs, and show they have closed forms for computing the quantities of interest.
5. We provide experimental results to demonstrate the applicability of NS-HSVI in practice for neural networks whose preimage (or that of their approximation) is in polyhedral form.

Related work. Various approaches have been proposed to solve continuous-state POMDPs, including point-based value iteration [8, 11, 12], simulation-based policy iteration [13], discrete space approximation [14], locally-valid approximation [15] and tree search planning [16]. However, these approaches focus on traditional symbolic systems and, while extended to continuous transitions via sampling [8], they are not adapted to data-driven perception functions. HSVI is a point-based value iteration for finite-state POMDPs [10, 17], which was recently extended to stochastic games [18] and works in the continuous belief space, but, to the best of our knowledge, has not been applied to continuous-state POMDPs.

Approaches based on discretisation suffer from loss of accuracy and exponential growth in the number of states and the finite horizon. The point-based methods of [8, 11, 12] use α -functions, which is similar to our approach, but they represent value functions as Gaussian mixtures or dynamic Bayes nets, which may result in looser approximation for NNs than our polyhedral representation. This is because our P-PWLC representation exploits the underlying piecewise constant structure of the continuous-state model and the neural perception mechanism (for which the value function may not be piecewise constant).

While our VI and NS-HSVI algorithms work directly in the continuous state space of the POMDP, most existing approaches rely on constructing a finite-state POMDP to approximate the continuous-state POMDP and then solving the finite-state model. PBVI [19] was the first point-based algorithm to demonstrate good performance on large POMDPs. HSVI [10, 17] uses effective heuristics to guide the forward exploration towards beliefs that significantly reduce the gap between the upper and lower bounds on the optimal value function. FSVI [20], also a point-based value iteration method, explores the belief space by maintaining the true states, using the optimal value function of the underlying MDP to decide which action to take and then sampling the next states and observations. SARSOP [21], one of the fastest existing point-based algorithms, first approximates the optimally reachable belief space in each iteration by sampling a belief according to its stored lower and upper bound functions, then performs backups at selected nodes in the belief tree and finally prunes the α -vectors that are dominated by others over a neighbourhood of the belief tree.

Formal verification approaches for neuro-symbolic systems have been developed for the non-stochastic case [4] and for stochastic multi-agent systems [22, 23, 24] but under full observability. When the controller is data-driven which is a counterpart to the neural perception, the risk of the closed-loop stochastic systems is verified in [25]. Verified NN-based POMDP policies are synthesised in [26], though only for the finite-state setting. Our focus in this paper is on *optimal* policy synthesis for neuro-symbolic systems, motivated by the need for such guarantees in safety-critical domains. To the best of our knowledge, our approach is the first value computation method for partially observable continuous-state neuro-symbolic systems.

Structure of the paper. The remainder of the paper is structured as follows. Section 2 overviews the preliminaries of the POMDP framework.

Section 3 proposes our model of neuro-symbolic POMDPs, together with its belief MDP, and gives an illustrative example. Section 4 introduces piecewise constant representations for functions in NS-POMDPs, and shows that they have a finite representation (P-PWLC) that ensures closure under the Bellman operator. A new value iteration (VI) algorithm is also proposed, and we prove the convexity and continuity of the value function. Section 5 presents a new HSVI algorithm for NS-POMDPs, which uses P-PWLC functions and belief-value induced functions to approximate the value function from below and above, and considers two belief representations for the implementation. Section 6 presents a prototype implementation and experimental evaluation of our algorithm for solving and optimal strategy synthesis for NS-POMDPs on two case studies. Section 7 concludes the paper. To ease presentation, proofs of the theorems and lemmas have been placed in the Appendix.

2. Background

This section introduces the notation and preliminaries concerning Markov decision processes (MDPs) and their partially observable variant (POMDPs), execution paths and strategies (also called policies), and the construction of the (fully observable) belief MDP of a POMDP.

Notation. The space of probability measures on a Borel space X is denoted $\mathbb{P}(X)$, the space of bounded real-valued functions on X is denoted $\mathbb{F}(X)$ and the subset of piecewise constant (PWC) functions of $\mathbb{F}(X)$ is $\mathbb{F}_C(X)$.

MDPs. We focus on (Borel measurable) continuous-state MDPs, which model a single agent executing in a continuous environment by transitioning probabilistically between states. Formally, an MDP is given as a tuple $\mathbf{M} = (S, Act, \Delta, \delta)$, where S is a Borel measurable set of states, Act a finite set of actions, $\Delta : S \rightarrow 2^{Act}$ an available action function and $\delta : (S \times Act) \rightarrow \mathbb{P}(S)$ a probabilistic transition function.

When in state s of MDP \mathbf{M} , the agent has a choice between available actions $\Delta(s)$ and, if $a \in \Delta(s)$ is chosen, then the probability of moving to state s' is $\delta(s, a)(s')$. A path of \mathbf{M} is a sequence $\pi = s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \dots$ such that $s_i \in S$, $a_i \in \Delta(s_i)$ and $\delta(s_i, a_i)(s_{i+1}) > 0$ for all i . We let $\pi(i) = s_i$ and $\pi[i] = a_i$ for all i . $FPath_{\mathbf{M}}$ is the set of finite paths of \mathbf{M} and $last(\pi)$ is the last state of π for any $\pi \in FPath_{\mathbf{M}}$.

A *strategy (policy)* of \mathbf{M} resolves the choices in each state based on the execution so far. Formally, a strategy σ is a Borel measurable mapping

$\sigma : FPath_M \rightarrow \mathbb{P}(Act)$ such that, if $\sigma(\pi)(a) > 0$, then $a \in \Delta(last(\pi))$. We denote by Σ_M the set of strategies of M . A strategy is memoryless if the choice depends only on the last state of each path and deterministic if it always selects an action with probability 1. Fixing a strategy σ , the behaviour of M from an initial state s is represented by a probability measure \mathbb{P}_s^σ over infinite paths starting in s .

POMDPs. POMDPs are an extension of MDPs, in which the agent cannot perceive the underlying state but instead must infer it based on observations. Formally, a POMDP is a tuple $P = (S, Act, \Delta, \delta, \mathcal{O}, obs)$, where (S, Act, Δ, δ) is an MDP, \mathcal{O} is a finite set of observations and $obs : S \rightarrow \mathcal{O}$ is a labelling of states with observations such that, for any $s, s' \in S$, if $obs(s) = obs(s')$ then $\Delta(s) = \Delta(s')$. Note that the underlying state space of the POMDP is uncountably infinite with a continuous-state structure.

When in a state s of a POMDP P , a strategy cannot determine this state s , but only the observation $obs(s)$. The definitions of paths and strategies for P carry over from MDPs. However, the set of strategies Σ_P of P includes only *observation-based strategies*. Formally, a strategy σ is observation-based if, for paths $\pi = s_0 \xrightarrow{a_0} \dots \xrightarrow{a_{n-1}} s_n$ and $\pi' = s'_0 \xrightarrow{a_0} \dots \xrightarrow{a_{n-1}} s'_n$ such that $obs(s_i) = obs(s'_i)$ for $0 \leq i \leq n$, then we have $\sigma(\pi) = \sigma(\pi')$.

Objectives, values and optimal strategies. We focus on the *discounted accumulated reward* objectives, since they balance the importance of immediate rewards compared to future rewards, and allow optimizing the behaviour over an infinite horizon. We note that the problem of undiscounted reward objectives is undecidable already for finite-state POMDPs. For reward structure $r = (r_A, r_S)$, where $r_A : (S \times Act) \rightarrow \mathbb{R}$ and $r_S : S \rightarrow \mathbb{R}$ are action and state bounded reward functions, the discounted accumulated reward for a path π of a POMDP P is given by:

$$Y(\pi) = \sum_{k=0}^{\infty} \beta^k (r_A(\pi(k), \pi[k]) + r_S(\pi(k)))$$

where $\beta \in (0, 1)$ is the discount factor. Given a state s and strategy σ of P , $\mathbb{E}_s^\sigma[Y]$ denotes the expected value of Y when starting from s under σ . Solving P means finding an *optimal strategy* $\sigma^* \in \Sigma_P$ that maximises the objective function and the (optimal) *value function* $V^* : S \rightarrow \mathbb{R}$ is defined as $V^*(s) = \mathbb{E}_s^{\sigma^*}[Y]$ for $s \in S$.

Belief MDP. A strategy of POMDP P can infer the current state from the observations and actions performed. The usual way of representing this

knowledge is as a *belief* $b \in \mathbb{P}(S)$. In general, observation-based strategies represent more informative than belief-based strategies. However, since we focus on accumulated discounted rewards, under the Markov assumption belief-based strategies carry sufficient information to plan optimally [27], and therefore, for a given objective Y , there exists an *optimal (observation-based) strategy* σ of \mathbf{P} , which can be represented as $\sigma : \mathbb{P}(S) \rightarrow \text{Act}$. The strategy updates its belief b to $b^{a,o}$ via Bayesian inference based on the executed action a and observation o , i.e. for $s' \in S$:

$$b^{a,o}(s') = (P(o \mid s')/P(o \mid b, a)) \int_{s \in S} \delta(s, a)(s')b(s)ds.$$

Using this update we can define the corresponding (fully observable) *belief MDP* in a standard way [28], from which an optimal strategy can be derived. We remark that belief spaces for continuous-state POMDPs are continuous and have infinitely many dimensions.

3. Neuro-Symbolic POMDPs

In this section we introduce our model of neuro-symbolic POMDPs, aimed at scenarios where the agent perceives its environment using a data-driven perception function. While the model admits a wide class of perception functions, including those synthesised using machine learning methods such as regression or random forests, our main focus is on demonstrating practical applicability for models using neural network perception, which are being increasingly used in real-world applications [29] to partition continuous environments. This trend necessitates an integrated and automated approach to model and verify such systems. After defining the model, we give an illustrative example and then describe how a (fully observable) belief MDP can be obtained for an NS-POMDP.

NS-POMDPs. The model of *neuro-symbolic POMDPs* comprises a neuro-symbolic *agent* acting in a continuous-state environment. This model is a single-agent partially observable variant of the fully-observable neuro-symbolic concurrent stochastic game model of [23, 24] and shares its syntax. The agent has finitely many local states and actions, and is endowed with a perception mechanism through which it can observe the state of the environment, recording such observations as (a discrete set of finitely many) *percepts*. Before discussing the special case of NN perception functions, we consider the general case, defined formally as follows.

Definition 1 (Syntax of NS-POMDPs). *An NS-POMDP P comprises an agent $\mathbf{Ag} = (S_A, Act, \Delta_A, obs_A, \delta_A)$ and environment $E = (S_E, \delta_E)$ where:*

- *$S_A = Loc \times Per$ is a set of states for \mathbf{Ag} , where $Loc \subseteq \mathbb{R}^b$ and $Per \subseteq \mathbb{R}^d$ are finite sets of local states and percepts, respectively;*
- *$S_E \subseteq \mathbb{R}^e$ is a closed set of continuous environment states;*
- *Act is a nonempty finite set of actions for \mathbf{Ag} ;*
- *$\Delta_A : S_A \rightarrow 2^{Act}$ is an available action function for \mathbf{Ag} ;*
- *$obs_A : (Loc \times S_E) \rightarrow Per$ is \mathbf{Ag} 's perception function;*
- *$\delta_A : (S_A \times Act) \rightarrow \mathbb{P}(Loc)$ is \mathbf{Ag} 's probabilistic transition function;*
- *$\delta_E : (S_E \times Act) \rightarrow \mathbb{P}(S_E)$ is a finitely-branching probabilistic transition function for the environment.*

NS-POMDPs are a subclass of continuous-state POMDPs with discrete observations (i.e., agent states S_A , which are pairs consisting of a local state and percept) and actions. This model captures a number of key properties of POMDP models that we target. The environment is continuous, as many real-world systems such as robot navigation are naturally modelled by continuous states, and probabilities are used to account for uncertainties. At the same time, the agent's state space is finite to ensure tractability.

The system executes as follows. A (global) state for an NS-POMDP P comprises an agent state $s_A = (loc, per)$, where loc is its local state and per is the percept, and environment state s_E . In state $s = (s_A, s_E)$, the agent \mathbf{Ag} chooses an action a available in s_A , then updates its local state to loc' according to the distribution $\delta_A(s_A, a)$ and, at the same time, the environment updates its state to s'_E according to $\delta_E(s_E, a)$. Finally, the agent, based on loc' (since it may require different information regarding the environment depending on its local state), observes s'_E to generate a new percept $per' = obs_A(loc', s'_E)$ and P reaches the state $s' = ((loc', per'), s'_E)$.

While the NS-POMDP model admits any (deterministic) function obs_A from the continuous environment to percepts, in this work we focus on perception functions implemented via (trained) neural networks $f : \mathbb{R}^{b+e} \rightarrow \mathbb{P}(Per)$, yielding normalised scores over different percept values. A rule is then applied that selects the percept value with the maximum score. While restricting perception to deterministic functions with discrete outputs is limiting,

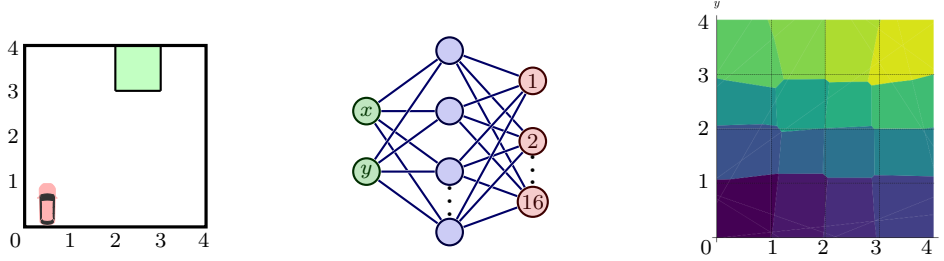


Figure 1: Car parking example, perception NN and perception FCP of its preimage consisting of 62 polygons and 16 classes.

it is well aligned with NN classifiers in applications such as object detection and localisation that we target. A polyhedral decomposition of the continuous state space can then be obtained by computing the preimage of the perception function [30].

To motivate our definition of NS-POMDPs, we consider a dynamic vehicle parking example, in which an autonomous vehicle uses an NN for localisation while looking for a parking spot. We are interested in automated synthesis of an optimal strategy to reach the spot.

Example 1. We consider the problem of an agent Ag (vehicle) looking for the green parking spot (Fig. 1, left). The vehicle uses an NN as a perception mechanism (Fig. 1, middle) that subdivides the continuous environment $\mathcal{R} = \{(x, y) \in \mathbb{R}^2 \mid 0 \leq x, y \leq 4\}$ into 16 cells, resulting in a grid-like abstraction of the environment. We trained an NN with one hidden ReLU layer on randomly generated data to take the coordinates of the vehicle as input and output one of the 16 abstract grid points (percepts). The parking spot region is $\mathcal{R}_P = \{(x, y) \in \mathbb{R}^2 \mid 2 \leq x \leq 3 \wedge 3 \leq y \leq 4\}$. We assume the agent can start from any position and has constant speed.

The environment’s state space \mathcal{R} corresponds to the continuous coordinates of the vehicle, with the percept value stored locally by the agent. The agent’s actions are to move *up*, *down*, *left* or *right*, or *park*, and a *suggested* subset of these actions is associated with each percept (see Table 1). Since the NN is trained from data, the percepts do not perfectly align with the abstract grid (Fig. 1, right), the agent additionally records a trust value to reflect whether actions recommended by the perception function but disallowed (see Table 1), e.g., to *park* before physically reaching the parking spot, are actually taken by the agent.

At each step, the agent updates the trust level in the recommended actions and receives a percept of the environment to keep track of its position

in the abstract grid. Then, the agent takes an action based on the path of previous trust-percept pairs. Next, the agent increases the trust level if the percept is compliant with the executed action and decreases it probabilistically otherwise. The environment's transition function corresponds to the vehicle moving in the direction specified by the agent for a fixed time step. Finally, the agent updates its percept of the updated environment state using its NN observation function.

Formally, the car parking example can be modeled as an NS-POMDP with the following components.

- $S_A = Loc \times Per$ where $Loc = \{1, \dots, 5\}$ (local states) are the 5 trust levels and $Per = \{1, \dots, 16\}$ (percepts) are the 16 abstract grid points which are ordered according to Table 1.
- $S_E = \mathcal{R} = \{(x, y) \in \mathbb{R}^2 \mid 0 \leq x, y \leq 4\}$.
- $Act = \{up, down, left, right, park\}$.
- For $(tr, per) \in S_A$ we have $\Delta_A(tr, per) = Act$ if $per = 15$ and otherwise $\Delta_A(tr, per) = \{up, down, left, right\}$.
- For $tr \in Loc$ and $(x, y) \in S_E$ we have $obs_A(tr, (x, y)) = \text{argmax}(f(x, y))$, where f is implemented via a feed-forward NN with one hidden ReLU layer and 14 neurons, takes the coordinate vector of the vehicle as input and then outputs one of the 16 abstract grid points (Fig. 1, middle). The boundary coordinate is resolved by assigning the grid point with the smallest label.
- For $s_A = (tr, per) \in S_A$, $tr' \in Loc$ and $a \in Act$, if a is compliant with per , see Table 1, then:

$$\delta_A(s_A, a)(tr') = \begin{cases} 1 & \text{if } (tr \leq 4) \wedge (tr' = tr + 1) \\ 1 & \text{if } (tr = 5) \wedge (tr' = tr) \\ 0 & \text{otherwise} \end{cases}$$

on the other hand, if a is not compliant with per , then:

$$\delta_A(s_A, a)(tr') = \begin{cases} 0.5 & \text{if } (tr \geq 2) \wedge (tr' = tr - 1) \\ 0.5 & \text{if } (tr \geq 2) \wedge (tr' = tr) \\ 1 & \text{if } (tr = 1) \wedge (tr' = tr) \\ 0 & \text{otherwise.} \end{cases}$$

Cell label (<i>per</i>)	Abstract grid point	Suggested actions	Cell label (<i>per</i>)	Abstract grid point	Suggested actions
1	(1, 1)	<i>up, right</i>	9	(1, 3)	<i>up, right</i>
2	(2, 1)	<i>up, right</i>	10	(2, 3)	<i>up, right</i>
3	(3, 1)	<i>up</i>	11	(3, 3)	<i>up</i>
4	(4, 1)	<i>up, left</i>	12	(4, 3)	<i>up, left</i>
5	(1, 2)	<i>up, right</i>	13	(1, 4)	<i>right</i>
6	(2, 2)	<i>up, right</i>	14	(2, 4)	<i>right</i>
7	(3, 2)	<i>up</i>	15	(3, 4)	<i>park</i>
8	(4, 2)	<i>up, left</i>	16	(4, 4)	<i>left</i>

Table 1: Suggested actions for each percept of car parking (4×4), where the abstract grid point (i, j) is the i th and j th cell along the positive x -axis and y -axis, respectively.

- For $(x, y), (x', y') \in S_E$ and $a \in Act$ if $x'' = x + \Delta t d_{ax}$ and $y'' = y + \Delta t d_{ay}$, then

$$\delta_E((x, y), a)(x', y') = \begin{cases} 1 & \text{if } (x'', y'') \in \mathcal{R} \text{ and } (x', y') = (x'', y'') \\ 1 & \text{if } (x'', y'') \notin \mathcal{R} \text{ and } (x', y') = (x, y) \\ 0 & \text{otherwise} \end{cases}$$

where $\Delta t = 1.0$ is the time step and $d_a = (d_{ax}, d_{ay})$ is the direction of movement of the action a , e.g., $d_{up} = (0, 1)$ and $d_{left} = (-1, 0)$. ■

NS-POMDP semantics. The semantics of an NS-POMDP \mathbf{P} is a POMDP $\llbracket \mathbf{P} \rrbracket$ over the product of the (discrete) states of the agent and the (continuous) states of the environment, except that we restrict those to states that are percept compatible. A state $s = ((loc, per), s_E)$ is *percept compatible* if $per = obs_A(loc, s_E)$. The semantics of an NS-POMDP is closed with respect to percept compatible states.

Definition 2 (Semantics of NS-POMDPs). *Given an NS-POMDP \mathbf{P} , the semantics of \mathbf{P} is the POMDP $\llbracket \mathbf{P} \rrbracket = (S, Act, \Delta, \delta, S_A, obs)$ where:*

- $S \subseteq S_A \times S_E$ is the set of percept compatible states, which contain both discrete and continuous elements;
- $\Delta(s_A, s_E) = \Delta_A(s_A)$ for $(s_A, s_E) \in S$;
- $obs(s_A, s_E) = s_A$ for $(s_A, s_E) \in S$;
- for $s = (s_A, s_E), s' = (s'_A, s'_E) \in S$ and $a \in \Delta(s)$, if $s'_A = (loc', per')$ is percept compatible, then $\delta(s, a)(s') = \delta_A(s_A, a)(loc')\delta_E(s_E, a)(s'_E)$ and $\delta(s, a)(s') = 0$ otherwise.

