Available online at www.sciencedirect.com

ScienceDirect

Artificial Intelligence 00 (2022) 1–34

Artificial Intelligence

Competence-Aware Systems

Connor Basich*[a], Justin Svegliato[b], Kyle H. Wray[c], Stefan Witwicki[c], Joydeep Biswas[d], Shlomo Zilberstein[a]

[a] *University of Massachusetts Amherst, Amherst, MA, USA {cbasich, shlomo}@cs.umass.edu*
[b] *University of California, Berkeley, Berkeley, CA, USA, jsvegliato@berkeley.edu*
[c] *Alliance Innovation Lab Silicon Valley, Santa Clara, CA, USA, {kyle.wray, stefan.witwicki}@nissan-usa.com*
[d] *The University of Texas at Austin, Austin, TX, USA joydeepb@cs.texas.edu*
* *Corresponding Author*

Abstract

Building autonomous systems for deployment in the open world has been a longstanding objective in both artificial intelligence and robotics. The open world, however, presents challenges that question some of the assumptions often made in contemporary AI models. Autonomous systems that operate in the open world face complex, non-stationary environments wherein enumerating all situations the system may face over the course of its deployment is intractable. Nevertheless, these systems are expected to operate safely and reliably for extended durations. Consequently, AI systems often rely on some degree of human assistance to mitigate risks while completing their tasks, and are hence better treated as *semi-autonomous systems*. In order to reduce unnecessary reliance on humans and optimize autonomy, we propose a novel introspective planning model—*competence-aware systems* (CAS)—that enables a semi-autonomous system to reason about its own competence and allowed level of autonomy by leveraging human feedback or assistance. A CAS learns to adjust its level of autonomy based on experience and interactions with a human authority so as to reduce improper reliance on the human and optimize the degree of autonomy it employs in any given circumstance. To handle situations in which the initial CAS model has insufficient state information to properly discriminate feedback received from humans, we introduce a methodology called *iterative state space refinement* that gradually increases the granularity of the state space online. The approach exploits information that exists in the standard CAS model and requires no additional input from the human. The result is an agent that can more confidently predict the correct feedback from the human authority in each level of autonomy, enabling it learn its competence in a larger portion of the state space.

Keywords: probabilistic planning, human-agent systems, competence-aware systems, risk-aware autonomy, adjustable autonomy, decision making under uncertainty

1. Introduction

Autonomous systems are increasingly deployed in the *open world*, involving highly complex and unstructured domains. Examples of these systems include space exploration rovers [36, 63], autonomous underwater vehicles [20, 54, 85], service robots [10, 43, 59], and autonomous vehicles [15, 16, 28]. Because it is infeasible to completely model the open world, these systems must rely on approximate models of their domains that may not be sufficient for handling every situation [78, 89], introducing potentially risky behavior when the system attempts to act autonomously where it is not competent to do so. Nevertheless, these systems are expected to maintain safe and reliable operation over the course of potentially long-term deployments. To accomplish that, they often rely on various forms of human supervision, assistance, and intervention. In that sense, many of the sophisticated AI systems under development today are at best *semi-autonomous* in that they operate autonomously only under certain conditions, and otherwise require human intervention in order to complete their assigned tasks [25, 104].

Reliance on human assistance has been explored extensively to address the limited competence of autonomous systems [33, 37, 46, 62, 66, 77, 93]. Often, this has been explored in the context of *varying levels of autonomy*, a paradigm for modeling gradations in autonomous behavior within a human-agent team [64, 83], where each level of autonomy corresponds to some set of constraints, limitations, or requirements on autonomous operation. For example, on the two extremes would be full autonomous operation, and full human control (no autonomy). This paradigm has already taken hold in several industrial applications where safety and reliability are critical, including driving automation [76], robotic medical devices [6, 34, 101], and autonomous legal reasoning [31, 32].

Human assistance may be available in different forms or modalities, corresponding to different degrees of competence of a semi-autonomous system. Different forms of human assistance compensate for the limitations imposed in each level of autonomy and consequently mitigate the potential for risky behavior, while still ensuring that the system's task is completed. For example, Veloso et al. [92, 93] designed the CoBot system that can aid humans in an office environment as an assistive robot in a variety of pick-up and delivery tasks. However, as the CoBot has no arms to grasp objects, it cannot perform its tasks entirely autonomously, and must instead seek assistance from humans to compensate for its limitation, for example by placing or removing objects in its basket. Ficuciello et al. [34] proposed a level of autonomy framework for a surgical assistive medical robot with four levels of autonomy, where the lowest two involve purely assistive actions to aid the human who is the primary executor, and the highest two involve fully autonomous execution by the robot with assistance from the human in the form of surgical strategy selection.

In this work, we are primarily concerned with the risk associated with a system that operates at a level of autonomy that is inappropriate for a task given its capabilities; for instance, an office robot that autonomously handles fragile items it is not competent to handle safely (i.e., without a high risk of breaking). Hence, we aim to develop systems that are aware of their *own competence*, which we define to be the *optimal level of autonomy* to employ in any given situation conditioned on the availability of suitable human assistance. A system that is aware of its own competence when generating plans can therefore mitigate the potential for risky behavior by optimizing the degree of human assistance that it requests, leveraging the human where the system's competence is low, and acting autonomously where it is high.

To further mitigate risks, humans may impose constraints on autonomous operation based on the perceived competence of the system, for instance, by allowing them to intervene in time to prevent risky behavior or by disallowing autonomous behavior entirely. In fact, the perceived risks may be outside the scope of what the autonomous system can detect or reason about, hence enabling us to mitigate a broader range of risks. For example, a robot's sensors may be unable to perceive black ice on a sidewalk, or a nearby obstacle in dense fog, leading to risky behavior if left to operate without supervision in these conditions.

Determining the exact competence of an autonomous system at design time can be very difficult, particularly when the environment is not fully specified or is simply too complex to fully anticipate. For example, a self-driving car may initially be authorized to drive autonomously without supervision only on highways and during the daytime with clear weather. Hence, an initial level of autonomy may be determined *a priori* through testing and evaluation, but adjustments must be made when the system is deployed. Even when developers aim to err on the side of caution, initializing the level of autonomy to be below the system's true competence, it is possible to unintentionally infer from initial testing that the system is more competent than it really is [69, 89]. Therefore, developing mechanisms to explicitly represent, reason about, and adjust the level of autonomy is critical for the success of autonomous systems deployed in the open world.

We propose a planning model called *competence-aware system* (CAS) for operating at multiple levels of autonomy where each level is associated with different forms of human assistance that compensate for the constrained abilities of the system. Motivated by ideas from *collaborative control* [35], the structure of a CAS is illustrated in Figure 1. The model associates with each type of human assistance a set of feedback signals that the system can receive from the human, the likelihood of which can be learned over time. This model enables the system to operate more reliably in the open world, reduce improper reliance on the human and ultimately optimize the autonomous behavior of the system [5]. To address situations where the initial domain model has insufficient information to correctly model human feedback, we introduce an iterative approach to refine the system's state space in order to better discriminate human feedback, producing a more nuanced partitioning of the state-action space with different levels of competence, and allowing the system to better learn and act at its true competence [4].