Since δ_E has finite branching and S_A is finite, the branching set $\Theta_{s_E}^a = \{s'_E \mid \delta_E(s_E, a)(s'_E) > 0\}$ is finite for all $s_E \in S_E$ and $a \in Act$, and the branching set $\Theta_s^a = \{s' \mid \delta(s, a)(s') > 0\}$ is finite for all $s \in S$ and $a \in \Delta(s)$. Note that, while NS-POMDPs are finite branching, they are not discrete.

NS-POMDP strategies. As $\llbracket P \rrbracket$ is a POMDP, we consider *observation-based strategies*, which can be represented by memoryless strategies over its belief MDP $\llbracket P \rrbracket_B$. Given agent state $s_A = (loc, per)$, we let $S_E^{s_A} = \{s_E \in S_E \mid obs_A(loc, s_E) = per\}$, i.e., the environment states generating percept *per* given *loc*. Since agent states are observable and states of $\llbracket P \rrbracket$ are percept compatible, beliefs can be represented as pairs (s_A, b_E) , where $s_A \in S_A$ is an agent state, $b_E \in \mathbb{P}(S_E)$ is a belief over environment, and $b_E(s_E) = 0$ for all $s_E \in S_E \setminus S_E^{s_A}$, i.e., those states that are not percept compatible.

Before giving the definition of $\llbracket P \rrbracket_B$, we consider how beliefs are updated in this setting. Therefore, suppose s_A is the current agent state, i.e., what is observable, and b_E is the current belief about the environment. Then if action a is executed and s'_A is observed, the updated belief is such that for any $s'_E \in S_E$:

$$b_E^{s_A, a, s'_A}(s'_E) = \frac{P((s'_A, s'_E) \mid (s_A, b_E), a)}{P(s'_A \mid (s_A, b_E), a)} \text{ if } s'_E \in S_E^{s'_A} \text{ and equals 0 otherwise.} \quad (1)$$

Belief MDP and belief updates. We can now derive the belief MDP of an NS-POMDP, which follows through a standard construction [28] while relying on Borel measurability of the underlying uncountable state space of the NS-POMDP.

Definition 3 (Belief MDP). *The belief MDP of an NS-POMDP P is the MDP $\llbracket P \rrbracket_B = (S_B, Act, \Delta_B, \delta_B)$, where:*

- $S_B \subseteq S_A \times \mathbb{P}(S_E)$ is the set of percept compatible beliefs;
- $\Delta_B(s_A, b_E) = \Delta_A(s_A)$ for $(s_A, b_E) \in S_B$;
- for $(s_A, b_E), (s'_A, b'_E) \in S_B$, and $a \in \Delta_B(s_A, b_E)$:

$$\delta_B((s_A, b_E), a)(s'_A, b'_E) = \begin{cases} P(s'_A \mid (s_A, b_E), a) & \text{if } b'_E = b_E^{s_A, a, s'_A} \\ 0 & \text{otherwise.} \end{cases}$$

Finally, in this section we discuss how the beliefs and probabilities of Definition 3 can be computed. For any $(s_A, b_E), (s'_A, b'_E) \in S_B$ and $s'_A = (loc', per')$, we have that $P(s'_A \mid (s_A, b_E), a)$ equals:

$$\delta_A(s_A, a)(loc') \left(\int_{s_E \in S_E} b_E(s_E) \sum_{s'_E \in S_E^{s'_A}} \delta_E(s_E, a)(s'_E) ds_E \right). \quad (2)$$

Furthermore, $P((s'_A, s'_E) \mid (s_A, b_E), a)$ equals:

$$\delta_A(s_A, a)(loc') \left(\int_{s_E \in S_E} b_E(s_E) \delta_E(s_E, a)(s'_E) ds_E \right) \quad (3)$$

if $s'_E \in S_E^{s'_A}$ and 0 otherwise. Thus, using (1) we have that $b_E^{s_A, a, s'_A}(s'_E)$ equals:

$$\frac{\int_{s_E \in S_E} b_E(s_E) \delta_E(s_E, a)(s'_E) ds_E}{\int_{s_E \in S_E} b_E(s_E) \sum_{s''_E \in S_E^{s'_A}} \delta_E(s_E, a)(s''_E) ds_E} \text{ if } s'_E \in S_E^{s'_A} \text{ and 0 otherwise.} \quad (4)$$

We note that the belief MDP $\llbracket \mathbf{P} \rrbracket_B$ is continuous and infinite-dimensional, with finite branching. Thus, solving it exactly is intractable as closed-form operations and parametric forms for continuous functions are required. For efficient computation, beliefs also need to be in closed form.

4. Value Iteration

A common approach to solving continuous-state POMDPs is to discretise or approximate the continuous components with a grid and use methods for finite-state POMDPs. As this may compromise accuracy and leads to an exponential growth in the number of states, we instead aim to operate directly in the continuous domain. Since functions over continuous spaces can have arbitrary forms not amenable to computation, we will extend α -functions to the setting of NS-POMDPs, aided by the theoretical formulation of [8], where it was proved that continuous-state POMDPs with discrete observations and actions have a piecewise linear and convex value function. Rather than work with Gaussian mixtures as in [8], which would require approximations, we will directly exploit the structure of the model to induce a finite (polyhedral) representation of the value function.

More specifically, in this section we show that *piecewise constant* representations for the perception, reward and transition functions are sufficient for NS-POMDPs under mild assumptions, in the sense that they offer a finite

representation and are closed with respect to belief update and the Bellman operator. We next propose a value iteration (VI) algorithm that utilises piecewise constant α -functions, which does not scale but serves as a basis for designing a practical point-based algorithm in Section 5. We conclude this section by investigating the convexity and continuity of the value function.

Value functions. We work with the belief MDP $\llbracket \mathbf{P} \rrbracket_B = (S_B, Act, \Delta_B, \delta_B)$ of an NS-POMDP \mathbf{P} and consider discounted accumulated reward objectives Y . The *value function* is given by $V^* : S_B \rightarrow \mathbb{R}$, where $V^*(s_A, b_E) = \mathbb{E}_{(s_A, b_E)}^*[Y]$ for all $(s_A, b_E) \in S_B$. We require the following notation to evaluate beliefs through a function over the state space S . Given $f : S \rightarrow \mathbb{R}$ and belief (s_A, b_E) , let:

$$\langle f, (s_A, b_E) \rangle = \int_{s_E \in S_E} f(s_A, s_E) b_E(s_E) ds_E \quad (5)$$

for which an integral over S_E is required.

Recall that $\mathbb{F}(S_B)$ denotes the space of functions over the beliefs.

Definition 4 (Bellman operator). *Given $V \in \mathbb{F}(S_B)$, the operator $T : \mathbb{F}(S_B) \rightarrow \mathbb{F}(S_B)$ is defined as follows: $[TV](s_A, b_E)$ equals*

$$\max_{a \in \Delta_A(s_A)} \left\{ \langle R_a, (s_A, b_E) \rangle + \beta \sum_{s'_A \in S_A} P(s'_A \mid (s_A, b_E), a) V(s'_A, b_E^{s_A, a, s'_A}) \right\} \quad (6)$$

for $(s_A, b_E) \in S_B$, where $R_a(s) = r_A(s, a) + r_S(s)$ for $s \in S$.

Since $\llbracket \mathbf{P} \rrbracket_B$, the semantics of NS-POMDP \mathbf{P} , is a continuous-state POMDP with discrete observations and actions, according to [8] the value function V^* is the unique fixed point of the operator T , and thus, theoretically, value iteration can be used to compute V^* . However, as the functions involved are defined over probability density functions from $\mathbb{P}(S_E)$ and S_E is a continuous space, to ensure feasible computation we require a finite parameterisable representation for the value function. To this end, we will extend the class of α -functions with special structure introduced for continuous-state POMDPs in [8], which generalise α -vector representations for finite-state POMDPs [31].

We first observe that perception functions are piecewise constant (PWC), and can therefore be used to induce a finite partition of the continuous state space consisting of connected and observationally-equivalent *regions* by computing the preimage of the perception function. We then impose mild assumptions on the NS-POMDP structure (Assumption 1) to ensure that the agent and environment transition functions preserve the PWC properties of this partition, and on the reward function to ensure region-based reward accumulation (Assumption 2).

4.1. PWC Representations

A *finite connected partition (FCP)* of S , denoted Φ , is a finite collection of disjoint connected subsets (regions) that cover S .

Definition 5 (PWC function). *A function $f : S \rightarrow \mathbb{R}$ is piecewise constant (PWC) if there exists an FCP Φ of S such that $f : \phi \rightarrow \mathbb{R}$ is constant for all $\phi \in \Phi$. Such an FCP Φ is called constant-FCP of S for f .*

Since the perception function is PWC, we can show that the continuous-state space of an NS-POMDP can be decomposed into a finite set of regions such that the states in each region have the same observation.

Lemma 1 (Perception FCP). *There exists a smallest FCP of S , called the perception FCP, denoted Φ_P , such that all states in any $\phi \in \Phi_P$ are observationally equivalent, i.e., if $(s_A, s_E), (s'_A, s'_E) \in \phi$, then $s_A = s'_A$ and we let $s_A^\phi = s_A$.*

The perception FCP Φ_P can be used to find the set $S_E^{s'_A}$ for any agent state $s'_A \in S_A$ over which we integrate beliefs in closed form, see e.g., (2) and (4). If the perception function obs_A is specified as an NN, the corresponding FCP Φ_P can be extracted, or approximated, by analyzing its pre-image [30], which can be computed offline.

Implementing the transition functions δ and δ_E over continuous-state spaces is intractable. Since the perception function induces a decomposition into a finite set of regions, we further assume that such a decomposition is preserved under the transition function, so that states in a given FCP region reach the same regions of some other FCP under δ (and likewise the same rewards). We assume that δ_E is represented by a probabilistic choice over $N_e \in \mathbb{N}$ (deterministic) continuous transition functions and that the reward structure is bounded PWC.

Assumption 1 (Transitions). *For $a \in Act$ and FCP Φ of S , there exists an FCP Φ' of S , called the pre-image FCP of Φ for a , where for $\phi \in \Phi$ and $\phi' \in \Phi'$ either $\Theta_s^a \cap \phi = \emptyset$ for all $s \in \phi'$, or $\Theta_s^a \cap \phi \neq \emptyset$ for all $s \in \phi'$ and if $s, \tilde{s} \in \phi'$, then $\sum_{s' \in \Theta_s^a \cap \phi} \delta(s, a)(s') = \sum_{\tilde{s}' \in \Theta_{\tilde{s}}^a \cap \phi} \delta(\tilde{s}, a)(\tilde{s}')$. Furthermore, $\delta_E = \sum_{i=1}^{N_e} \mu_i \delta_E^i$ where $\delta_E^i : (S_E \times Act) \rightarrow S_E$ is piecewise continuous, $\mu_i \geq 0$ and $\sum_{i=1}^{N_e} \mu_i = 1$.*

Assumption 2 (Rewards). *The reward functions $r_A(\cdot, a), r_S : S \rightarrow \mathbb{R}$ are bounded PWC for all $a \in \text{Act}$. Therefore, for each action $a \in \text{Act}$, there exists a smallest FCP of S , called the reward FCP under action a and denoted Φ_R^a , such that all states in any $\phi \in \Phi_R^a$ have the same rewards, i.e., if $s, s' \in \phi$, then $r_A(s, a) = r_A(s', a)$ and $r_S(s) = r_S(s')$.*

Example 2. Fig. 1 (right) shows an FCP representation for the pre-image of the perception function of Example 1. The FCP was constructed via the exact computation method from [30], and is composed of 62 polygons. Each colour indicates one of the grid cells as perceived by the agent. In the reward structure, all action rewards are zero and the state reward function is such that for any $(s_A, (x, y)) \in S$: $r_S(s_A, (x, y)) = 1000$ if $(x, y) \in \mathcal{R}_P$ and 0 otherwise, i.e., there is a positive reward if the parking spot is found. ■

We emphasize that, although the states in any region of the perception FCP are observationally equivalent, by Assumption 1 the transitions have finite representations, and by Assumption 2 the states in any region of the reward FCP have the same reward, such states can still have different values as taking the same actions can yield paths that need not be observationally equivalent. Therefore, the value function V^* may not be piecewise constant. Our results demonstrate that analysing NS-POMDPs under these PWC restrictions remains challenging, since any discretisation would imply that all states contained in an abstract region have the same sequences of transitions and rewards given a sequence of actions, and thus have the same value. Thus, it is not possible to construct, a priori, a partition of the state space that reduces the problem to finding the values of a finite-state POMDP. It would be possible to find, from some initial belief, all reachable states up to some finite depth and then compute an approximate value for the initial belief. However, this approach can yield an exponential blow up in the number of beliefs, or even infinitely many reachable states, for instance, when the initial belief has positive probabilities over a region of the continuous-state space.

Instead of unrolling, our algorithm progressively subdivides the continuous state space during value backups. Additionally, we remark that finite branching of the environment transition function does not make the NS-POMDP discrete because, unlike in finite-state POMDPs, these transitions, represented by a finite number of piecewise continuous transition functions, cannot be characterized via a finite set of state-to-state transitions. Besides, if the current belief has positive probabilities over an infinite number

of states, then the updated belief can also have an infinite number of states with positive probabilities.

4.2. PWC α -Function Value Iteration

We can now show, utilising the results for continuous-state POMDPs [8], that V^* is the limit of a sequence of α -functions, called *piecewise linear and convex under PWC α -functions (P-PWLC)*, where each such function can be represented by a (finite) set of PWC functions (concretely, as a finite set of FCP regions and a value vector).

Definition 6 (P-PWLC function). *A function $V : S_B \rightarrow \mathbb{R}$ is piecewise linear and convex under PWC α -functions (P-PWLC) if there exists a finite set $\Gamma \subseteq \mathbb{F}_C(S)$ such that $V(s_A, b_E) = \max_{\alpha \in \Gamma} \langle \alpha, (s_A, b_E) \rangle$ for all $(s_A, b_E) \in S_B$ where the functions in Γ are called PWC α -functions.*

Definition 6 implies that, if $V \in \mathbb{F}(S_B)$ is P-PWLC, then it can be represented by a set Γ of PWC continuous functions over S . For NS-POMDPs, we demonstrate that, under Assumptions 1 and 2, a P-PWLC representation of value functions is closed under the Bellman operator and the value iteration converges.

Theorem 1 (P-PWLC closure and convergence). *If $V \in \mathbb{F}(S_B)$ and P-PWLC, then so is $[TV]$. If $V^0 \in \mathbb{F}(S_B)$ and P-PWLC, then the sequence $(V^t)_{t=0}^\infty$, such that $V^{t+1} = [TV^t]$ are P-PWLC and converges to V^* .*

We remark that an implementation of this exact value iteration is feasible, since each α -function involved is PWC and thus allows for a finite representation. However, as the number of α -functions grows exponentially in the number of agent states, it is not scalable.

4.3. Convexity and Continuity of the Value Function

In Section 5, we will derive a variant of HSVI for lower and upper bounding of the value function, which is more scalable. To this end, the following properties will be required.

Using Theorem 1 the value function can be represented as a pointwise maximum $V^*(s_A, b_E) = \sup_{\alpha \in \Gamma} \langle \alpha, (s_A, b_E) \rangle$ for $(s_A, b_E) \in S_B$, where $\Gamma \subseteq \mathbb{F}_C(S)$ may be infinite. We now show that V^* is convex and continuous for any fixed $s_A \in S_A$. Since we assume bounded reward functions, the value function V^* has lower and upper bounds:

$$L = \min_{s \in S, a \in Act} R_a(s)/(1 - \beta) \quad \text{and} \quad U = \max_{s \in S, a \in Act} R_a(s)/(1 - \beta). \quad (7)$$

Theorem 2 (Convexity and continuity). *For any $s_A \in S_A$, the value function $V^*(s_A, \cdot) : \mathbb{P}(S_E) \rightarrow \mathbb{R}$ is convex and for any $b_E, b'_E \in \mathbb{P}(S_E)$:*

$$|V^*(s_A, b_E) - V^*(s_A, b'_E)| \leq K(b_E, b'_E) \quad (8)$$

where $K(b_E, b'_E) = (U - L) \int_{s_E \in S_E^{b_E > b'_E}} (b_E(s_E) - b'_E(s_E)) ds_E$ and $S_E^{b_E > b'_E} = \{s_E \in S_E^{s_A} \mid b_E(s_E) - b'_E(s_E) > 0\}$.

5. Heuristic Search Value Iteration

Value iteration with point-based updates has been proposed for finite-state POMDPs [8, 10, 7, 17, 32], relying on the fact that performing many fast approximate updates often results in a more useful value function than performing a few exact updates. HSVI [10] approximates the value function V^* at a given initial belief via lower and upper bound functions, which are updated through heuristically generated beliefs. SARSOP [21] improves efficiency, but sacrifices convergence guarantees due to aggressive pruning. These methods focus on finite-state POMDPs and are not directly applicable to continuous-state NS-POMDPs, as they rely on discretisation or approximation.

We now present a new HSVI algorithm for NS-POMDPs, which uses P-PWLC functions and belief-value induced functions to approximate V^* from below and above. This HSVI algorithm progressively subdivides the continuous state space during value backups, to obtain a piecewise constant lower bound and a lower K -Lipschitz envelope of a convex hull upper bound on V^* that itself may not be piecewise constant.

We first introduce the representations of the lower and upper bound functions to the value function, then present point-based updates followed by our HSVI algorithm, and finally consider two belief representations for the implementation, both with closed forms for the quantities of interest, one based on particles (individually sampled points) and the other on regions (polyhedra) of the continuous space.

5.1. Lower and Upper Bound Representations

Lower bound function. Selecting an appropriate representation for α -functions requires closure properties with respect to the Bellman operator, which includes both the transition function and the reward function.

Rather than relying on Gaussian mixtures [8], which require both the transition and reward functions to be in this form, we represent the lower bound $V_{LB}^\Gamma \in \mathbb{F}(S_B)$ as a P-PWLC function for the finite set $\Gamma \subseteq \mathbb{F}_C(S)$ of PWC α -functions (see Definition 6), for which closure is guaranteed by Theorem 1. This is finitely representable as each α -function is PWC. In contrast to Gaussian mixtures, our P-PWLC representation is designed to match the NS-POMDP perfectly, with the necessary closure properties ensured by exploiting the structure of the NS-POMDP.

Upper bound function. The upper bound $V_{UB}^\Upsilon \in \mathbb{F}(S_B)$ is represented by a finite set of belief-value points $\Upsilon = \{((s_A^i, b_E^i), y_i) \mid i \in I\}$, where y_i is an upper bound of $V^*(s_A^i, b_E^i)$. Since $V^*(s_A, \cdot)$ is convex by Theorem 2, letting $I_{s_A} = \{i \in I \mid s_A^i = s_A\}$, for any $\lambda_i \geq 0$ such that $\sum_{i \in I_{s_A}} \lambda_i = 1$, we have:

$$V^*(s_A, \sum_{i \in I_{s_A}} \lambda_i b_E^i) \leq \sum_{i \in I_{s_A}} \lambda_i V^*(s_A^i, b_E^i) \leq \sum_{i \in I_{s_A}} \lambda_i y_i.$$

This fact is used in HSVI for finite-state POMDPs [10], as any new belief is a convex combination of the beliefs in Υ , and therefore the convexity of $V^*(s_A, \cdot)$ yields an upper bound. However, there is no such convex combination guarantee in NS-POMDPs since, as Υ is finite and beliefs are over a continuous-state space, any convex combinations of beliefs in Υ cannot cover the belief space. Therefore, the upper bound V_{UB}^Υ is instead defined as the lower envelope of the lower convex hull of the points in Υ satisfying the following problem:

$$\begin{aligned} V_{UB}^\Upsilon(s_A, b_E) = & \text{minimize } \sum_{i \in I_{s_A}} \lambda_i y_i + K_{UB}(b_E, \sum_{i \in I_{s_A}} \lambda_i b_E^i) \\ & \text{subject to: } \lambda_i \geq 0, \sum_{i \in I_{s_A}} \lambda_i = 1 \text{ for all } (s_A, b_E) \in S_B \end{aligned} \quad (9)$$

where $K_{UB} : \mathbb{P}(S_E) \times \mathbb{P}(S_E) \rightarrow \mathbb{R}$ measures the difference between two beliefs such that, if K is from Theorem 2 showing the continuity of the value function, then for any $b_E, b'_E \in \mathbb{P}(S_E)$:

$$K_{UB}(b_E, b'_E) \geq K(b_E, b'_E) \quad \text{and} \quad K_{UB}(b_E, b_E) = 0. \quad (10)$$

It can be seen that (9) is close to the classical upper bound function used in regular HSVI for finite-state spaces, except for the function K_{UB} that measures the difference between two beliefs (two functions). We require that K_{UB} satisfies (10) to ensure that (9) is an upper bound after a value backup, as stated in Lemma 4 below.

Bound initialization. The lower bound V_{LB}^Γ is initialized using the lower bound of the blind strategies of the form “always choose action $a \in Act$ ”, which is given by $\sum_{k=0}^{\infty} \beta^k \inf_{s \in S} R_a(s)$. Therefore, a lower bound for V_{LB}^Γ is given by:

$$R_{LB} = \max_{a \in Act} \left(\sum_{k=0}^{\infty} \beta^k \inf_{s \in S} R_a(s) \right) = 1/(1 - \beta) \max_{a \in Act} \inf_{s \in S} R_a(s).$$

The PWC α -function set Γ for the initial V_{LB}^Γ contains a single PWC function α , where $\alpha(s) = R_{LB}$ for all $s \in S$ and the associated FCP is the perception FCP Φ_P . We initialize the upper bound V_{UB}^Υ by sampling a set of initial beliefs $\{(s_A^i, b_E^i)\}_{i \in I}$ and letting $y_i = U$ for all (s_A^i, b_E^i) .

5.2. Point-Based Updates

Lower bound updates. For the lower bound V_{LB}^Γ , in each iteration we add a new PWC α -function α^* to Γ leading to Γ' at a belief $(s_A, b_E) \in S_B$ such that:

$$\langle \alpha^*, (s_A, b_E) \rangle = [TV_{LB}^\Gamma](s_A, b_E). \quad (11)$$

To that end, let \bar{a} be an action maximizing the Bellman backup (6) at (s_A, b_E) , i.e., \bar{a} is a maximizer when computing $[TV_{LB}^\Gamma](s_A, b_E)$. If action \bar{a} is taken, then $\bar{S}_A = \{s'_A \in S_A \mid P(s'_A \mid (s_A, b_E), \bar{a}) > 0\}$ are agent states that can be observed. If s'_A is observed, then the backup value at belief (s_A, b_E) from an α -function $\alpha \in \Gamma$ equals $\int_{s_E \in S_E} bval((s_A, s_E), \bar{a}, s'_A, \alpha) b_E(s_E) ds_E$, where for any $s_E \in S_E$:

$$bval((s_A, s_E), \bar{a}, s'_A, \alpha) = \beta \delta_A(s_A, \bar{a})(loc') \sum_{s'_E \in \Theta_{s_E}^{\bar{a}} \cap S_E^{s'_A}} \delta_E(s_E, \bar{a})(s'_E) \alpha(s'_A, s'_E).$$

For $s'_A \in \bar{S}_A$, let $\alpha^{s'_A} \in \Gamma$ be an α -function maximizing the backup value, i.e., $\alpha^{s'_A} \in \arg\max_{\alpha \in \Gamma} \int_{s_E \in S_E} bval((s_A, s_E), \bar{a}, s'_A, \alpha) b_E(s_E) ds_E$.

Using \bar{a} , $\alpha^{s'_A}$ for $s'_A \in \bar{S}_A$ and the perception FCP Φ_P , Algorithm 1 computes a new α -function α^* at belief (s_A, b_E) . To guarantee (11) and improve the efficiency, we only compute the backup values for regions $\phi \in \Phi_P$ over which (s_A, b_E) has positive probabilities, i.e. $s_A^\phi = s_A$ (recall s_A^ϕ is the unique agent state appearing in ϕ) and $\int_{(s_A, s_E) \in \phi} b_E(s_E) ds_E > 0$ and assign the trivial lower bound L otherwise. More precisely, for each such region ϕ and $(\hat{s}_A, \hat{s}_E) \in \phi$:

$$\alpha^*(\hat{s}_A, \hat{s}_E) = R_{\bar{a}}(\hat{s}_A, \hat{s}_E) + \sum_{s'_A \in \bar{S}_A} bval((\hat{s}_A, \hat{s}_E), \bar{a}, s'_A, \alpha^{s'_A}) \quad (12)$$

Algorithm 1 Point-based $Update(s_A, b_E)$ of $(V_{LB}^\Gamma, V_{UB}^\Upsilon)$

- 1: $\bar{a} \leftarrow$ the maximum action in computing $[TV_{LB}^\Gamma](s_A, b_E)$
 - 2: $\bar{S}_A \leftarrow \{s'_A \in S_A \mid P(s'_A \mid (s_A, b_E), \bar{a}) > 0\}$
 - 3: $\alpha^{s'_A} \leftarrow \operatorname{argmax}_{\alpha \in \Gamma} \int_{s_E \in S_E} bval((s_A, s_E), \bar{a}, s'_A, \alpha) b_E(s_E) ds_E$ for all $s'_A \in \bar{S}_A$
 - 4: **for** $\phi \in \Phi_P$ **do**
 - 5: **if** $s_A^\phi = s_A$ and $\int_{(s_A, s_E) \in \phi} b_E(s_E) ds_E > 0$ **then**
 - 6: Compute $\alpha^*(\hat{s}_A, \hat{s}_E)$ by (12) for $(\hat{s}_A, \hat{s}_E) \in \phi$ \triangleright ISPP backup
 - 7: **else** $\alpha^*(\hat{s}_A, \hat{s}_E) \leftarrow L$ for $(\hat{s}_A, \hat{s}_E) \in \phi$
 - 8: $\Gamma \leftarrow \Gamma \cup \{\alpha^*\}$
 - 9: $p^* \leftarrow [TV_{UB}^\Upsilon](s_A, b_E)$
 - 10: $\Upsilon \leftarrow \Upsilon \cup \{((s_A, b_E), p^*)\}$
-

where if $s'_A \notin \bar{S}_A$, then $\alpha^{s'_A}$ can be any α -function in Γ . Computing the backup values (12) state by state is computationally intractable, as region ϕ contains an infinite number of states. However, the following lemma shows that α^* is PWC, thus resulting in a tractable region-by-region backup. The lemma also shows that the lower bound function increases uniformly, is valid after each update, and performs no worse than the Bellman backup at the current belief.

Lemma 2 (Lower bound). *At belief $(s_A, b_E) \in S_B$, the function α^* generated by Algorithm 1 is a PWC α -function satisfying (11), $V_{LB}^\Gamma \leq V_{LB}^{\Gamma'} \leq V^*$ and $V_{LB}^{\Gamma'}(s_A, b_E) \geq [TV_{LB}^\Gamma](s_A, b_E)$.*

Since α^* is PWC, we next present a new backup for (12) through finite region-by-region backups. Recall from Assumption 1 that δ_E can be represented as $\sum_{i=1}^{N_e} \mu_i \delta_E^i$. Algorithm 2 presents an Image-Split-Preimage-Product (ISPP) backup method to compute (12) region by region. This method, inspired by Lemma 2, is to divide a region ϕ into subregions, where for each subregion α^* is constant, illustrated in Fig. 2. Given any reachable local state loc' under \bar{a} and continuous transition function δ_E^i , the *image* of ϕ under \bar{a} and δ_E^i to loc' is divided into *image* regions Φ_{image} such that the states in each region have a unique agent state. Each image region ϕ_{image} is then split into subregions by a constant-FCP of the PWC function $\alpha^{s_A^{\phi_{\text{image}}}}$ by pairwise intersections, and thus Φ_{image} is *split* into a set of refined image regions Φ_{split} . An FCP over ϕ , denoted by Φ_{pre} , is constructed by computing the *preimage* of each $\phi_{\text{image}} \in \Phi_{\text{split}}$ to ϕ . Finally, the *product* of these FCPs Φ_{pre} for all

Algorithm 2 Image-Split-Preimage-Product (ISPP) backup over a region

Input: region ϕ , action \bar{a} , PWC $\alpha^{s'_A}$ for all $s'_A \in S_A$

```

1:  $Loc' \leftarrow \{loc' \in Loc \mid \delta_A(s_A^\phi, \bar{a})(loc') > 0\}$ ,  $\Phi_{\text{product}} \leftarrow \phi$ 
2: for  $loc' \in Loc', i = 1, \dots, N_e$  do
3:    $\phi'_E \leftarrow \{\delta_E^i(s_E, \bar{a}) \mid (s_A^\phi, s_E) \in \phi\}$  ▷ Image
4:    $\Phi_{\text{image}} \leftarrow$  divide  $\phi'_E$  into regions over  $S$  by  $obs_A(loc', \cdot)$ 
5:    $\Phi_{\text{split}} \leftarrow \emptyset$  ▷ Split
6:   for  $\phi_{\text{image}} \in \Phi_{\text{image}}$  do
7:      $\Phi_\alpha \leftarrow$  a constant-FCP of  $S$  for the PWC function  $\alpha^{s_A^{\phi_{\text{image}}}}$ 
8:      $\Phi_{\text{split}} \leftarrow \Phi_{\text{split}} \cup \{\phi_{\text{image}} \cap \phi' \mid \phi' \in \Phi_\alpha\}$ 
9:    $\Phi_{\text{pre}} \leftarrow \emptyset$  ▷ Preimage
10:  for  $\phi_{\text{image}} \in \Phi_{\text{split}}$  do
11:     $\Phi_{\text{pre}} \leftarrow \Phi_{\text{pre}} \cup \{(s_A^\phi, s_E) \in \phi \mid \delta_E^i(s_E, \bar{a}) \in \phi_{\text{image}}\}$ 
12:   $\Phi_{\text{product}} \leftarrow \{\phi_1 \cap \phi_2 \mid \phi_1 \in \Phi_{\text{pre}} \wedge \phi_2 \in \Phi_{\text{product}}\}$  ▷ Product
13:  $\Phi_{\text{product}} \leftarrow \{\phi_1 \cap \phi_2 \mid \phi_1 \in \Phi_{\text{product}} \wedge \phi_2 \in \Phi_R^{\bar{a}}\}$ 
14: for  $\phi_{\text{product}} \in \Phi_{\text{product}}$  do ▷ Value backup
15:   Take one state  $(\hat{s}_A, \hat{s}_E) \in \phi_{\text{product}}$ 
16:    $\alpha^*(\phi_{\text{product}}) \leftarrow R_{\bar{a}}(\hat{s}_A, \hat{s}_E) + \sum_{s'_A \in S_A} bval((\hat{s}_A, \hat{s}_E), \bar{a}, s'_A, \alpha^{s'_A})$ 
17: return:  $(\Phi_{\text{product}}, \alpha^*)$ 

```

reachable local states and environment functions and reward FCP $\Phi_R^{\bar{a}}$, denoted Φ_{product} , is computed. The following lemma demonstrates that α^* is constant in each region of Φ_{product} , and therefore (12) can be computed by finite backups.

Lemma 3 (ISPP backup). *The FCP Φ_{product} returned by Algorithm 2 is a constant-FCP of ϕ for α^* and the region-by-region backup for α^* satisfies (12).*

Computing $bval((\hat{s}_A, \hat{s}_E), \bar{a}, s'_A, \alpha^{s'_A})$ in the value backup requires $\alpha^{s'_A}(s'_A, s'_E)$. To obtain this value, we need to find the region in the constant-FCP for $\alpha^{s'_A}$ containing (s'_A, s'_E) . Instead of searching, we record the region connections during ISPP, and thus can locate the region containing (s'_A, s'_E) improving the efficiency.

Upper bound updates. For the upper bound V_{UB}^{Υ} , working with the representation given in (9), at a belief $(s_A, b_E) \in S_B$ in each iteration, we add

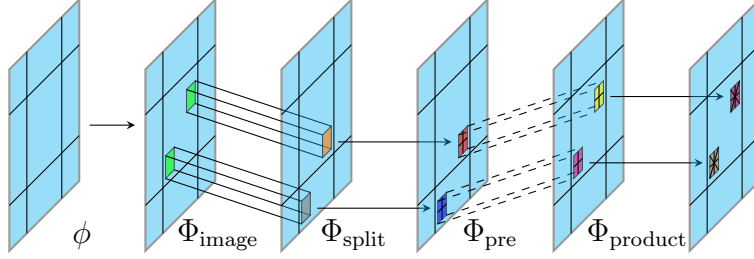


Figure 2: Illustration of the steps taken by the ISPP algorithm.

Algorithm 3 NS-HSVI for NS-POMDPs

```

1: Initialize  $V_{LB}^\Gamma$  and  $V_{UB}^\Upsilon$ 
2: while  $V_{UB}^\Upsilon((s_A^{init}, b_E^{init}) - V_{LB}^\Gamma(s_A^{init}, b_E^{init}) > \varepsilon$  do  $Explore((s_A^{init}, b_E^{init}), \varepsilon, 0)$ 
3: function  $Explore((s_A, b_E), \varepsilon, t)$ 
4:   if  $V_{UB}^\Upsilon(s_A, b_E) - V_{LB}^\Gamma(s_A, b_E) \leq \varepsilon \beta^{-t}$  then return
5:   for  $a \in \Delta_A(s_A), s'_A \in S_A$  do
6:      $p^{a, s'_A} \leftarrow P(s'_A | (s_A, b_E), a) V_{UB}^\Upsilon(s'_A, b_E^{s_A, a, s'_A})$ 
7:    $\hat{a} \leftarrow \operatorname{argmax}_{a \in \Delta_A(s_A)} \langle R_a, (s_A, b_E) \rangle + \beta \sum_{s'_A \in S_A} p^{a, s'_A}$ 
8:    $Update(s_A, b_E)$ 
9:    $\hat{s}_A \leftarrow \operatorname{argmax}_{s'_A \in S_A} excess_{t+1}(s'_A, b_E^{s_A, \hat{a}, s'_A})$ 
10:   $Explore((\hat{s}_A, b_E^{s_A, \hat{a}, \hat{s}_A}), \varepsilon, t + 1)$ 
11:   $Update(s_A, b_E)$ 

```

a new belief-value point $((s_A, b_E), p^*)$ to Υ such that $p^* = [TV_{UB}^\Upsilon](s_A, b_E)$. The following lemma shows that $p^* \geq V^*(s_A, b_E)$ required by (9), the upper bound function is decreasing uniformly, is valid after each update, and performs no worse than the Bellman backup at the current belief.

Lemma 4 (Upper bound). *Given belief $(s_A, b_E) \in S_B$, if $p^* = [TV_{UB}^\Upsilon](s_A, b_E)$, then p^* is an upper bound of V^* at (s_A, b_E) , i.e., $p^* \geq V^*(s_A, b_E)$, and if $\Upsilon' = \Upsilon \cup \{((s_A, b_E), p^*)\}$, then $V_{UB}^\Upsilon \geq V_{UB}^{\Upsilon'} \geq V^*$ and $V_{UB}^{\Upsilon'}(s_A, b_E) \leq [TV_{UB}^\Upsilon](s_A, b_E)$.*

5.3. NS-HSVI Algorithm

Algorithm 3 presents the NS-HSVI algorithm for NS-POMDPs. Similarly to the heuristic search in HSVI [10], the algorithm (lines 5–7) selects an action \hat{a} greedily according to the upper bound at belief $(s_A, b_E) \in S_B$, i.e.,

\hat{a} is a maximizer when computing $[TV_{UB}^{\Upsilon}](s_A, b_E)$. Furthermore, given $\varepsilon > 0$, it selects an agent state \hat{s}_A (observation) with the highest weighted excess approximation gap (line 9), denoted $excess_{t+1}(s'_A, b_E^{s_A, \hat{a}, s'_A})$, which equals:

$$P(s'_A \mid (s_A, b_E), \hat{a})(V_{UB}^{\Upsilon}(s'_A, b_E^{s_A, \hat{a}, s'_A}) - V_{LB}^{\Gamma}(s'_A, b_E^{s_A, \hat{a}, s'_A}) - \varepsilon\beta^{t+1})$$

where t is the depth of (s_A, b_E) from the initial belief $s_B^{init} = (s_A^{init}, b_E^{init}) \in S_B$. NS-HSVI has the following convergence guarantees.

Theorem 3 (NS-HSVI). *Algorithm 3 will terminate and upon termination:*

1. $V_{UB}^{\Upsilon}(s_B^{init}) - V_{LB}^{\Gamma}(s_B^{init}) \leq \varepsilon$;
2. $V_{LB}^{\Gamma}(s_B^{init}) \leq V^*(s_B^{init}) \leq V_{UB}^{\Upsilon}(s_B^{init})$;
3. $V^*(s_B^{init}) - \mathbb{E}_{s_B^{init}}^{\hat{\sigma}}[Y] \leq \varepsilon$ where $\hat{\sigma}$ is the one-step lookahead strategy from V_{LB}^{Γ} .

Proof. Given belief $(s_A, b_E) \in S_B$, through Lemma 2 after updating a lower bound V_{LB}^{Γ} we have:

$$V_{LB}^{\Gamma} \leq V_{LB}^{\Gamma'} \leq V^* \quad \text{and} \quad V_{LB}^{\Gamma'}(s_A, b_E) \geq [TV_{LB}^{\Gamma}](s_A, b_E) \quad (13)$$

and through Lemma 4 after updating an upper bound V_{UB}^{Υ} , we have:

$$V_{UB}^{\Upsilon} \geq V_{UB}^{\Upsilon'} \geq V^* \quad \text{and} \quad V_{UB}^{\Upsilon'}(s_A, b_E) \leq [TV_{UB}^{\Upsilon}](s_A, b_E). \quad (14)$$

Now, since V_{LB}^{Γ} and V_{UB}^{Υ} are initially bounded and from Lemmas 2 and 4 are uniformly improvable, δ has finite branching and $\beta \in (0, 1)$, using [33, Theorem 6.8] we have that Algorithm 3 terminates after finite steps.

Next, combining (13) and (14), and using [33, Section 6.5] both 1 and 2 follow directly. Finally, concerning 3, by (B.1), we have

$$\langle \alpha^*, (\hat{s}_A, \hat{s}_E) \rangle \leq [TV_{LB}^{\Gamma}](\hat{s}_A, \hat{s}_E) \quad (15)$$

for all $(\hat{s}_A, \hat{s}_E) \in S_B$. If $V_{LB}^{\Gamma} \leq TV_{LB}^{\Gamma}$, we have $V_{LB}^{\Gamma'} \leq TV_{LB}^{\Gamma}$ using (15). Then, since Algorithm 3 terminates, according to [33, Theorem 3.18]:

$$V^*(s_A^{init}, b_E^{init}) - \mathbb{E}_{(s_A^{init}, b_E^{init})}^{\hat{\sigma}}[Y] \leq V_{UB}^{\Upsilon}(s_A^{init}, b_E^{init}) - V_{LB}^{\Gamma}(s_A^{init}, b_E^{init}) \leq \varepsilon$$

which completes the proof. \square

Pruning. We apply the following pruning to speed up Algorithm 3. First, a new α -function α^* is added to Γ at belief (s_A, b_E) in each update only if α^* strictly improves the value at (s_A, b_E) , i.e., $\langle \alpha^*, (s_A, b_E) \rangle > V_{LB}^\Gamma(s_A, b_E)$. This leads to fewer α -functions in Γ without changing convergence, and thus faster lower bound computation. Second, for the heuristic search, since the action \hat{a} (line 6) maximizing the upper bound backup may not be unique and different \hat{a} could result in different maximum gaps (line 8), we keep all maximizers and select the pair (\hat{a}, \hat{s}_A) with the largest gap. We find this new excess heuristic to be empirically superior, as it tends to reduce the uncertainty the most.

Convergence. Each belief update of Algorithm 3 is focused on a single belief, and therefore the number of iterations can be higher than value iteration; on the other hand, iterations are cheaper to perform. In the finite-state case, an upper bound on the number of HSVI iterations required can be calculated [33, Theorem 6.8]. However, such analysis would be difficult in our setting, as the number of points to update depends on the initial beliefs, and which beliefs are updated at each iteration, and varies as the algorithm progresses.

5.4. Two Belief Representations

An implementation of the NS-HSVI algorithm crucially depends on the representations of beliefs, as a closed form is needed when computing belief $b_E^{s_A, a, s'_A}$, expected values $\langle \alpha, (s_A, b_E) \rangle$ and $\langle R_a, (s_A, b_E) \rangle$, probability $P(s'_A | (s_A, b_E), a)$ and upper bound $V_{UB}^\Gamma(s_A, b_E)$. We first consider the popular particle-based belief representation and then propose a region-based belief representation to overcome the problem of requiring many particles to converge in the particle-based representation [34].