One of the main characteristics of CAS is that the system must *recognize* the limits on its autonomy, but it is not required to *know the reasons* for these restrictions. This could be seen as a limitation, but we argue that it is an

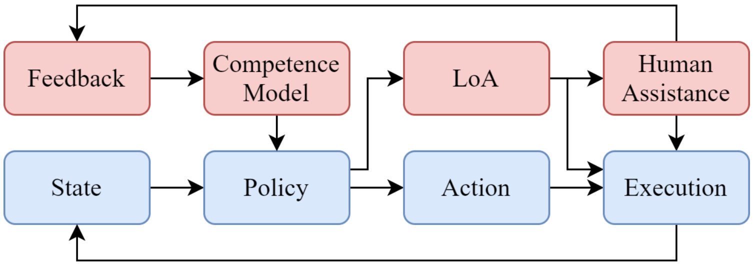


Figure 1: An overview of how competence modeling impacts planning and execution. Here, the system's current state is provided as input to the system's policy which traditionally would only output an action, but in our case also outputs a level of autonomy determined by the competence model in which to perform the action. The level of autonomy dictates the type and degree of human assistance used in the execution of the intended action. The human assistance can also provide additional feedback to the system, which can be used to update and refine the competence model online.

advantage because it allows us to build autonomous systems that respect constraints on autonomy derived from human knowledge that is beyond the scope of the system's reasoning abilities. While we allow situations in which the system does not have complete knowledge of the risks that justify the limitations on autonomy, the system may acquire that knowledge over time.

Our contributions are three fold: (1) a mathematically rigorous formalization of *competence* for automated decision making; (2) a planning framework for a *competence-aware system* that integrates a model of competence with a planning model to enable the system to reduce unnecessary reliance on humans and optimize its autonomous behavior; and (3) a method called *iterative state space refinement* that enables a competence-aware system to refine the granularity of its state representation online. We provide a theoretical analysis of our model and algorithm, a concrete example of a CAS and considerations in its design and implementation, empirical evaluations of our contributions in simulation, and the lessons learned from a preliminary testing of the approach on an autonomous vehicle prototype.

2. Related Work

Researchers in automated planning [38] and reinforcement learning [87] have produced a vast literature devoted to models, languages and algorithms that enable agents to reason about their environment and choose actions intelligently. In this work, we specifically focus on advancing proactive reasoning under uncertainty about when and how to obtain human assistance in order to improve goal achievement or safety. We discuss below three areas of research that are particularly relevant to competence aware systems.

2.1. Systems with Variable Levels of Autonomy

Recognizing the value of human knowledge in planning has led to several research efforts on human-agent collaboration in automated planning and control. *Mixed-initiative planning/control* [14, 18, 33, 37] is a paradigm based on *mixed-initiative interaction* [1, 48] wherein multiple different agents, generally a human an an autonomous system, can take the initiative to act at different stages to best utilize their respective abilities. Recent work has investigated applying mixed-initiative control in the context of variable autonomy [23] in which the level of autonomy (LoA) can change dynamically online. Chiou et al. [22] introduced the *expert-guided mixed-initiative control switcher*, which dynamically adjusts the level of autonomy based on a comparison of the expected performance of a task expert and the observed performance of the current system. Petousakis et al. [66] extended this approach by explicitly modeling the cognitive availability of the human based on real-time vision of the human to better inform the LoA switching decision between the autonomous agent and the human. Our work differs from this prior work in several key aspects. First, we assume that an automated planner determines the level of autonomy for the human-agent team, thereby designating the workload to both the human and the autonomous agent rather than allowing for each to initiate control on their own. Second, we are focused on the problem of learning the true competence of the human-agent system online through the acquisition of feedback from the human in response to actions taken by the agent at different levels of autonomy.

Finally, much of the previous work is either tied to, or focused on, systems with only two levels of autonomy—no autonomy and full autonomy—whereas we emphasize a general model for arbitrary levels of autonomy.

Rigter et al. [72] considered a similar setting in which control of a system is selected from a set of autonomous controllers and a human operator. To reduce the reliance on the human over time, they propose to learn one of the controllers online from demonstrations gained from the human operator. While similarly motivated, we consider a slightly different problem setting. First, we consider one agent operating in different levels of autonomy, each of which may involve some degree of human assistance, rather than all-or-nothing involvement of the human, and allow for the level to change at every time step, rather than being fixed throughout an episode. The idea of learning a controller from human demonstrations is similar to how we propose to learn a model of the human's transition function when they are in control, but in our case we use it only to predict their behavior, not to learn or alter autonomous control.