Particle-based beliefs. Particle-based representations have been widely used in applications from computer vision [35], robotics [36, 8] to machine learning [37]. They can approximate arbitrary beliefs (given sufficient particles), handle nonlinear and non-Gaussian systems, and allow efficient computations.

Definition 7 (Particle-based belief). A belief $(s_A, b_E) \in S_B$ is represented by a weighted particle set $\{(s_E^i, w_i)\}_{i=1}^{N_b}$ with normalized weights if

$$b_E(s_E) = \sum_{i=1}^{N_b} w_i D(s_E - s_E^i)$$

where $w_i > 0$, $s_E^i \in S_E$ for all $1 \leq i \leq N_b$ and $D(s_E - s_E^i)$ is a Dirac delta function centered at 0. Let $B(s_E)$ be a small neighborhood of s_E , and $P(s_E; b_E) = \int_{s_E' \in B(s_E)} b_E(s_E') ds_E'$ be the probability of particle s_E under b_E .

Given an initial particle-based belief (s_A^{init}, b_E^{init}) , the number of states reachable in any finite number of steps is finite, and therefore standard methods for finite-state POMDPs can be used to solve the resulting finite-state game tree, similarly to [22] under fully-observable strategies. However, the size of the game tree can increase exponentially as the number of steps increases, particularly given that the reachable states are likely to be distinct due to the continuous-state space.

To implement NS-HSVI given in Algorithm 3 using particle-based beliefs, we must demonstrate that V_{LB}^Γ and V_{UB}^Υ are eligible representations [8] for particle-based beliefs, that is, there are closed forms for the quantities of interest. For a particle-based belief (s_A, b_E) with weighted particle set $\{(s_E^i, w_i)\}_{i=1}^{N_b}$, it follows from (4) that for belief $b_E^{s_A, a, s_A'}$ we have, for any $s_E' \in S_E$, $b_E^{s_A, a, s_A'}(s_E')$ equals:

$$\frac{\sum_{i=1}^{N_b} w_i \delta_E(s_E^i, a)(s_E')}{\sum_{i=1}^{N_b} w_i \sum_{s_E'' \in S_E^{s_A'}} \delta_E(s_E^i, a)(s_E'')} \text{ if } s_E' \in S_E^{s_A'} \text{ and equals 0 otherwise.} \quad (16)$$

Similarly, we can compute $\langle \alpha, (s_A, b_E) \rangle$, $\langle R_a, (s_A, b_E) \rangle$ and $P(s_A' | (s_A, b_E), a)$ as simple summations. It remains to compute V_{UB}^Υ in (9), which we achieve by designing a function K_{UB} that measures belief differences that satisfy (10). However, (10) is hard to check as, for beliefs b_E and b_E' , calculating $K(b_E, b_E')$ involves the integral over the region $S_E^{b_E > b_E'}$. For particle-based beliefs, we propose the function K_{UB} where:

$$K_{UB}(b_E, b_E') = (U - L)N_b \max_{s_E \in S_E \wedge b_E(s_E) > 0} |P(s_E; b_E) - P(s_E; b_E')| \quad (17)$$

where N_b is the number of particles in b_E , which is shown to satisfy (10) and given $\Upsilon = \{((s_A^i, b_E^i), y_i) \mid i \in I\}$, the upper bound can be computed by solving a linear program (LP) as demonstrated by the following lemma.

Lemma 5 (LP for upper bound). *The function K_{UB} from (17) satisfies (10), and for particle-based belief (s_A, b_E) represented by $\{(s_E^i, w_i)\}_{i=1}^{N_b}$, we have that $V_{UB}^\Upsilon(s_A, b_E)$ is the optimal value of the LP:*

$$\begin{aligned} & \text{minimize: } \sum_{k \in I_{s_A}} \lambda_k y_k + (U - L)N_b c \\ & \text{subject to: } c \geq |w_i - \sum_{k \in I_{s_A}} \lambda_k P(s_E^i; b_E^k)| \text{ for } 1 \leq i \leq N_b \\ & \lambda_k \geq 0 \text{ for } k \in I_{s_A} \text{ and } \sum_{k \in I_{s_A}} \lambda_k = 1. \end{aligned}$$

Since all quantities of interest in Algorithm 3 are computed exactly, the convergence guarantee in Theorem 3 holds for any initial particle-based belief.

Region-based beliefs. Particle filter approaches [35] are required to approximate the updated belief of particle-based representations if the current belief has zero weight at the true state due to partial observations and random perturbations. However, for NS-POMDPs the usual sampling importance re-sampling (SIR) approach [38] requires many particles, which can be computationally expensive. Therefore, we propose a new belief representation using *regions* of the continuous state space and show that it performs well empirically in handling the uncertainties. For any connected subset (region) $\phi_E \subseteq S_E$, let $\text{vol}(\phi_E) = \int_{s_E \in \phi_E} ds_E$.

Definition 8 (Region-based belief). *A belief $(s_A, b_E) \in S_B$ is represented by a weighted region set $\{(\phi_E^i, w_i)\}_{i=1}^{N_b}$ if $b_E(s_E) = \sum_{i=1}^{N_b} \chi_{\phi_E^i}(s_E) w_i$ where $w_i > 0$, ϕ_E^i is a region of $S_E^{s_A}$ and $\chi_{\phi_E^i} : S_E \rightarrow \mathbb{R}$ is such that $\chi_{\phi_E^i}(s_E) = 1$ if $s_E \in \phi_E^i$ and 0 otherwise for $1 \leq i \leq N_b$, and $\sum_{i=1}^{N_b} w_i \text{vol}(\phi_E^i) = 1$.*

In the case of region-based beliefs, finite-state POMDPs are not applicable even when approximating by finding all reachable states up to some finite depth, as from an initial (region-based) belief this would yield infinitely many reachable states. Region-based beliefs assume a uniform distribution over each region and allow the regions to overlap. Ensuring that belief updates of region-based beliefs result in region-based beliefs is difficult [39], as even simple transitions of variables with simple distributions can lead to complex distributions. Assumption 1 only ensures a finite partitioning of the state space for the transitions, but not that the updated belief places a uniform distribution over each region. We now provide conditions on the deterministic continuous components δ_E^i , see Assumption 1, of the environment transition function, under which region-based beliefs are closed.

Lemma 6 (Region-based belief closure). *If $\delta_E^i(\cdot, a) : S_E \rightarrow \delta_E^i(S_E, a)$ is piecewise differentiable and invertible and the Jacobian determinant of the inverse function is PWC for any $a \in \text{Act}$ and $1 \leq i \leq N_e$, then region-based beliefs are closed under belief updates.*

We next present an implementation of NS-HSVI using region-based beliefs for environment transition functions satisfying Lemma 6. For a region-based belief (s_A, b_E) , Algorithm 4 computes the belief update as the image of each

Algorithm 4 Region-based belief update

Input: (s_A, b_E) represented by $\{(\phi_E^i, w_i)\}_{i=1}^{N_b}$, action a , observation $s'_A = (loc', per')$

- 1: **if** $\delta_A(s_A, a)(loc') > 0$ **then**
- 2: $\mathcal{P} \leftarrow \emptyset$
- 3: **for** $i = 1, \dots, N_b, j = 1, \dots, N_e$ **do**
- 4: $\phi'_E \leftarrow \{\delta_E^j(s_E, a) \mid s_E \in \phi_E^i\}$ \triangleright Image
- 5: $\Phi_{\text{image}} \leftarrow$ divide ϕ'_E into regions over S_E by $obs_A(loc', \cdot)$
- 6: $w' \leftarrow (\text{vol}(\phi_E^i)/\text{vol}(\phi'_E))w_i\mu_j$ \triangleright Weight update
- 7: $\mathcal{P} \leftarrow \mathcal{P} \cup \{(\phi_E, w') \mid \phi_E \in \Phi_{\text{image}} \wedge \phi_E \subseteq S_E^{s'_A}\}$
- 8: Normalise the weights in \mathcal{P}
- 9: $b'_E(s'_E) \leftarrow \sum_{(\phi_E, w) \in \mathcal{P}} \chi_{\phi_E}(s'_E)w$ for all s'_E
- 10: **else** $b'_E(s'_E) \leftarrow 0$ for all s'_E
- 11: **return:** (s'_A, b'_E)

region, dividing the images by perception functions into regions of S_E , updating weights and selecting the regions with desired observations. The region-based belief update and expected values are summarised in Lemma 7.

Lemma 7 (Region-based belief update). *For region-based belief (s_A, b_E) represented by $\{(\phi_E^i, w_i)\}_{i=1}^{N_b}$, action a and observation s'_A : (s'_A, b'_E) returned by Algorithm 4 is region-based and $b'_E = b_E^{s_A, a, s'_A}$. Furthermore, if $h : S \rightarrow \mathbb{R}$ is PWC and Φ_E is a constant-FCP of S_E for h at s_A , then $\langle h, (s_A, b_E) \rangle = \sum_{i=1}^{N_b} \sum_{\phi_E \in \Phi_E} h(s_A, s_E)w_i \text{vol}(\phi_E^i \cap \phi_E)$ where $s_E \in \phi_E$.*

For the upper bound V_{UB}^Υ , the function K_{UB} has to compare beliefs over regions. We let $K_{UB} = K$, and thus (10) holds. Instead of a computationally expensive exact bound, which involves a large number of region intersections, Algorithm 5 is approximate, based on maximum densities, and involves solving an LP.

Lemma 8 (Region-based upper bound). *For region-based belief (s_A, b_E) represented by $\{(\phi_E^i, w_i)\}_{i=1}^{N_b}$ and $\Upsilon = \{((s_A^k, b_E^k), y_k) \mid k \in I\}$, if $K_{UB} = K$, (ϕ_E^{\max}, p) is returned by Algorithm 5, $b'_E = \sum_{k \in I_{s_A}} \lambda_k^* b_E^k$ and $\phi_E^{\max} \subseteq S_E^{b_E > b'_E}$ where λ_k^* is a solution to the LP of Algorithm 5, then p is an upper bound of V_{UB}^Υ at (s_A, b_E) . Furthermore, if $N_b = 1$, then $p = V_{UB}^\Upsilon(s_A, b_E)$.*

Algorithm 5 Approximate region-based upper bound via maximum density

Input: (s_A, b_E) represented by $\{(\phi_E^i, w_i)\}_{i=1}^{N_b}$, $\Upsilon = \{((s_A^k, b_E^k), y_k) \mid k \in I\}$

- 1: $I_b \leftarrow \operatorname{argmax}_{I_b \subseteq \{1, \dots, N_b\}} \sum_{i \in I_b} w_i$ subject to: $\cap_{i \in I_b} \phi_E^i \neq \emptyset$
- 2: $\phi_E^{\max} \leftarrow \cap_{i \in I_b} \phi_E^i$ \triangleright Maximum density
- 3: $p = \operatorname{minimize} \sum_{k \in I_{s_A}} \lambda_k y_k + (U - L)c$
- 4: subject to: $c \geq 1 - \sum_{k \in I_{s_A}} \sum_{j=1}^{N_b^k} \lambda_k w_{kj} \operatorname{vol}(\phi_E^{kj} \cap \phi_E^{\max})$,
 $\lambda_k \geq 0, \sum_{k \in I_{s_A}} \lambda_k = 1$
- 5: **return:** (ϕ_E^{\max}, p)

6. Implementation and Experimental Evaluation

In this section, we present a prototype implementation and experimental evaluation of our NS-HSVI algorithm for solution and optimal strategy synthesis on NS-POMDPs. We first summarise the details of the experimental setup, then discuss the results of two case studies, and conclude the section by discussing performance comparison.

6.1. Implementation Overview

We have developed a prototype Python implementation using the Parma Polyhedra Library [40] to build and operate over perception FCP representations of preimages of NNs, α -functions and reward structures. We recall that both α -functions and reward functions are piecewise constant over the continuous environment. They can thus be represented by subdividing the entire environment into *regions*, namely polyhedra over the continuous variables to which we associate a value. We remark that, since our method crucially depends on the states in a given region, and those in the subregions arising from subsequent refinements, being mapped to the same percept, arbitrary discretisation is not applicable. We use h -representations, which describe polyhedra through linear constraints for intersecting finite half-spaces. Upper bound computation is performed by solving LPs with Gurobi [41]. To sample points with polyhedra, we use the SMT solver Z3 [42].

We use the method of [30] to compute the (exact) preimage of piecewise linear NNs, which iterates backwards through the layers. This method is only applicable when the NN has piecewise linear decision boundaries, for which the basic building blocks are polytopes. This includes NNs with ReLU or linear layers, but can also be applied to approximations of NNs obtained via, for example, linear relaxation. With this pre-image, we then construct

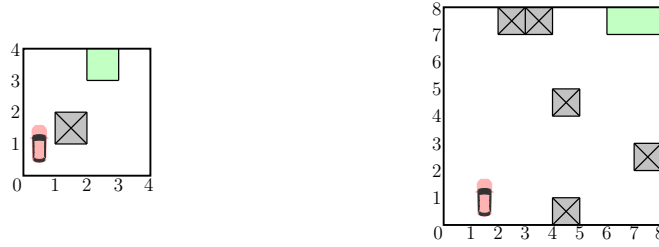


Figure 3: Car parking with obstacles.

a polyhedral representation of the environment space corresponding to the perception FCP. Regarding boundary points, we order regions and then assign boundary points to the region with the highest order, resolving ties via a measurable rule.

6.2. Car Parking Case Study

The first case study is the dynamic vehicle parking problem from Example 1, which we extend both with obstacles and to a larger environment. We were able to compute optimal strategies that lead the vehicle to the parking spot while avoiding obstacles (if present).

4×4 environment. To extend this example to the case when there is an obstacle region $\mathcal{R}_O = \{(x, y) \in \mathbb{R}^2 \mid 1 \leq x, y \leq 2\}$, see Fig. 3 (left), the state reward function changes such that for any $(s_A, (x, y)) \in S$: $r_S(s_A, (x, y)) = 1000$ if $(x, y) \in \mathcal{R}_P$, -1000 if $(x, y) \in \mathcal{R}_O$ and 0 otherwise, i.e., there is a negative reward if the vehicle hits the obstacle. The accuracy ε is 10^{-3} .

Strategy synthesis (4×4). Fig. 4 presents paths (π_1 , π_2 and π_3) for synthesised strategies starting from three particles in a given initial belief in two different scenarios, as well as the corresponding lower bound values for different regions of the environment. It also shows (on the right) the lower and upper bound values computed for the initial belief at each iteration. In both cases, there is an obstacle highlighted with black border. We consider strategies for when the reward associated to a collision is defined as in the reward structure in the model’s description, i.e., $(r_S(s_A, (x, y))) = -1000$ if $(x, y) \in \mathcal{R}_O$ (Fig. 4 top), and when that penalty is increased to -5000 (Fig. 4 bottom). We assume a uniform distribution over the points in the initial belief. We see that, when the negative reward of a collision with the obstacle is increased, Fig. 4 (bottom), all the generated paths avoid the cell with the obstacle. We also see that, in the first step, the action chosen is to move *left*; while this is possible for path π_0 (red), taking that action from the other two

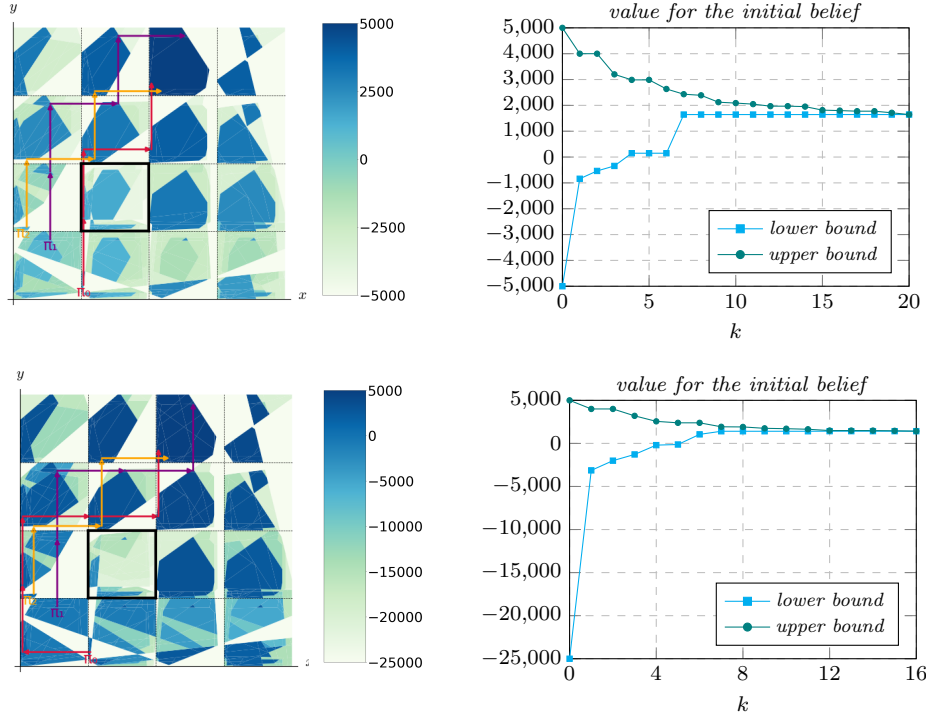


Figure 4: Paths and values for car parking (obstacle indicated with black border, $\beta = 0.8$, collision rewards equal to -1000 (top) and -5000 (bottom)).

initial belief points would take the agent out of the environment, in which case the agent would not move. For the scenario with the original reward structure, Fig. 4 (top), since the negative reward associated with a collision with the obstacle is lower, we see that such a reward can be compensated for by the agent afterwards, i.e., it can choose to move upwards from all points in the initial belief, resulting in a possibly unsafe strategy where a collision could happen.

Similarly, Fig. 5 shows values and strategies computed for the same scenario when considering a region-based belief. The regions reached from the initial position until arriving at the parking spot are indicated in orange, with the current state labelled by \mathbf{x} . The lower and upper bound values at each iteration are shown on the right-hand side, and the convergence demonstrates that the approximate upper bound for the region-based beliefs is tight if the belief has a unique region (see Lemma 8). We notice that the synthesised strategy avoids the obstacle while also reaching the parking spot with the least number of possible steps, maximising the agent’s reward.

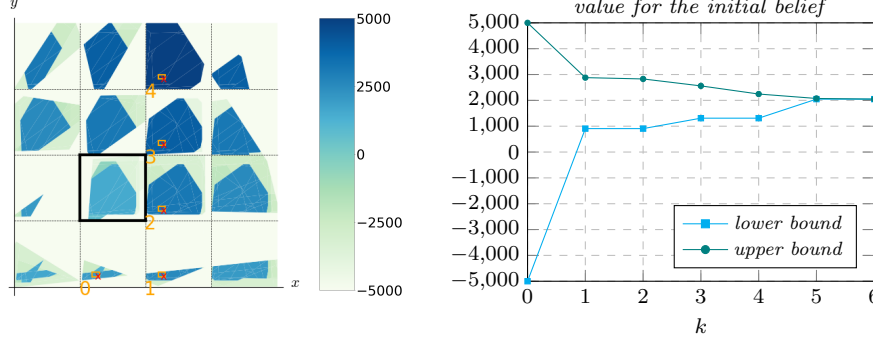


Figure 5: Region-based paths and values for car parking with the obstacle, $\beta = 0.8$.

Fig. 6 illustrates how computation progresses for Algorithm 3. Initially, we have an α -function for each local state whose underlying structure is the same as the perception FCP (see Fig. 1 right), with all regions initialised with the lower bound as described in Section 5.1. With each iteration, we refine the representation for the regions containing visited points and update their values. The figure shows the initial representation (left) and the maximum (over all local states) of the first 5, 25, and finally all the generated α -functions, coinciding then with the values presented in Fig. 4 (bottom). We can see how the values for the regions progressively increase as the computation proceeds (top row, left to right), as well as how the subsequent representations are refinements of the initial FCP (bottom row).

8x8 environment. We consider a larger 8x8 environment $\mathcal{R} = \{(x, y) \in \mathbb{R}^2 \mid 0 \leq x, y \leq 8\}$ with 4 obstacles \mathcal{R}_O (Fig. 3, right). In this model the parking spot is given by $\mathcal{R}_P = \{(x, y) \in \mathbb{R}^2 \mid 6 \leq x \leq 8 \wedge 7 \leq y \leq 8\}$, and the same reward structure is considered. To extend the NS-POMDP from Example 1 to this setting, the following changes to the components S_A , S_E , Δ_A and obs_A need to be made:

- $S_A = Loc \times Per$ with 5 trust levels $Loc = \{1, \dots, 5\}$ and 64 abstract grid points $Per = \{1, \dots, 64\}$ (percepts), which are ordered in the same way as Table 1;
- $S_E = \mathcal{R} = \{(x, y) \in \mathbb{R}^2 \mid 0 \leq x, y \leq 8\}$;
- $\Delta_A(tr, per) = Act$ if $per \in \{63, 64\}$ and $\Delta_A(tr, per) = \{up, down, left, right\}$ otherwise for all $tr \in Loc$ and $per \in Per$;

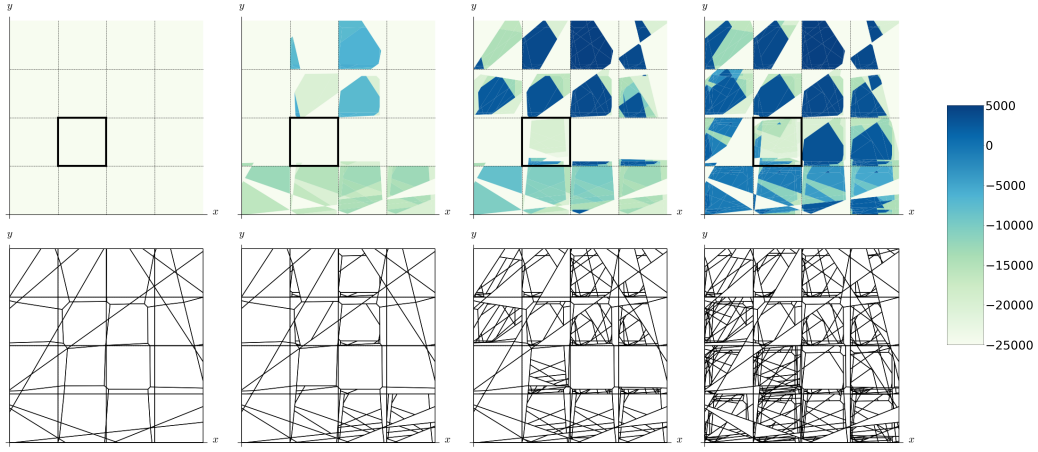


Figure 6: Values (top) and region outlines (bottom) for the initial and the maximum (over all local states) of the first 5, 25 and all the generated α -functions (respectively from left to right) for the 4×4 car parking example with obstacle, $\beta = 0.8$.

- $obs_A(tr, (x, y)) = \text{argmax}(f(x, y))$, where f , which is implemented via a feed-forward NN with one hidden ReLU layer with 15 neurons, takes the coordinate vector of the vehicle as input and then outputs one of the 64 abstract grid points.

Strategy synthesis (8×8). Fig. 8 (left) shows the perception FCP for the 8×8 environment. For this extended model, Fig. 7 (left) presents the paths from the three particles in the initial belief for the synthesised strategy, as well as lower bound values for the regions of the environment. As the figure demonstrates, the vehicle is able to reach the parking spot while avoiding the obstacles. As the full set of α -functions is large (see Table 4), to reduce computational effort we show approximate values obtained by maximizing over a set of sampled α -functions. Fig. 7 (right) shows how the lower and upper bound values for the initial belief change as the number of iterations of the NS-HSVI algorithm increases.

6.3. VCAS Case Study

In this case study there are two commercial aircraft: an ownship aircraft equipped with an NN-controlled vertical collision avoidance system (VCAS) and an intruder aircraft. Each second, the avoidance system gives a vertical climb-acceleration advisory ad to the pilot of the ownship to avoid near mid-air collisions (NMACs), which occur when the aircraft are separated by less than 100 ft vertically and 500 ft horizontally. The avoidance system extends

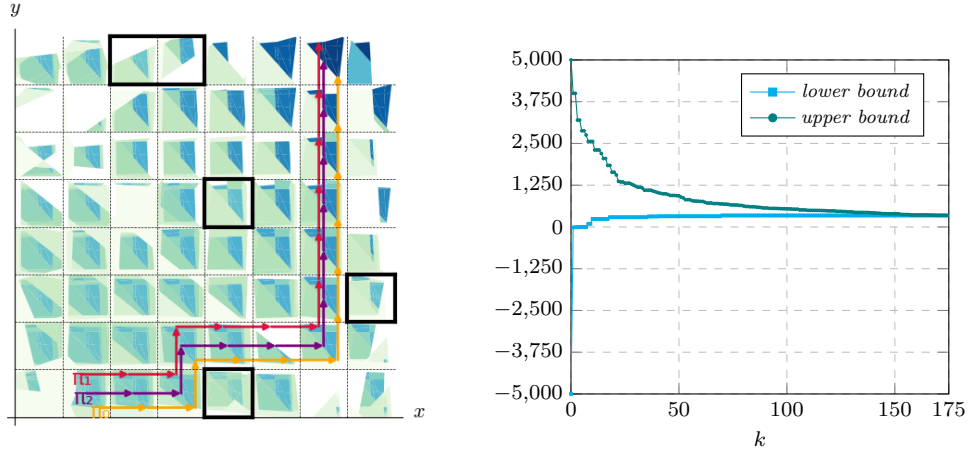


Figure 7: Paths and values for car parking (8×8 , $\beta = 0.8$, partially reconstructed).

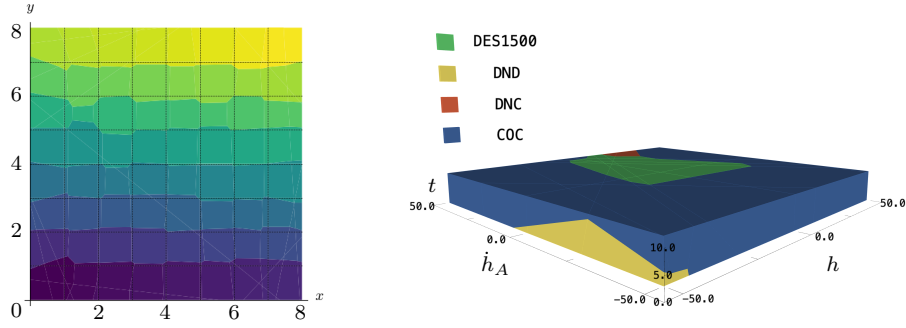


Figure 8: Perception FCP for car parking (8×8), and a slice of the perception FCP for the COC advisory of the VCAS (h scaled 10:1).

the classical VCAS [29], both by adding trust to measure uncertainty and by allowing for deviations from the advisories arising from the additional belief information. Regarding the intruder, unlike in the VCAS model of [29], we allow a non-zero constant climb-rate for the intruder. We were able to compute optimal strategies that safely guide the ownship by avoiding the collision zone.

VCAS as an NS-POMDP. The input to VCAS is a tuple (h, \dot{h}_A, t) , where h is the relative altitude of the two aircraft, \dot{h}_A the climb rate of ownship, and t the time until the loss of horizontal separation between the aircraft. VCAS is implemented via nine feed-forward NNs, each of which outputs the scores of nine possible advisories, see Table 2. Each advisory will provide a set of acceleration values and the ownship then either accelerates at one of these values or does not accelerate. Each NN of VCAS has one hidden ReLU layer

with 16 neurons, and therefore the regions in its pre-image are polytopes. If we had instead considered HorizontalCAS [43], the nonlinear environment transition function twists polytopes into non-polytopes, and would destroy our finite representations.

We model VCAS as an NS-POMDP in which the agent **Ag** is the ownship. The agent has four trust levels $\{1, \dots, 4\}$, which represent the trust it has in the previous advisory. These levels increase if the current advisory is compliant with the executed action, and decrease with probability 0.5 otherwise. A local state of the agent is of the form (ad_{pre}, tr) consisting of the previous advisory and the trust level and the percept of the agent is the current VCAS advisory. An environment state is a tuple (h, \dot{h}_A, t) corresponding to the input of VCAS. Formally, we have:

- $S_A = Loc \times Per$ with $Loc = \{1, \dots, 9\} \times \{1, \dots, 4\}$ and $Per = \{1, \dots, 9\}$;
- $S_E = [-2000, 2000] \times [-50, 50] \times [0, 20]$;
- $Act = \{0, \pm 3.0, \pm 7.33, \pm 8.33, \pm 9.33, \pm 9.7, \pm 10.7, \pm 11.7\}$;
- $\Delta_A(loc, per) = Act$ for all $loc \in Loc$ and $per \in Per$;
- $obs_A((ad_{pre}, tr), s_E) = \text{argmax}(f_{ad_{pre}}(s_E))$, where $f_{ad_{pre}}$ is the NN associated with the previous advisory ad_{pre} and the boundary point is resolved by assigning the advisory with the smallest label in Table 2;
- for $s_A = ((ad_{pre}, tr), ad) \in S_A$, $(ad', tr') \in Loc$ and $a \in Act$, if a is compliant with ad (see Table 2), then:

$$\delta_A(s_A, a)((tr', ad')) = \begin{cases} 1 & \text{if } (tr \leq 3) \wedge (tr' = tr + 1) \wedge (ad' = ad) \\ 1 & \text{if } (tr = 4) \wedge (tr' = tr) \wedge (ad' = ad) \\ 0 & \text{otherwise} \end{cases}$$

if a is not compliant with ad , then:

$$\delta_A(s_A, a)((tr', ad')) = \begin{cases} 0.5 & \text{if } (tr \geq 2) \wedge (tr' = tr - 1) \wedge (ad' = ad) \\ 0.5 & \text{if } (tr \geq 2) \wedge (tr' = tr) \wedge (ad' = ad) \\ 1 & \text{if } (tr = 1) \wedge (tr' = tr) \wedge (ad' = ad) \\ 0 & \text{otherwise;} \end{cases}$$

Label (ad_i)	Advisory	Description	Actions ft/s ²
1	COC	Clear of Conflict	-3, 0, +3
2	DNC	Do Not Climb	-9.33, -8.33, -7.33
3	DND	Do Not Descend	+7.33, +8.33, +9.33
4	DES1500	Descend at least 1500 ft/min	-9.33, -8.33, -7.33
5	CL1500	Climb at least 1500 ft/min	+7.33, +8.33, +9.33
6	SDES1500	Strengthen Descend to at least 1500 ft/min	-11.7, -10.7, -9.7
7	SCL1500	Strengthen Climb to at least 1500 ft/min	+9.7, +10.7, +11.7
8	SDES2500	Strengthen Descend to at least 2500 ft/min	-11.7, -10.7, -9.7
9	SCL2500	Strengthen Climb to at least 2500 ft/min	+9.7, +10.7, +11.7

Table 2: Available/suggested actions for each advisory of VCAS [4].

- for $s = (h, \dot{h}_A, t)$, $s' = (h', \dot{h}'_A, t') \in S$ if

$$\begin{aligned} h'' &= h - \Delta t(\dot{h}_A - \dot{h}_{\text{int}}) - 0.5\Delta t^2\ddot{h}_A \\ \dot{h}''_A &= \dot{h}_A + \ddot{h}_A\Delta t \\ t'' &= t - \Delta t \end{aligned}$$

then

$$\delta_E(s, a)(s') = \begin{cases} 1 & \text{if } (h'', \dot{h}''_A, t'') \in S_E \text{ and } s' = (h'', \dot{h}''_A, t'') \\ 1 & \text{if } (h'', \dot{h}''_A, t'') \notin S_E \text{ and } s' = s \\ 0 & \text{otherwise} \end{cases}$$

where $\Delta t = 1.0$ is the time step and the intruder is assumed to be a constant climb-rate $\dot{h}_{\text{int}} = 30$.

In the reward structure we consider, all action rewards are zero and the state reward function is such that for any $s \in S$: $r_S(s) = -1000$ if $t \in [0, 1] \wedge h \in [-100, 100]$ and 0 otherwise, i.e., there is a negative reward if altitudes of the aircraft are within 100 ft at time 0 or 1. The accuracy ε is 10^{-1} .

Strategy synthesis. To compute the perception FCP Φ_P , i.e., the pre-images of the NNs for this case study, we first trained these NNs. This involved computing an MDP table policy using local approximate value iteration, re-formatting this into training data and training the NNs [44]. To generate the pre-images, we adapted the method of [30], which was used to compute exact pre-images for the NNs of HorizontalCAS [43]. For example, the pre-image for the COC (Clear of Conflict) advisory is shown in Fig. 8 (right), which shows VCAS next issuing the advisory DES1500 (Descend at least 1500 ft/min) for the environment states in the green region to avoid an NMAC given the small values of h and t .

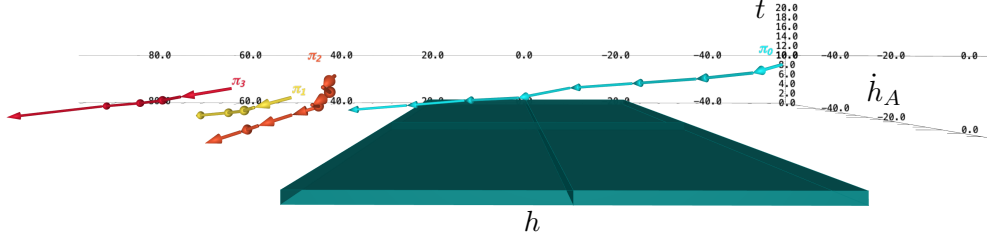


Figure 9: Paths from synthesised safe strategies for VCAS (h scaled 5:1).

Model	Belief type	Initial pts./regions	Discount factor	Pts./vol. updated	Iter.	Comput. time(s)
Car parking (no obstacles, 4×4)	Particle-based	3	0.8	205	15	32.7
		5	0.8	392	11	36.3
	Region-based	1	0.8	7.6	15	99.1
Car parking (w/ obstacle, 4×4)	Particle-based	3	0.8	210	17	46.2
		5	0.8	390	15	41.9
	Region-based	1	0.8	7.6	8	80.4
Car parking (w/ obstacles, 8×8)	Particle-based	3	0.8	960	174	1820
		5	0.8	1600	119	1337
	Region-based	1	0.8	43.2	59	2075
VCAS (3 actions)	Particle-based	4	0.75	441	40	228.2
		5	0.75	649	43	475.6
		6	0.75	476	23	1467
	Region-based	1	0.75	4725	10	994.6
VCAS (15 actions)	Particle-based	4	0.75	259	11	183.7
		5	0.75	425	15	357.7
		6	0.75	228	6	127.4
	Region-based	1	0.75	4059	7	2419

Table 3: Statistics for a set of NS-POMDP solution instances.

Fig. 9 shows the paths from the four particles in the initial belief of a synthesized strategy for the VCAS case study. For the particles that would reach the collision zone at time 0 or 1 (coloured green in Fig. 9), there is a course correction that enables the ownship to narrowly escape a collision.

6.4. Performance Analysis

To conclude the experimental analysis, we first discuss the performance of the implementation based on the statistics for two case studies, and then compare the performance of particle-based and region-based beliefs, and against SARSOP.

Experimental results. The experimental results reported in this section were generated on a 2.10GHz Intel Xeon Gold. Our NS-HSVI implementation is able to compute values and strategies for particle-based and region-based instances of the models we considered in less than 1 hour (Table 3). In the table, we report the model we consider, the belief type, the number of initial

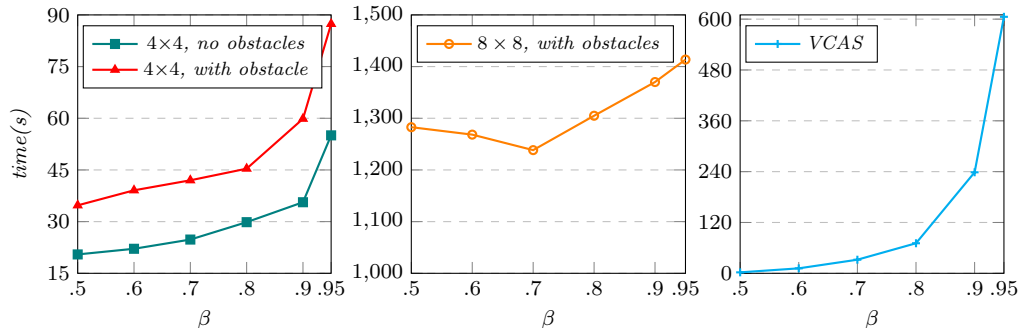


Figure 10: Solution times for different discount factors (for particle-based beliefs).

points or regions, the discount factor (β), the number of updated points or the volume of the updated regions (depending on the belief type), and the overall number of iterations of Algorithm 3 as well as the time taken until convergence. We found that the branching factor of the environment transition function, the number of agent states and actions, and the number of polyhedra in the perception FCP Φ_P can all have a significant impact on the computation time. Table 3 shows that computation for region-based beliefs normally takes longer because the number of regions of the perception FCP Φ_P over which the algorithm puts positive probabilities is usually larger, and thus it requires more ISPP backups. Moreover, while the update for particle-based beliefs only involves simple operations, updating region-based beliefs is far more complex due to the need of the polyhedra image computations, intersections and volume calculations.

Another crucial aspect is the choice of the discount factor (β). Fig. 10 shows how verification times vary for the different case studies as a function of that parameter. As expected, the trend we are able to observe is that it takes longer for the algorithm to converge as the value of β increases. The small drop in the curve for the 8×8 version of the car parking example for the lower values of β can be explained by the inherent nondeterminism of HSVI exploration, especially in the early stages of the computation when many regions may have the same lower and upper bounds. This may lead to the algorithm being indifferent with respect to the actions it takes, and thus constructing paths that have lower impact on the values of the initial belief. Finally, another element that impacts the running time is the choice of the initial belief and the model’s dynamics. This can be especially noticed when comparing the two instances of VCAS. The beliefs for the version with 15 actions have lower values for t and are thus much closer to the boundaries of

Model	Belief type #initial	Total regions (α -functions)	Lower bound	Upper bound	Strat. time (s)	Following ratio	Avg. trust
Car parking (no obstacles, 4×4)	PB, 3	80,494	2389.3309	2389.3333	19.3	88%	3.6
	PB, 5	42,224	2047.9989	2048.0000	14.0	100%	3.9
	RB, 1	36,467	2047.9992	2048.0000	50.0	100%	3.9
Car parking (w/ obstacle, 4×4)	PB, 3	99,513	2218.6653	2218.6666	24.5	78%	3.3
	PB, 5	47,719	2047.9990	2048.0000	14.2	100%	3.9
	RB, 1	35,751	2047.9988	2048.0000	39.4	100%	3.9
Car parking (w/ obstacles, 8×8)	PB, 3	1,410,799	343.5969	343.5974	338.9	85%	4.3
	PB, 5	547,753	343.5970	343.5974	158.4	97%	4.4
	RB, 1	550,685	343.5964	343.5974	473.8	80%	4.3
VCAS (3 actions)	PB, 4	154,009	-1.2281	0.0	75.3	-	-
	PB, 5	278,447	-1.2398	0.0	127.5	-	-
	PB, 6	868,257	-0.2498	0.0	400.8	-	-
	RB, 1	22,919	-0.0715	0.0	65.5	-	-
VCAS (15 actions)	PB, 4	32,387	-0.6718	0.0	18.7	33%	1.3
	PB, 5	30,003	-0.9874	0.0	21.7	0%	1.0
	PB, 6	19,218	-1.0789	0.0	13.0	33%	1.3
	RB, 1	21,102	-0.6133	0.0	49.9	0%	1.0

Table 4: Further statistics for a set of NS-POMDP solution instances.

the environment, which considerably limits the number of reachable states and makes it possible for the algorithm to converge more quickly despite the higher number of actions.

Table 4 shows, for a number of instances of both case studies and for each belief type, particle-based (PB) and region-based (RB): the total number of polyhedra that make up the α -functions computed, the lower and upper bounds on values for the initial belief and the time required for strategy synthesis, i.e., reading α -functions, finding maximum actions and updating beliefs. We also show the compliance ratio with respect to the suggested actions as well as average trust values over 20 paths generated from the synthesised strategies.

For the car parking case study (recall the accuracy is 10^{-3}), in general, the more iterations that are needed for convergence, the higher the number of α -functions generated and consequently the total number of regions. Strategy synthesis for region-based beliefs tends to be comparatively slower due to the complexity of the mathematical operations involved. The following ratio and average trust values are both high for this case study as the suggested actions in Table 1 are close to the optimal strategies.