Symbiotic autonomy is similar in that the aim is to enable the completion of complex tasks by distributing tasks and information across multiple agents. However, the term has been used both to represent human-agent systems where the two agents act asynchronously to perform tasks for *each other*, that is both the human and agent may seek assistance from the other to complete their task [75, 92, 93], as well as systems in which there is a *smart environment* in addition to the autonomous agent and human that provides auxiliary information to the autonomous agent to facilitate it [19, 25, 77]. Generally, our work differs in that we do not consider the environment and we emphasize the use of human assistance to better facilitate the completion of the autonomous agent's task, rather than asynchronously acting in order to help the other agent with their task.

Adjustable autonomy [13, 29, 62, 80, 81, 91, 103] is a closely related paradigm for human-agent teams that is characterized by the ability to dynamically change between different levels, or modes, of autonomy, each of which corresponds to some set of constraints or allowances that affect the actions the human-agent team can successfully perform. It is worth noting that these approaches are largely complementary, and there has been work specifically designed to combine multiple of these approaches [13, 60]. Our work falls generally in the category of adjustable autonomy, but adds two important capabilities to such systems, on top of the fundamental notion of competence. First, we explicitly model multiple forms of human feedback and use this feedback to enable a semi-autonomous system to learn its competence over time. Second, in the CAS model the system learns a predictive model of the human's feedback allowing the system to converge to the optimal level of autonomy over time.

2.2. Learning from Human Feedback

Our approach is highly related to the general area of learning from human feedback. In reinforcement learning, some work has investigated the effect of additional information provided by a guiding human. Specifically, Chernova and Veloso [21] consider the inclusion of a guidance period after a robot's action which can restrict the set of actions that the robot can take in the next step to improve the efficiency of the learning process. Moreira et al. [61] apply this method in the context of deep reinforcement learning to expedite the learning process of a deployed system in a new environment. Similarly, Rosenstein and Barto [74] propose a generalization to the actor-critic reinforcement learning framework [3] that includes a supervisor who can provide additional feedback to the system in the form of auxiliary guiding rewards, action selection guidance, or even direct control of the system. These differ from our work in that we assume that the agent has access to a well-defined and fully-specified model of its domain, including the reward (or cost) function from which to compute an optimal policy, and hence we are not concerned with learning a better world model online (rather, we are only concerned with learning the system's competence model online).

On the other hand, Knox et al. [50, 51] proposed a framework for training a robot *solely* from human feedback (sometimes called interactive shaping or interactive reinforcement learning) in which the human supervising the robot provides real-valued rewards for the actions that were just taken by the robot in a way that is assumed to account for the long-term impacts of the action. However, in our work we are not training the agent to act by learning a reward function, but rather providing the agent labeled data from which it can compute a distribution that is integrated into an explicit transition function. Additionally, we do not consider the use of real-valued feedback from the human, but rather discrete information tokens. More similar to our learning setting, Griffith et al. [41] proposed an approach in which the agent learns two policies in parallel, one derived from reward signals from the environment, and one derived from "right/wrong" labels from the human in order to infer what the *human* believes is the optimal policy, and then combines the two policies into one that is used for action exploitation. The key difference from Griffith et al. [41] is that we seek to learn a predictive model of the human's feedback rather than what the human believes the correct policy to be, and then use this predictive model to analytically determine the optimal policy given the domain model.