Regarding VCAS, the statistics in Table 4 are for the accuracy of 10^{-1} . The α -functions generally have a large number of regions, as the perception FCP for each of the 9 NNs of VCAS has many regions, and hence many intersections. In addition, we note that, for this model, the following ratio and average trust values are low, and in fact have been omitted for the

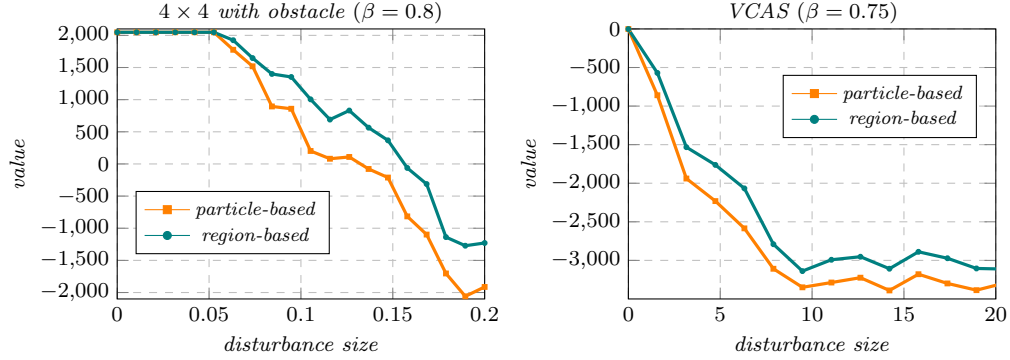


Figure 11: Comparison between particle-based and region-based values.

model with 3 actions. This is because (see Table 2) the number of suggested actions associated to each advisory is only a fraction of the 15 actions we considered and, for a given belief, there are many strategies that can lead to the optimal value. Recall also that it is assumed that the intruder aircraft is always climbing and the beliefs we considered were all reasonably close to the collision zone. We analysed the synthesised strategies and found that, in many cases, the agent chose actions that would at first lead to a faster descent than those suggested in Table 2, but then compensated by descending less, or not at all, at later stages. While the values of the actions differed, all strategies we observed led to the ownship lowering its altitude, which would lead to an increase of the overall height difference so as to escape a potential collision. Thus, the low following ratios do not reflect an inadequacy of the advisories.

Performance comparison. Finally, we compare values obtained for particle-based and region-based initial beliefs where the initial region covers the particles, after they have been disturbed by shifting their position along a sampled direction. This models a realistic scenario, in which the actual initial belief differs from the initial belief used to compute offline lower and upper bound functions, for example due to measurement imprecision. For a range of disturbance sizes (the distances by which the particles are shifted), the lower bound values for the average of 100 sampled points are presented in Fig. 11. The results show that, in all cases, the region-based belief values are greater than or equal to the particle-based values, and therefore the region-based approach is more robust to disturbance (i.e., generates lower bound values closer to the optimum).

As the number of reachable states for a given number of transitions from

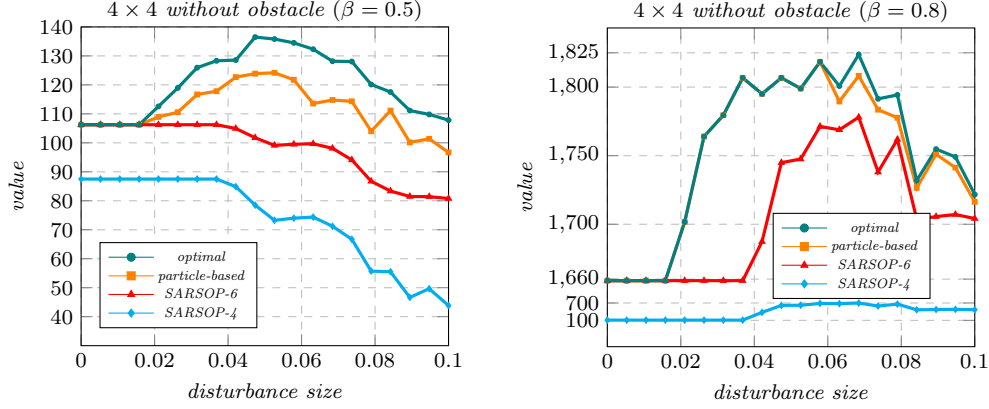


Figure 12: Comparison between particle-based and SARSOP values.

an initial particle-based belief is finite, we also compare the robustness of values obtained with our particle-based NS-HSVI and the finite-state POMDP solver SARSOP, for the 4×4 dynamic vehicle parking without obstacles in Fig. 12. For an initial particle-based belief, we build two finite-state POMDPs by unrolling the model’s execution when considering 4 and 6 transitions, respectively. Note that no new distinct states can be reached for paths whose length exceeds 6 in this example, as any cell in the grid can be reached from any other cell in 6 steps. Then, we compute the value function, represented as a set of α -vectors, for each finite-state POMDP with SARSOP. Using the value function, we approximate the values of beliefs disturbed by shifting as above, in which each shifted particle takes the value of the closest point in the finite-state space of the unrolled POMDP. The optimal value of each shifted belief is computed by unrolling from the shifted belief for a maximum of 6 transitions and solving the resulting finite-state POMDP with SARSOP.

SARSOP performs better with respect to the computational time taken, which is understandable as SARSOP takes as input a discretised version of the model and does not operate over a continuous abstraction, as NS-HSVI does, requiring expensive operations over polyhedra. Nevertheless, the results shown in Fig. 12 demonstrate that the values achieved by strategies generated using SARSOP highly depend on how much of the model’s execution we are able to construct beforehand, as the impact of missing reward-critical states with a shorter horizon can be considerable. It also shows that particle-based NS-HSVI obtains greater or equal lower bound values compared to SARSOP within a small disturbance range. This is due to the fact that, when performing the ISPP backup, we update not only the values for the

visited points but also for the regions that contain them. The optimal values of the shifted beliefs indicate that the values of the particle-based NS-HSVI and SARSOP are both valid lower bounds.

7. Conclusions

We have introduced NS-POMDPs, the first partially observable neuro-symbolic model for an agent operating in continuous state space and perceiving the environment using NNs. Motivated by the need for safety guarantees for such systems, we focus on *optimal* policy synthesis with discounting. By placing mild assumptions on the structure of NS-POMDPs, we are able to exploit their structure to approximate the value function from below and above using a representation of PWC α -functions and belief-value induced functions. Using NS-HSVI, a variant of the classical HSVI algorithm, we synthesised optimal strategies for an agent parking a car and safe strategies for an agent using an aircraft collision avoidance system, employing the popular particle-based and novel region-based beliefs. Our main achievement is demonstrating the practicality of the methodology for small systems with realistic neural network components. To make progress in this challenging problem domain, similarly to other POMDP approaches, we initially focus on discounted objectives, and aim to later extend to the more complex undiscounted case (which is already undecidable for finite-state POMDPs). However, as the case studies demonstrate, we can use our approach to synthesise strategies that can then be shown to be safe in terms of provably avoiding “unsafe” parts of the state space. Further work includes efficiency improvement by incorporating sampling, adapting NS-HSVI to more general perception NNs and extending the approach to multi-agent systems.

Acknowledgements. This project was funded by the ERC under the European Union’s Horizon 2020 research and innovation programme (FUN2MODEL, grant agreement No.834115).

Appendix A. Proofs from Section 4

Before we give the proofs of Section 4 we require the following definition.

Definition 9. For FCPs Φ_1 and Φ_2 of S , we denote by $\Phi_1 + \Phi_2$ the smallest FCP of S such that $\Phi_1 + \Phi_2$ is a refinement of both Φ_1 and Φ_2 , which can be computed by all combinations of intersections between regions in Φ_1 and Φ_2 .

Lemma 1 (Perception FCP). *There exists a smallest FCP of S , called the perception FCP, denoted Φ_P , such that all states in any $\phi \in \Phi_P$ are observationally equivalent, i.e., if $(s_A, s_E), (s'_A, s'_E) \in \phi$, then $s_A = s'_A$ and we let $s_A^\phi = s_A$.*

Proof. Since obs_A is PWC and S_A is finite, using Definition 1 we have that for any $s_A = (loc, per) \in S_A$ the set $S_E^{s_A} = \{s_E \in S_E \mid obs_A(loc, s_E) = per\}$ can be expressed as a number of disjoint regions of S_E and we let $\Phi_E^{s_A}$ be such a representation that minimises the number of such regions. It then follows that $\{\{(s_A, s_E) \mid s_E \in \phi_E\} \mid \phi_E \in \Phi_E^{s_A} \wedge s_A \in S_A\}$ is a smallest FCP of S such that all states in any region are observationally equivalent. \square

Theorem 1 (P-PWLC closure and convergence). *If $V \in \mathbb{F}(S_B)$ and P -PWLC, then so is $[TV]$. If $V^0 \in \mathbb{F}(S_B)$ and P -PWLC, then the sequence $(V^t)_{t=0}^\infty$, such that $V^{t+1} = [TV^t]$ are P -PWLC and converges to V^* .*

Proof. Consider any $V \in \mathbb{F}(S_B)$ that is P -PWLC, by Definition 6 there exists a finite set $\Gamma \subseteq \mathbb{F}_C(S)$ such that:

$$V(s_A, b_E) = \max_{\alpha \in \Gamma} \langle \alpha, (s_A, b_E) \rangle \text{ for all } (s_A, b_E) \in S_B. \quad (\text{A.1})$$

Now consider any $(s_A, b_E), (s'_A, b'_E) \in S_B$ where $s'_A = (loc', per')$ and action $a \in \Delta_A(s_A)$, and letting $P_1 := P(s'_A \mid (s_A, b_E), a)$, by (A.1) we have:

$$\begin{aligned} V(s'_A, b_E^{s_A, a, s'_A}) &= \max_{\alpha \in \Gamma} \langle \alpha, (s'_A, b_E^{s_A, a, s'_A}) \rangle \\ &= \max_{\alpha \in \Gamma} \int_{s'_E \in S_E} \alpha(s'_A, s'_E) b_E^{s_A, a, s'_A}(s'_E) ds'_E && \text{by (5)} \\ &= \max_{\alpha \in \Gamma} \int_{s'_E \in S_E} \alpha(s'_A, s'_E) \frac{P((s'_A, s'_E) \mid (s_A, b_E), a)}{P(s'_A \mid (s_A, b_E), a)} ds'_E && \text{by (1)} \\ &= \max_{\alpha \in \Gamma} \int_{s'_E \in S_E} \alpha(s'_A, s'_E) \frac{P((s'_A, s'_E) \mid (s_A, b_E), a)}{P_1} ds'_E && \text{by definition of } P_1 \\ &= \frac{1}{P_1} \max_{\alpha \in \Gamma} \int_{s'_E \in S_E} \alpha(s'_A, s'_E) P((s'_A, s'_E) \mid (s_A, b_E), a) ds'_E && \text{rearranging} \\ &= \frac{1}{P_1} \max_{\alpha \in \Gamma} \int_{s'_E \in S_E} \alpha(s'_A, s'_E) \delta_A(s_A, a)(loc') \left(\int_{s'_E \in S_E^{s_A} \wedge s_E \in S_E} b_E(s_E) \delta_E(s_E, a)(s'_E) ds_E \right) ds'_E \\ &&& \text{by (3)} \end{aligned}$$

$$\begin{aligned}
&= \frac{1}{P_1} \max_{\alpha \in \Gamma} \int_{s_E \in E} \left(\delta_A(s_A, a)(loc') \int_{s'_E \in S_E^{s'_A}} \alpha(s'_A, s'_E) \delta_E(s_E, a)(s'_E) ds'_E \right) b_E(s_E) ds_E \\
&\hspace{25em} \text{rearranging.} \\
&\hspace{25em} (A.2)
\end{aligned}$$

Next, for any $\alpha \in \mathbb{F}_C(S)$, $s'_A \in S_A$ and $a \in Act$ we let $\alpha^{a, s'_A} : S \rightarrow \mathbb{R}$ be the function where for any $(s_A, s_E) \in S$ if $a \in \Delta_A(s_A)$, then:

$$\begin{aligned}
\alpha^{a, s'_A}(s_A, s_E) &= \delta_A(s_A, a)(loc') \left(\int_{s'_E \in S_E^{s'_A}} \alpha(s'_A, s'_E) \delta_E(s_E, a)(s'_E) ds'_E \right) \\
&= \delta_A(s_A, a)(loc') \left(\sum_{s'_E \in \Theta_{s_E}^a \cap S_E^{s'_A}} \alpha(s'_A, s'_E) \delta_E(s_E, a)(s'_E) \right) \quad \text{rearranging} \\
&\hspace{25em} (A.3)
\end{aligned}$$

and otherwise $\alpha^{a, s'_A}(s_A, s_E) = L$. Now combining (A.2) and (A.3) we have:

$$V(s'_A, b_E^{s_A, a, s'_A}) = \frac{1}{P(s'_A \mid (s_A, b_E), a)} \max_{\alpha \in \Gamma} \int_{s_E \in E} \alpha^{a, s'_A}(s_A, s_E) b_E(s_E) ds_E. \quad (A.4)$$

We next prove that α^{a, s'_A} is PWC, i.e., $\alpha^{a, s'_A} \in \mathbb{F}_C(S)$. Since $\alpha \in \mathbb{F}_C(S)$ there exists an FCP Φ of S such that α is constant in each region of Φ . According to Assumption 1, there exists a preimage FCP Φ' of $\Phi + \Phi_P$ for action a , where Φ_P is the perception FCP from Lemma 1. Consider any region $\phi' \in \Phi'$ and let ϕ be any region of $\Phi + \Phi_P$ such that $\Theta_s^a \cap \phi \neq \emptyset$ for all $s \in \phi'$. Since Φ_P is the perception FCP, there exists $s'_A \in S_A$ such that if $s' \in \phi$, then $s' = (s'_A, s'_E)$ for some $s'_E \in S_E$ and let $\phi_E = \{s_E \in S_E \mid (s'_A, s_E) \in \phi\}$. If $s, \tilde{s} \in \phi'$ such that $s = (s_A, s_E)$ and $\tilde{s} = (\tilde{s}_A, \tilde{s}_E)$, then using Assumption 1 we have $\sum_{s' \in \Theta_s^a \cap \phi} \delta(s, a)(s') = \sum_{\tilde{s}' \in \Theta_{\tilde{s}}^a \cap \phi} \delta(\tilde{s}, a)(\tilde{s}')$ and $s_A = \tilde{s}_A$. Now combining this fact with Definition 2, it follows that:

$$\begin{aligned}
&\delta_A(s_A, a)(loc') \left(\sum_{s'_E \in \Theta_{s_E}^a \cap \phi_E} \delta_E(s_E, a)(s'_E) \right) \\
&= \delta_A(\tilde{s}_A, a)(loc') \left(\sum_{\tilde{s}'_E \in \Theta_{\tilde{s}_E}^a \cap \phi_E} \delta_E(\tilde{s}_E, a)(\tilde{s}'_E) \right).
\end{aligned}$$

Furthermore, since $\alpha(s'_A, s'_E) = \alpha(s'_A, \tilde{s}'_E)$ for any $(s'_A, s'_E), (s'_A, \tilde{s}'_E) \in \phi$ and $S_E^{s'_A} = \{s_E \in S_E \mid obs_A(loc', s_E) = per'\}$ is equal to $\{\phi_E \mid \phi \in \Phi^{s'_A}\}$ for some finite set of regions $\Phi^{s'_A} \subseteq \Phi + \Phi_P$, it follows that $\alpha^{a, s'_A}(s) = \alpha^{a, s'_A}(\tilde{s})$, implying that α^{a, s'_A} is constant in each region of Φ' .

Now substituting (A.4) into Definition 4 it follows that $[TV](s_A, b_E)$ equals:

$$\begin{aligned} & \max_{a \in \Delta_A(s_A)} \left\{ \langle R_a, (s_A, b_E) \rangle + \beta \sum_{s'_A \in S_A} \max_{\alpha \in \Gamma} \int_{s_E \in E} \alpha^{a, s'_A}(s_A, s_E) b_E(s_E) ds_E \right\} \\ &= \max_{a \in \Delta_A(s_A)} \left\{ \langle R_a, (s_A, b_E) \rangle + \beta \sum_{s'_A \in S_A} \max_{\alpha \in \Gamma} \langle \alpha^{a, s'_A}, (s_A, b_E) \rangle \right\} \quad \text{by (5).} \end{aligned}$$

Therefore letting $\Gamma^{a, s'_A} = \{\alpha^{a, s'_A} \mid \alpha \in \Gamma\}$ and

$$\alpha_{b_E}^{a, s'_A} \in \arg \max_{\alpha^{a, s'_A} \in \Gamma^{a, s'_A}} \langle \alpha^{a, s'_A}, (s_A, b_E) \rangle$$

where $\alpha_{b_E}^{a, s'_A}$ is independent of s_A due to (A.3), it then follows from Definition 4 that:

$$\begin{aligned} [TV](s_A, b_E) &= \max_{a \in \Delta_A(s_A)} \left\{ \langle R_a, (s_A, b_E) \rangle + \beta \sum_{s'_A \in S_A} \langle \alpha_{b_E}^{a, s'_A}, (s_A, b_E) \rangle \right\} \\ &= \max_{a \in \Delta_A(s_A)} \left\langle R_a + \beta \sum_{s'_A \in S_A} \alpha_{b_E}^{a, s'_A}, (s_A, b_E) \right\rangle \quad \text{by (5).} \quad (\text{A.5}) \end{aligned}$$

Furthermore, we have that

$$\Gamma_{(s_A, b_E)} = \left\{ R_a + \beta \sum_{s'_A \in S_A} \alpha_{b_E}^{a, s'_A} \mid a \in \Delta_A(s_A) \right\}$$

and from (A.5):

$$[TV](s_A, b_E) = \max_{\alpha \in \Gamma_{(s_A, b_E)}} \langle \alpha, (s_A, b_E) \rangle.$$

Finally, since S_A , Act and Γ are finite, R_a is PWC by Assumption 2 and α^{a, s'_A} is PWC, defining Γ' to be the set containing the PWC functions:

$$R_a + \beta \sum_{s'_A \in S_A} \alpha^{a, s'_A} \in \mathbb{F}_C(S)$$

for all $a \in Act$, $s'_A \in S_A$ and $\alpha^{a, s'_A} \in \Gamma^{a, s'_A}$, we have for any $(s_A, b_E) \in S_B$:

$$[TV](s_A, b_E) = \max_{\alpha \in \Gamma'} \langle \alpha, (s_A, b_E) \rangle.$$

Therefore, $[TV]$ is P-PWLC. As can be seen $|\Gamma'| = |Act||\Gamma|^{|S_A|}$, and hence the number of α -functions representing $[TV]$ given grows exponentially in the number of agent states for those representing V .

The remainder of the proof follows from Banach's fixed point theorem and the fact we have proved that if $V \in \mathbb{F}(S_B)$ and P-PWLC, so is $[TV]$. \square

Theorem 2 (Convexity and continuity). *For any $s_A \in S_A$, the value function $V^*(s_A, \cdot) : \mathbb{P}(S_E) \rightarrow \mathbb{R}$ is convex and for any $b_E, b'_E \in \mathbb{P}(S_E)$:*

$$|V^*(s_A, b_E) - V^*(s_A, b'_E)| \leq K(b_E, b'_E) \quad (\text{A.6})$$

where $K(b_E, b'_E) = (U - L) \int_{s_E \in S_E^{b_E > b'_E}} (b_E(s_E) - b'_E(s_E)) ds_E$ and $S_E^{b_E > b'_E} = \{s_E \in S_E^{s_A} \mid b_E(s_E) - b'_E(s_E) > 0\}$.

Proof. According to Theorem 1 there exists a (possibly infinite) set $\Gamma \subseteq \mathbb{F}_C(S)$ such that for any $(s_A, b_E) \in S_B$:

$$V^*(s_A, b_E) = \sup_{\alpha \in \Gamma} \langle \alpha, (s_A, b_E) \rangle. \quad (\text{A.7})$$

Given $s_A \in S_A$, consider any $b_E, b'_E \in \mathbb{P}(S_E)$ and $\lambda \in [0, 1]$, and we have:

$$\begin{aligned} & \lambda V^*(s_A, b_E) + (1 - \lambda) V^*(s_A, b'_E) \\ &= \lambda \sup_{\alpha \in \Gamma} \langle \alpha, (s_A, b_E) \rangle + (1 - \lambda) \sup_{\alpha \in \Gamma} \langle \alpha, (s_A, b'_E) \rangle && \text{by (A.7)} \\ &= \sup_{\alpha \in \Gamma} \langle \alpha, (s_A, \lambda b_E) \rangle + \sup_{\alpha \in \Gamma} \langle \alpha, (s_A, (1 - \lambda) b'_E) \rangle && \text{by (5)} \\ &\geq \sup_{\alpha \in \Gamma} \langle \alpha, (s_A, \lambda b_E + (1 - \lambda) b'_E) \rangle && \text{rearranging} \\ &= V^*(s_A, \lambda b_E + (1 - \lambda) b'_E) && \text{by (A.7)} \end{aligned}$$

which proves that $V^*(s_A, \cdot)$ is convex.

Next given α and s_A , let $V_{\alpha, s_A}(b_E) := \langle \alpha, (s_A, b_E) \rangle$ for $(s_A, b_E) \in S_B$. For any $(s_A, b_E), (s_A, b'_E) \in S_B$, without loss of generality, we can assume that $V_{\alpha, s_A}(b_E) \geq V_{\alpha, s_A}(b'_E)$, and therefore:

$$\begin{aligned} & |V_{\alpha, s_A}(b_E) - V_{\alpha, s_A}(b'_E)| = V_{\alpha, s_A}(b_E) - V_{\alpha, s_A}(b'_E) \\ &= \langle \alpha, (s_A, b_E) \rangle - \langle \alpha, (s_A, b'_E) \rangle && \text{by definition of } V_{\alpha, s_A} \\ &= \int_{s_E \in S_E^{s_A}} \alpha(s_A, s_E) b_E(s_E) ds_E - \int_{s_E \in S_E^{s_A}} \alpha(s_A, s_E) b'_E(s_E) ds_E && \text{by (5)} \\ &= \int_{s_E \in S_E^{s_A}} \alpha(s_A, s_E) (b_E(s_E) - b'_E(s_E)) ds_E && \text{rearranging.} \end{aligned} \quad (\text{A.8})$$

Since $b_E, b'_E \in \mathbb{P}(S_E)$ and $(s_A, b_E), (s_A, b'_E) \in S_B$, we have:

$$\int_{s_E \in S_E^{s_A}} b_E(s_E) ds_E = \int_{s_E \in S_E^{s_A}} b'_E(s_E) ds_E = 1. \quad (\text{A.9})$$

Now, letting $S_E^+ = \{s_E \in S_E^{s_A} \mid b_E(s_E) - b'_E(s_E) > 0\}$ and $S_E^- = \{s_E \in S_E^{s_A} \mid b_E(s_E) - b'_E(s_E) \leq 0\}$, rearranging (A.9) and using the fact that $S_E^+ \cup S_E^- = S_E^{s_A}$ it follows that:

$$\int_{s_E \in S_E^-} (b_E(s_E) - b'_E(s_E)) ds_E = - \int_{s_E \in S_E^+} (b_E(s_E) - b'_E(s_E)) ds_E. \quad (\text{A.10})$$

Next, using (A.8), the definition of V_{α, s_A} and (5), it follows that $|V_{\alpha, s_A}(b_E) - V_{\alpha, s_A}(b'_E)|$ equals:

$$\begin{aligned} & \int_{s_E \in S_E^+} \alpha(s_A, s_E) (b_E(s_E) - b'_E(s_E)) ds_E + \int_{s_E \in S_E^-} \alpha(s_A, s_E) (b_E(s_E) - b'_E(s_E)) ds_E \\ & \leq \int_{s_E \in S_E^+} U (b_E(s_E) - b'_E(s_E)) ds_E + \int_{s_E \in S_E^-} L (b_E(s_E) - b'_E(s_E)) ds_E \\ & \quad \text{by definition of } S_E^+, S_E^-, U \text{ and } L \\ & = U \int_{s_E \in S_E^+} (b_E(s_E) - b'_E(s_E)) ds_E - L \int_{s_E \in S_E^+} (b_E(s_E) - b'_E(s_E)) ds_E \\ & \quad \text{rearranging and using (A.10)} \\ & = k \int_{s_E \in S_E^+} (b_E(s_E) - b'_E(s_E)) ds_E \quad \text{rearranging and letting } k = U - L. \end{aligned} \quad (\text{A.11})$$

We can now derive the following upper bound for $V^*(s_A, b_E)$:

$$\begin{aligned} V^*(s_A, b_E) &= \sup_{\alpha \in \Gamma} \langle \alpha, (s_A, b_E) \rangle && \text{by (A.7)} \\ &= \sup_{\alpha \in \Gamma} V_{\alpha, s_A}(b_E) && \text{by definition of } V_{\alpha, s_A} \\ &= \sup_{\alpha \in \Gamma} (V_{\alpha, s_A}(b'_E) + V_{\alpha, s_A}(b_E) - V_{\alpha, s_A}(b'_E)) && \text{rearranging} \\ &= \sup_{\alpha \in \Gamma} (V_{\alpha, s_A}(b'_E) + |V_{\alpha, s_A}(b_E) - V_{\alpha, s_A}(b'_E)|) && \text{rearranging} \\ &\leq \sup_{\alpha \in \Gamma} \left\{ V_{\alpha, s_A}(b'_E) + k \int_{s_E \in S_E^+} (b_E(s_E) - b'_E(s_E)) ds_E \right\} && \text{by (A.11)} \\ &= \sup_{\alpha \in \Gamma} (V_{\alpha, s_A}(b'_E)) + k \int_{s_E \in S_E^+} (b_E(s_E) - b'_E(s_E)) ds_E && \text{rearranging} \\ &= \sup_{\alpha \in \Gamma} \langle \alpha, (s_A, b'_E) \rangle + k \int_{s_E \in S_E^+} (b_E(s_E) - b'_E(s_E)) ds_E && \text{by definition of } V_{\alpha, s_A} \end{aligned}$$

$$= V^*(s_A, b'_E) + k \int_{s_E \in S_E^+} (b_E(s_E) - b'_E(s_E)) ds_E \quad \text{by (A.7).}$$

Using similar steps we can also show:

$$V^*(s_A, b'_E) \leq V^*(s_A, b_E) + k \int_{s_E \in S_E^+} (b_E(s_E) - b'_E(s_E)) ds_E$$

and therefore the second part of the theorem follows. \square

Appendix B. Proofs from Section 5

Lemma 2 (Lower bound). *At belief $(s_A, b_E) \in S_B$, the function α^* generated by Algorithm 1 is a PWC α -function satisfying (11), $V_{LB}^\Gamma \leq V_{LB}^{\Gamma'} \leq V^*$ and $V_{LB}^{\Gamma'}(s_A, b_E) \geq [TV_{LB}^\Gamma](s_A, b_E)$.*

Proof. By following the proof of Theorem 1 and how \bar{a} and $\alpha^{s'_A}$ are constructed for all $s'_A \in S_A$, we can easily verify that α^* in Algorithm 1 is a PWC α -functions that satisfies (11).

For any $V_1, V_2 \in \mathbb{F}(S_B)$, we use the shorthand $V_1 \leq V_2$ to denote that $V_1(\hat{s}_A, \hat{b}_E) \leq V_2(\hat{s}_A, \hat{b}_E)$ for all $(\hat{s}_A, \hat{b}_E) \in S_B$. Now, in the case of the lower bound consider any V_{LB}^Γ such that $V_{LB}^\Gamma \leq V^*$. Since $\Gamma' = \Gamma \cup \{\alpha^*\}$ after updating V_{LB}^Γ at (s_A, b_E) through Algorithm 1, for any $(\hat{s}_A, \hat{b}_E) \in S_B$:

$$\max_{\alpha \in \Gamma} \langle \alpha, (\hat{s}_A, \hat{b}_E) \rangle \leq \max_{\alpha \in \Gamma'} \langle \alpha, (\hat{s}_A, \hat{b}_E) \rangle.$$

Therefore combining this with the fact that V_{LB}^Γ is a P-PWLC function for the finite set Γ , see Definition 6, we have $V_{LB}^\Gamma(\hat{s}_A, \hat{b}_E) \leq V_{LB}^{\Gamma'}(\hat{s}_A, \hat{b}_E)$ and since (\hat{s}_A, \hat{b}_E) was arbitrary $V_{LB}^\Gamma \leq V_{LB}^{\Gamma'}$.

Next, again using the fact V_{LB}^Γ is a P-PWLC function for the finite set Γ we have:

$$\begin{aligned} V_{LB}^{\Gamma'}(s_A, b_E) &= \max_{\alpha \in \Gamma'} \langle \alpha, (s_A, b_E) \rangle \\ &\geq \langle \alpha^*, (s_A, b_E) \rangle && \text{rearranging} \\ &= [TV_{LB}^\Gamma](s_A, b_E) && \text{by (11).} \end{aligned}$$

In Algorithm 1, if the backup at line 6 is executed, then the Bellman operator is applied for some states in ϕ which may result in non-optimal Bellman

backup for the other states in ϕ , and if the backup at line 7 is executed, α^* is assigned the lower bound L in ϕ . Therefore we have for any $(\hat{s}_A, \hat{b}_E) \in S_B$:

$$\begin{aligned} \langle \alpha^*, (\hat{s}_A, \hat{b}_E) \rangle &\leq [TV_{LB}^\Gamma](\hat{s}_A, \hat{b}_E) \\ &\leq [TV^*](\hat{s}_A, \hat{b}_E) && \text{since } V_{LB}^\Gamma \leq V^* \\ &= V^*(\hat{s}_A, \hat{b}_E) && \text{by Theorem 1.} \end{aligned} \quad (\text{B.1})$$

Combining this inequality with $V_{LB}^\Gamma \leq V^*$, we have $V_{LB}^{\Gamma'} \leq V^*$ as required. \square

Lemma 3 (ISPP backup). *The FCP Φ_{product} returned by Algorithm 2 is a constant-FCP of ϕ for α^* and the region-by-region backup for α^* satisfies (12).*

Proof. For the PWC α -functions in the input of Algorithm 2, let $\Phi = \sum_{s'_A \in \bar{S}_A} \Phi_{s'_A}$, where $\Phi_{s'_A}$ is an FCP of S for $\alpha^{s'_A}$.

According to Assumption 1, there exists a preimage-FCP of Φ for action \bar{a} . Through the image, split, preimage and product operations of Algorithm 2, all the states in any region $\phi' \in \Phi_{\text{product}}$ have the same reward and reach the same regions of Φ . Since each α -function $\alpha^{s'_A}$ is constant over each region in Φ , all states in ϕ' have the same backup value from $\alpha^{s'_A}$ for $s'_A \in \bar{S}_A$. This implies that Φ_{product} is a preimage FCP of Φ for action \bar{a} . Since the value backup (12) is used for each region in Φ_{product} and the image is from the region ϕ , then Φ_{product} is a constant-FCP of ϕ for α^* , and thus the value backup (12) for α^* is achieved by considering the regions of Φ_{product} . \square

Lemma 4 (Upper bound). *Given belief $(s_A, b_E) \in S_B$, if $p^* = [TV_{UB}^\Upsilon](s_A, b_E)$, then p^* is an upper bound of V^* at (s_A, b_E) , i.e., $p^* \geq V^*(s_A, b_E)$, and if $\Upsilon' = \Upsilon \cup \{((s_A, b_E), p^*)\}$, then $V_{UB}^\Upsilon \geq V_{UB}^{\Upsilon'} \geq V^*$ and $V_{UB}^{\Upsilon'}(s_A, b_E) \leq [TV_{UB}^\Upsilon](s_A, b_E)$.*

Proof. Consider an upper bound V_{UB}^Υ such that $V_{UB}^\Upsilon \geq V^*$. By construction, each pair $((s_A^i, b_E^i), y_i)$ in Υ satisfies $V^*(s_A^i, b_E^i) \leq y_i$.

Now suppose for belief $(s_A, b_E) \in S_B$ we let $p^* = [TV_{UB}^\Upsilon](s_A, b_E)$ and $\Upsilon' = \Upsilon \cup \{((s_A, b_E), p^*)\}$. The new upper bound $V_{UB}^{\Upsilon'}$ after updating V_{UB}^Υ at (s_A, b_E) through Algorithm 1, satisfies $V_{UB}^\Upsilon \geq V_{UB}^{\Upsilon'}$ by (9). By construction of p^* we have:

$$p^* = [TV_{UB}^\Upsilon](s_A, b_E)$$

$$\begin{aligned} &\geq [TV^*](s_A, b_E) && \text{since } V_{UB}^\Upsilon \geq V^* \\ &= V^*(s_A, b_E) && \text{by Theorem 1.} \end{aligned}$$

Next we have:

$$\begin{aligned} V_{UB}^{\Upsilon'}(s_A, b_E) &\leq p^* && \text{since } ((s_A, b_E), p^*) \in \Upsilon' \text{ and (9)} \\ &= [TV_{UB}^\Upsilon](s_A, b_E) && \text{by construction of } p^*. \end{aligned}$$

It therefore remains to prove the last part, i.e. that $V_{UB}^{\Upsilon'} \geq V^*$. Now for any $(s'_A, b'_E) \in S_B$, if $s'_A \neq s_A$, then using the fact that $\Upsilon' = \Upsilon \cup \{((s_A, b_E), p^*)\}$ and (9) we have:

$$\begin{aligned} V_{UB}^{\Upsilon'}(s'_A, b'_E) &= V_{UB}^\Upsilon(s'_A, b'_E) \\ &\geq V^*(s'_A, b'_E) && \text{since } V_{UB}^\Upsilon \geq V^*. \end{aligned}$$

On the other hand, if $s'_A = s_A$, then using (9) there exists $\langle \hat{\lambda}_i \rangle_{i \in I_{s_A}}$ with $\hat{\lambda}_i \geq 0$ and $\sum_{i \in I_{s_A}} \hat{\lambda}_i = 1$ such that:

$$V_{UB}^{\Upsilon'}(s'_A, b'_E) = \sum_{i \in I_{s_A}} \hat{\lambda}_i y_i + K_{UB} \left(b'_E, \sum_{i \in I_{s_A}} \hat{\lambda}_i b_E^i \right). \quad (\text{B.2})$$

Now using Theorem 2 we have:

$$\begin{aligned} V^*(s'_A, b'_E) &\leq V^*(s_A, \sum_{i \in I_{s_A}} \hat{\lambda}_i b_E^i) + K \left(b'_E, \sum_{i \in I_{s_A}} \hat{\lambda}_i b_E^i \right) \\ &\leq \sum_{i \in I_{s_A}} \hat{\lambda}_i V^*(s_A, b_E^i) + K \left(b'_E, \sum_{i \in I_{s_A}} \hat{\lambda}_i b_E^i \right) && \text{since } V^* \text{ is convex in } S_B \\ &\leq \sum_{i \in I_{s_A}} \hat{\lambda}_i V^*(s_A, b_E^i) + K_{UB} \left(b'_E, \sum_{i \in I_{s_A}} \hat{\lambda}_i b_E^i \right) && \text{by (10)} \\ &\leq \sum_{i \in I_{s_A}} \hat{\lambda}_i y_i + K_{UB} \left(b'_E, \sum_{i \in I_{s_A}} \hat{\lambda}_i b_E^i \right) && \text{since if } i \in I_{s_A}, \text{ then } ((s_A, b_E^i), y_i) \in \Upsilon \\ &= V_{UB}^{\Upsilon'}(s'_A, b'_E) && \text{by (B.2).} \end{aligned}$$

Therefore since these are the only cases to consider for $(s'_A, b'_E) \in S_B$ we have $V_{UB}^{\Upsilon'} \geq V^*$ as required. \square

Lemma 5 (LP for upper bound). *The function K_{UB} from (17) satisfies (10), and for particle-based belief (s_A, b_E) represented by $\{(s_E^i, w_i)\}_{i=1}^{N_b}$, we have that $V_{UB}^\Upsilon(s_A, b_E)$ is the optimal value of the LP:*

$$\begin{aligned} &\text{minimize: } \sum_{k \in I_{s_A}} \lambda_k y_k + (U - L) N_b c \\ &\text{subject to: } c \geq |w_i - \sum_{k \in I_{s_A}} \lambda_k P(s_E^i; b_E^k)| \text{ for } 1 \leq i \leq N_b \\ &\quad \lambda_k \geq 0 \text{ for } k \in I_{s_A} \text{ and } \sum_{k \in I_{s_A}} \lambda_k = 1. \end{aligned}$$

Proof. Consider any particle-based beliefs (s_A, b_E) and (s_A, b'_E) where (s_A, b_E) is represented by the weighted particle set $\{(s_E^i, w_i)\}_{i=1}^{N_b}$. Recall that $S_E^{b_E > b'_E} = \{s_E \in S_E^{s_A} \mid b_E(s_E) - b'_E(s_E) > 0\}$, now by definition of $K(b_E, b'_E)$, see Theorem 2, we have:

$$\begin{aligned}
K(b_E, b'_E) &= (U - L) \int_{s_E \in S_E^{b_E > b'_E}} (b_E(s_E) - b'_E(s_E)) ds_E \\
&= (U - L) \int_{s_E \in S_E^{b_E > b'_E}} |b_E(s_E) - b'_E(s_E)| ds_E && \text{by definition of } S_E^{b_E > b'_E} \\
&\leq (U - L) \sum_{i=1}^{N_b} |P(s_E^i; b_E) - P(s_E^i; b'_E)| && \text{by Definition 7} \\
&\leq (U - L) N_b \max_{1 \leq i \leq N_b} |P(s_E^i; b_E) - P(s_E^i; b'_E)| && \text{rearranging} \\
&= (U - L) N_b \max_{s_E \in S_E \wedge b_E(s_E) > 0} |P(s_E; b_E) - P(s_E; b'_E)| && \text{rearranging} \\
&= K_{UB}(b_E, b'_E) && \text{by (17).}
\end{aligned}$$

Furthermore, (17) implies $K_{UB}(b_E, b_E) = 0$, and therefore K_{UB} satisfies (10).

Next suppose $\Upsilon = \{((s_A^k, b_E^k), y_k) \mid k \in I\}$, letting $b'_E = \sum_{k \in I_{s_A}} \lambda_k b_E^k$ and $c = \max_{s_E \in S_E \wedge b_E(s_E) > 0} |P(s_E; b_E) - P(s_E; b'_E)|$, by definition of K_{UB} , see (17), we have:

$$\begin{aligned}
K_{UB}(b_E, b'_E) &= (U - L) N_b \max_{s_E \in S_E \wedge b_E(s_E) > 0} |P(s_E; b_E) - P(s_E; b'_E)| \\
&= (U - L) N_b c && \text{by definition of } c
\end{aligned}$$

and therefore:

$$\sum_{k \in I_{s_A}} \lambda_k y_k + K_{UB}(b_E, b'_E) = \sum_{k \in I_{s_A}} \lambda_k y_k + (U - L) N_b c.$$

Furthermore, for any $1 \leq i \leq N_b$, by definition of c and since (s_A, b_E) is represented by the weighted particle set $\{(s_E^i, w_i)\}_{i=1}^{N_b}$ we have:

$$\begin{aligned}
c &\geq |P(s_E^i; b_E) - P(s_E^i; b'_E)| \\
&= |w_i - P(s_E^i; b'_E)| && \text{since } \{(s_E^i, w_i)\}_{i=1}^{N_b} \text{ represents } b_E \\
&= \left| w_i - P\left(s_E^i; \sum_{k \in I_{s_A}} \lambda_k b_E^k\right) \right| && \text{by definition of } b'_E.
\end{aligned}$$

Therefore, the optimization problem (9) can be equivalently written as the LP of Lemma 5, i.e., the optimal value is equal to $V_{UB}^\Upsilon(s_A, b_E)$. \square

Lemma 6 (Region-based belief closure). *If $\delta_E^i(\cdot, a) : S_E \rightarrow \delta_E^i(S_E, a)$ is piecewise differentiable and invertible from S_E to $T \subseteq S_E$, and the Jacobian determinant of the inverse function, i.e., for any $s'_E \in T$:*

$$\text{Jac}(s'_E) := \det \left(\frac{d\delta_E^{i,-1}(s'_E, a)}{ds'_E} \right)$$

is PWC for $a \in \text{Act}$ and $1 \leq i \leq N_e$, then region-based beliefs are closed under belief updates.

Proof. Since $\delta_E^i(\cdot, a)$ is piecewise differentiable and piecewise invertible, let $\phi_E \subseteq S_E$ be a region over which $\delta_E^i(\cdot, a)$ is differentiable and invertible. Suppose that X_E is a random variable taking values in ϕ_E , and that X_E has a continuous uniform distribution with probability density function b_E . Due to the differentiability and thus continuity of $\delta_E^i(\cdot, a)$, the image $\phi'_E = \{s'_E \mid s'_E = \delta_E^i(s_E, a) \wedge s_E \in \phi_E\}$ is a region in S_E . Furthermore, suppose $X'_E = \delta_E^i(X_E, a)$ is a new random variable taking values in ϕ'_E and let b'_E be the probability density function for X'_E over ϕ'_E . We next prove that b'_E is a PWC uniform distribution under the given conditions.

Let $\delta_E^{i,-1}(\cdot, a)$ be the inverse function of $\delta_E^i(\cdot, a)$ in ϕ_E . If $\phi'_1 \subseteq \phi'_E$, letting ϕ_1 be the preimage of ϕ'_1 , then

$$\begin{aligned} P(X'_E \in \phi'_1) &= P(\delta_E^i(X_E, a) \in \phi'_1) && \text{since } X'_E = \delta_E^i(X_E, a) \\ &= \int_{s_E \in \phi_1} b_E(s_E) ds_E && \text{by definition of } b_E. \end{aligned} \quad (\text{B.3})$$

Using the change of variables $s_E = \delta_E^{i,-1}(s'_E, a)$ we have that:

$$\begin{aligned} ds_E &= \det \left(\frac{d\delta_E^{i,-1}(s'_E, a)}{ds'_E} \right) ds'_E \\ &= \text{Jac}(s'_E) ds'_E && \text{by definition of the Jacobian determinant} \end{aligned}$$

and substituting this into (B.3) we have:

$$P(X'_E \in \phi'_1) = \int_{s'_E \in \phi'_1} b_E(\delta_E^{i,-1}(s'_E, a)) \text{Jac}(s'_E) ds'_E.$$

Therefore we have that for any $s'_E \in \phi'_1$:

$$b'_E(s'_E) = b_E(\delta_E^{i,-1}(s'_E, a)) \text{Jac}(s'_E)$$

and since $b_E(\delta_E^{i,-1}(s'_E, a)) = b_E(s_E)$ for $s_E \in \phi_E$ is constant and by construction $\text{Jac}(s'_E)$ is PWC, we have that b'_E is PWC over ϕ'_E as required.

We conclude that $\delta_E^i(\cdot, a)$ transforms a random variable which has a continuous uniform distribution in a region into a new random variable which has a continuous uniform distribution over finitely many regions. Therefore, region-based belief are closed under $\delta_E^i(\cdot, a)$. \square

Lemma 7 (Region-based belief update). *For region-based belief (s_A, b_E) represented by $\{(\phi_E^i, w_i)\}_{i=1}^{N_b}$, action a and observation s'_A : (s'_A, b'_E) returned by Algorithm 4 is region-based and $b'_E = b_E^{s_A, a, s'_A}$. Furthermore, if $h : S \rightarrow \mathbb{R}$ is PWC and Φ_E is a constant-FCP of S_E for h at s_A , then $\langle h, (s_A, b_E) \rangle = \sum_{i=1}^{N_b} \sum_{\phi_E \in \Phi_E} h(s_A, s_E) w_i \text{vol}(\phi_E^i \cap \phi_E)$ where $s_E \in \phi_E$.*

Proof. Consider a region-based belief (s_A, b_E) represented by $\{(\phi_E^i, w_i)\}_{i=1}^{N_b}$, action a and observation s'_A and suppose that the belief (s'_A, b'_E) is returned by Algorithm 4.

Since $\delta_E^i(\cdot, a)$ is piecewise continuous by Lemma 6, then for any region $\phi_E \subseteq \Phi_E$, the image $\{\delta_E^i(s_E, a) \mid s_E \in \phi_E\}$ can be represented as a union of regions. Furthermore, due to the invertibility of $\delta_E^i(\cdot, a)$, these regions are disjoint and the image is uniformly reached. Letting $\phi_{ij} = \{\delta_E^j(s_E, a) \mid s_E \in \phi_E^i\}$, according to the belief update (4) and the belief expression in Definition 8, we have:

$$\begin{aligned}
\int_{s_E \in S_E} b_E(s_E) \delta_E(s_E, a)(s'_E) ds_E &= \int_{s_E \in S_E} \left(\sum_{i=1}^{N_b} \chi_{\phi_E^i}(s_E) w_i \right) \delta_E(s_E, a)(s'_E) ds_E \\
&= \sum_{i=1}^{N_b} \left(\int_{s_E \in S_E} \chi_{\phi_E^i}(s_E) w_i \delta_E(s_E, a)(s'_E) ds_E \right) && \text{rearranging} \\
&= \sum_{i=1}^{N_b} \left(\int_{s_E \in \phi_E^i} w_i \delta_E(s_E, a)(s'_E) ds_E \right) && \text{by definition of } \chi_{\phi_E^i} \\
&= \sum_{i=1}^{N_b} \left(\int_{s_E \in \phi_E^i} w_i \left(\sum_{j=1}^{N_e} \chi_{\phi_E^{ij}}(s'_E) \frac{\mu_j}{\text{vol}(\phi_E^{ij})} ds_E \right) \right) \\
&\quad \text{by definition of } \phi_{ij} \text{ and since it is uniformly reached by Lemma 6} \\
&= \sum_{i=1}^{N_b} \sum_{j=1}^{N_e} \left(\int_{s_E \in \phi_E^i} w_i \chi_{\phi_E^{ij}}(s'_E) \frac{\mu_j}{\text{vol}(\phi_E^{ij})} ds_E \right) && \text{rearranging} \\
&= \sum_{i=1}^{N_b} \sum_{j=1}^{N_e} w_i \chi_{\phi_E^{ij}}(s'_E) \frac{\mu_j}{\text{vol}(\phi_E^{ij})} \left(\int_{s_E \in \phi_E^i} ds_E \right) && \text{rearranging}
\end{aligned}$$

$$= \sum_{i=1}^{N_b} \sum_{j=1}^{N_e} \chi_{\phi_E^{ij}}(s'_E) \frac{w_i \mu_j \text{vol}(\phi_E^i)}{\text{vol}(\phi_E^{ij})} \quad \text{by definition of vol.}$$

Therefore, b'_E can be constructed by normalizing $\int_{s_E \in S_E} b_E(s_E) \delta_E(s_E, a)(s'_E) ds_E$, which is a region-based belief.

Next, consider any observation s_A and PWC $h : S \rightarrow \mathbb{R}$ where Φ_E is a constant-FCP of S_E for h at s_A . By (5) we have:

$$\begin{aligned} \langle h, (s_A, b_E) \rangle &= \int_{s_E \in S_E} h(s_A, s_E) b_E(s_E) ds_E \\ &= \int_{s_E \in S_E} h(s_A, s_E) \sum_{i=1}^{N_b} \chi_{\phi_E^i}(s_E) w_i ds_E && \text{by Definition 8} \\ &= \sum_{i=1}^{N_b} \left(\int_{s_E \in S_E} h(s_A, s_E) \chi_{\phi_E^i}(s_E) w_i ds_E \right) && \text{rearranging} \\ &= \sum_{i=1}^{N_b} \sum_{\phi_E \in \Phi_E} \left(\int_{s_E \in \phi_E} h(s_A, s_E) \chi_{\phi_E^i}(s_E) w_i ds_E \right) && \text{since } \Phi_E \text{ is an FCP} \\ &= \sum_{i=1}^{N_b} \sum_{\phi_E \in \Phi_E} h(s_A, \phi_E) \left(\int_{s_E \in \phi_E} \chi_{\phi_E^i}(s_E) w_i ds_E \right) \\ &\quad \text{since } \Phi_E \text{ is a constant-FCP of } S_E \text{ for } h \text{ at } s_A \\ &= \sum_{i=1}^{N_b} \sum_{\phi_E \in \Phi_E} h(s_A, \phi_E) w_i \text{vol}(\phi_E^i \cap \phi_E) && \text{by definition of vol} \end{aligned}$$

which completes the proof. \square

Lemma 8 (Region-based upper bound). *For region-based belief (s_A, b_E) represented by $\{(\phi_E^i, w_i)\}_{i=1}^{N_b}$ and $\Upsilon = \{((s_A^k, b_E^k), y_k) \mid k \in I\}$, if $K_{UB} = K$, (ϕ_E^{\max}, p) is returned by Algorithm 5, $b'_E = \sum_{k \in I_{s_A}} \lambda_k^* b_E^k$ and $\phi_E^{\max} \subseteq S_E^{b_E > b'_E}$ where λ_k^* is a solution to the LP of Algorithm 5, then p is an upper bound of V_{UB}^Υ at (s_A, b_E) . Furthermore, if $N_b = 1$, then $p = V_{UB}^\Upsilon(s_A, b_E)$.*

Proof. Suppose that (s_A, b_E) is a region-based belief represented by $\{(\phi_E^i, w_i)\}_{i=1}^{N_b}$, $\Upsilon = \{((s_A^k, b_E^k), y_k) \mid k \in I\}$ and suppose $K_{UB} = K$ and (ϕ_E^{\max}, p) is returned by Algorithm 5, $b'_E = \sum_{k \in I_{s_A}} \lambda_k^* b_E^k$ where λ_k^* is a solution to the LP of Algorithm 5 and $\phi_E^{\max} \subseteq S_E^{b_E > b'_E}$. Furthermore for each $k \in I$ suppose that (s_A^k, b_E^k) is a region-based belief represented by $\{(\phi_E^{kj}, w_{kj})\}_{j=1}^{N_b^k}$.

Since $K_{UB} = K$, by definition of K (see Theorem 2) we have:

$$K_{UB}(b_E, b'_E) = (U - L) \int_{s_E \in S_E^{b_E > b'_E}} (b_E(s_E) - b'_E(s_E)) ds_E$$

$$\begin{aligned}
&= (U - L) \left(\int_{s_E \in S_E^{b_E > b'_E}} b_E(s_E) - \int_{s_E \in S_E^{b_E > b'_E}} b'_E(s_E) ds_E \right) && \text{rearranging} \\
&\leq (U - L) \left(\int_{s_E \in S_E} b_E(s_E) - \int_{s_E \in S_E^{b_E > b'_E}} b'_E(s_E) ds_E \right) && \text{rearranging} \\
&= (U - L) \left(1 - \int_{s_E \in S_E^{b_E > b'_E}} b'_E(s_E) ds_E \right) && \text{since } b_E \in \mathbb{P}(S_E).
\end{aligned} \tag{B.4}$$

Now since $\phi_E^{\max} \subseteq S_E^{b_E > b'_E}$ we have:

$$\begin{aligned}
&\int_{s_E \in S_E^{b_E > b'_E}} b'_E(s_E) ds_E \geq \int_{s_E \in \phi_E^{\max}} b'_E(s_E) ds_E \\
&= \int_{s_E \in \phi_E^{\max}} \left(\sum_{k \in I_{s_A}} \lambda_k^* b_E^k(s_E) \right) ds_E && \text{by definition of } b'_E \\
&= \sum_{k \in I_{s_A}} \left(\int_{s_E \in \phi_E^{\max}} \lambda_k^* b_E^k(s_E) ds_E \right) && \text{rearranging} \\
&= \sum_{k \in I_{s_A}} \left(\int_{s_E \in \phi_E^{\max}} \lambda_k^* \sum_{j=1}^{N_b^k} \chi_{\phi_E^{kj}}(s_E) w_{kj} ds_E \right) \\
&\hspace{15em} \text{since } \{(\phi_E^{kj}, w_{kj})\}_{j=1}^{N_b^k} \text{ represents } b_E^k \\
&= \sum_{k \in I_{s_A}} \sum_{j=1}^{N_b^k} \lambda_k^* \left(\int_{s_E \in \phi_E^{\max}} \chi_{\phi_E^{kj}}(s_E) w_{kj} ds_E \right) && \text{rearranging} \\
&= \sum_{k \in I_{s_A}} \sum_{j=1}^{N_b^k} \lambda_k^* w_{kj} \text{vol}(\phi_E^{kj} \cap \phi_E^{\max}) && \text{by definition of vol.}
\end{aligned} \tag{B.5}$$

Thus, substituting (B.5) into (B.4) we have:

$$K_{UB}(b_E, b'_E) \leq (U - L) \left(1 - \sum_{k \in I_{s_A}} \sum_{j=1}^{N_b^k} \lambda_k^* w_{kj} \text{vol}(\phi_E^{kj} \cap \phi_E^{\max}) \right)$$

and using (9), it follows that the optimal value p to the LP of Algorithm 5 is an upper bound of $V_{UB}^{\mathcal{R}}$ at (s_A, b_E) .

Finally, suppose that $N_b = 1$. Therefore $\phi_E^{\max} = \phi_E^1$ and since ϕ_E^1 is the unique region with positive probabilities for b_E , by definition of $S_E^{b_E > b'_E}$ it

follows that $S_E^{b_E > b'_E} \subseteq \phi_E^1$. Combining these with $\phi_E^{\max} \subseteq S_E^{b_E > b'_E}$, we have that $S_E^{b_E > b'_E} = \phi_E^{\max} = \phi_E^1$. Therefore, all the inequalities above become equalities, and therefore $p = V_{UB}^r(s_A, b_E)$. \square

References

- [1] J. Gupta, M. Egorov, M. Kochenderfer, Cooperative multi-agent control using deep reinforcement learning, in: Proc. AAMAS’17, Vol. 10643 of LNCS, Springer, 2017, pp. 66–83.
- [2] S. Amizadeh, H. Palangi, A. Polozov, Y. Huang, K. Koishida, Neuro-symbolic visual reasoning: Disentangling, in: Proc. ICML’20, PMLR, 2020, pp. 279–290.
- [3] S. Shalev-Shwartz, S. Shammah, A. Shashua, Safe, multi-agent, reinforcement learning for autonomous driving, [arXiv:1610.03295](https://arxiv.org/abs/1610.03295) (2016).
- [4] M. E. Akintunde, E. Botoeva, P. Kouvaros, A. Lomuscio, Verifying Strategic Abilities of Neural-symbolic Multi-agent Systems, in: Proc. KR’20, 2020, pp. 22–32.
- [5] C. H. Papadimitriou, J. N. Tsitsiklis, The complexity of Markov decision processes, Math. Oper. Res. 12 (3) (1987) 441–450.
- [6] K. Chatterjee, M. Chmelík, R. Gupta, A. Kanodia, Optimal cost almost-sure reachability in POMDPs, Artificial Intelligence 234 (2016) 26–48.
- [7] G. Shani, J. Pineau, R. Kaplow, A survey of point-based POMDP solvers, Auton. Agents Multi-Agent Syst. 27 (1) (2013) 1–51.
- [8] J. M. Porta, N. Vlassis, M. T. Spaan, P. Poupart, Point-based value iteration for continuous POMDPs, JMLR 7 (2006) 2329–2367.
- [9] M. K. Sarker, L. Zhou, A. Eberhart, P. Hitzler, Neuro-symbolic artificial intelligence, AI Comm. (2021) 1–13.
- [10] T. Smith, R. Simmons, Heuristic search value iteration for POMDPs, in: Proc. UAI’04, AUAI, 2004, p. 520–527.
- [11] L. Burks, I. Loefgren, N. R. Ahmed, Optimal continuous state POMDP planning with semantic observations: A variational approach, IEEE Trans. Robotics 35 (6) (2019) 1488–1507.

- [12] Z. Zamani, S. Sanner, P. Poupart, K. Kersting, Symbolic dynamic programming for continuous state and observation POMDPs, *Adv. Neural Inf. Process. Syst.* 25 (2012).
- [13] X. Jiang, J. Yang, X. Tan, H. Xi, Observation-based optimization for POMDPs with continuous state, observation, and action spaces, *IEEE Trans. Automatic Control* 64 (5) (2018) 2045–2052.
- [14] S. Brechtel, T. Gindele, R. Dillmann, Solving continuous POMDPs: Value iteration with incremental learning of an efficient space representation, in: *Proc. ICML’13*, PMLR, 2013, pp. 370–378.
- [15] J. Van Den Berg, S. Patil, R. Alterovitz, Efficient approximate value iteration for continuous gaussian POMDPs, in: *Proc. AAAI’12*, Vol. 26(1), 2012, pp. 1832–1838.
- [16] M. H. Lim, C. J. Tomlin, Z. N. Sunberg, Voronoi progressive widening: Efficient online solvers for continuous state, action, and observation POMDPs, in: *Proc. CDC’21*, 2021, pp. 4493–4500.
- [17] T. Smith, R. Simmons, Point-based POMDP algorithms: Improved analysis and implementation, in: *Proc. UA’05, AUA*, 2005, p. 542–549.
- [18] K. Horák, B. Bošanský, V. Kovařík, C. Kiekintveld, Solving zero-sum one-sided partially observable stochastic games, *Artificial Intelligence* 316 (2023) 103838.
- [19] J. Pineau, G. Gordon, S. Thrun, et al., Point-based value iteration: An anytime algorithm for POMDPs, in: *Proc. IJCAI’13*, Vol. 3, 2003, pp. 1025–1032.
- [20] G. Shani, R. I. Brafman, S. E. Shimony, Forward search value iteration for pomdps., in: *Proc. IJCAI’07*, 2007, pp. 2619–2624.
- [21] H. Kurniawati, D. Hsu, W. S. Lee, SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces., in: *Robotics: Science and Systems*, Vol. 4, 2008, pp. 65–72.
- [22] R. Yan, G. Santos, X. Duan, D. Parker, M. Kwiatkowska, Finite-horizon equilibria for neuro-symbolic concurrent stochastic games, in: *Proc. UAI’22*, Vol. 180, PMLR, 2022, pp. 2170–2180.

- [23] R. Yan, G. Santos, G. Norman, D. Parker, M. Kwiatkowska, Strategy synthesis for zero-sum neuro-symbolic concurrent stochastic games, [arXiv.2202.06255](#) (2022).
- [24] M. Kwiatkowska, G. Norman, D. Parker, G. Santos, R. Yan, Probabilistic Model Checking for Strategic Equilibria-Based Decision Making: Advances and Challenges, in: 47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022), Vol. 241, 2022, pp. 4:1–4:22.
- [25] M. Cleaveland, L. Lindemann, R. Ivanov, G. J. Pappas, Risk verification of stochastic systems with neural network controllers, *Artificial Intelligence* 313 (2022) 103782.
- [26] S. Carr, N. Jansen, U. Topcu, Verifiable RNN-based policies for POMDPs under temporal logic constraints, in: *Proc. IJCAI’20*, 2020, pp. 4121–4127.
- [27] D. Bertsekas, *Dynamic programming and optimal control: Volume I*, Athena scientific, 2012.
- [28] L. P. Kaelbling, M. L. Littman, A. R. Cassandra, Planning and acting in partially observable stochastic domains, *Artificial intelligence* 101 (1-2) (1998) 99–134.
- [29] K. D. Julian, M. J. Kochenderfer, A reachability method for verifying dynamical systems with deep neural network controllers, [arXiv.1903.00520](#) (2019).
- [30] K. Matoba, F. Fleuret, Computing preimages of deep neural networks with applications to safety, [openreview.net/forum?id=FN7_BUOG78e](#) (2020).
- [31] E. J. Sondik, The optimal control of partially observable Markov processes over the infinite horizon: Discounted costs, *Oper. Res.* 26 (2) (1978) 282–304.
- [32] K. Horák, B. Bošanský, K. Chatterjee, Goal-HSVI: Heuristic search value iteration for goal POMDPs, in: *Proc. JCAI’18*, 2018, pp. 4764–4770.

- [33] T. Smith, Probabilistic planning for robotic exploration, Carnegie Mellon University, 2007.
- [34] D. Crisan, A. Doucet, A survey of convergence results on particle filtering methods for practitioners, *IEEE Transactions on Signal Processing* 50 (3) (2002) 736–746.
- [35] A. Doucet, N. De Freitas, N. J. Gordon (Eds.), *Sequential Monte Carlo methods in practice*, Vol. 1(2), Springer, 2001.
- [36] M. Lauri, D. Hsu, J. Pajarinen, Partially observable Markov decision processes in robotics: A survey, *IEEE Transactions on Robotics* (2022) 1–20.
- [37] X. Ma, P. Karkus, D. Hsu, W. S. Lee, Particle filter recurrent neural networks, in: *Proc. AAAI’20*, Vol. 34(4), 2020, pp. 5101–5108.
- [38] A. Doucet, S. Godsill, C. Andrieu, On sequential Monte Carlo sampling methods for Bayesian filtering, *Statistics and computing* 10 (3) (2000) 197–208.
- [39] A. Papoulis, S. U. Pillai, *Probability, random variables, and stochastic processes*, Tata McGraw-Hill Education, 2002.
- [40] R. Bagnara, P. M. Hill, E. Zaffanella, The Parma Polyhedra Library: Toward a complete set of numerical abstractions for the analysis and verification of hardware and software systems, *Sci. Comput. Program.* 72 (1) (2008) 3–21, bugseng.com/ppl.
- [41] Gurobi Optimization, LLC, *Gurobi Optimizer Reference Manual*, gurobi.com (2021).
- [42] L. De Moura, N. Bjørner, Z3: An efficient SMT solver, in: *Proc. TACAS’08*, Vol. 4963 of LNCS, Springer, 2008, pp. 337–340, github.com/Z3Prover/z3.
- [43] K. D. Julian, M. J. Kochenderfer, Guaranteeing safety for neural network-based aircraft collision avoidance systems, in: *Proc. DASC’19*, IEEE, 2019, pp. 1–10, the source code is available from openreview.net/forum?id=FN7_BUOG78e.

- [44] K. D. Julian, S. Sharma, J.-B. Jeannin, M. J. Kochenderfer, Verifying aircraft collision avoidance neural networks through linear approximations of safe regions, [arXiv.1903.00762](https://arxiv.org/abs/1903.00762) The source code is available from github.com/sisl/VerticalCAS (2019).