Finally, Ramakrishnan et al. [70] examined a problem similar to what we consider in Section 5, wherein an autonomous agent trained in simulation may have "blind spots" when deployed in real-world environments driven by missing or ignoring features that are important in the real-world. Similar to how our method exploits human feedback to identify new features that the human is using in generating their feedback, their method applies imitation learning to demonstrations collected from the human to identify features used by the human but not by the agent. Our work differs primarily in the type of information that the human provides to the system as well as how the missing features are used. We integrate them into the existing model to improve the accuracy of the predicted human feedback which consequently improves the quality of the overall policies generated by the system. On the other hand, [70] use the learned information to learn blind spot models in the real world to perform safe transfer-of-control to a human when encountering a blind spot to avoid potentially dangerous situations.

2.3. Competence Modeling

The term *competence* has been used widely in the context of intelligent systems. The classification literature, in particular, has often defined the term as some measure of performance based on standard metrics for classification systems on their input space [53], including accuracy estimation [98], potential function estimates [71], Bayes-based confidence measures [47], relative performance to random guessing or otherwise randomized classifiers [95], and probabilistic models [56, 96, 97]. More recently, Platanios et al. [67] defined the competence of a curriculum learner to be the proportion of training data that the learner is allowed to use at any given time based on the difficulty of training samples, and Rabiee et al. [68] proposed competence as a distribution over failure classes that is learned via introspective perception in the context of robotic path-planning. Common across these examples is an evaluative approach to defining competence; that is, competence is a measure of the *performance* of a system or algorithm. Most closely related to the formalization of competence presented in this paper was suggested by Smyth and McKenna [84] who defined the competence of a case-based reasoning (CBR) system as the set of problems that the system can solve successfully. The authors provide a rigorous model and analysis of competence for CBR systems, but the work is highly specific to CBR systems on non-probabilistic domains, and consequently does not apply to stochastic decision-making processes considered in this work. Rather, our aim is to enable a system to handle *all* problems by utilizing the appropriate degree of human assistance to ensure safe operation.

Instead, we borrow insights from the definitions of competence posed in the context of human workers. While many definitions have been proposed over the last several decades [30, 79, 86, 90], they are largely atomistic and lacking a well-defined mathematical representation. Gilbert [39] defined it as a function of the ratio of valuable accomplishments to costly behavior, which while mathematically precise, leaves unaddressed both the relative performative capabilities of different agents with respect to a given task's satisfactory completion, an essential component of competence [42], as well as competence as an indication of authoritative permissibility. However, this definition together with the definition of competency as a performance capability implying performance at *a stated level* [90] inspires our definition formalized in Section 3.4. Intuitively, we propose that the competence of a system, much like that of a human, is the optimal level of autonomy to use conditioned on available resources. For example, we might say that a *competent worker* is one that knows when to perform tasks autonomously, when to ask for help and what type of help to ask for, or when to reach for additional sources of aid and information (e.g., via Internet search) to determine how to solve their task safely and reliably. Note that even human workers, when starting a new job for example, may not initially know their exact competence and instead must learn "on the fly" where and when they should solicit different forms of aid or assistance.

3. Competence-Aware Systems

We start with a description of a general *competence-aware system* that can operate in and plan for multiple *levels of autonomy*. Each level of autonomy is defined by a unique set of constraints on autonomous operation and consists of different forms of human feedback that can be provided to the autonomous agent. To enable the agent to reason about its own competence, it must have access to three different models: a *domain model*, an *autonomy model*, and a *feedback model*. Throughout this section, we use the problem setting in Example 1 as a running example to better illustrate the concepts and terminology that we introduce throughout the paper.

Example 1. *An autonomous vehicle (AV) with a human driver (shown in blue in Figure 2) encounters an obstruction (e.g., a parked truck) blocking its lane on a one-lane road (red). In order to overtake the obstruction, the AV would need to drive around the obstruction necessarily driving through the oncoming traffic's lane. In the oncoming lane, there may or may not be a vehicle (yellow), but while stopped behind the obstruction, the AV cannot detect it. The AV may* `Stop` *to let oncoming traffic go past or see if the obstruction resolves itself (e.g., starts moving again),* `Edge` *into the oncoming lane to gain better visibility without risking crashing, or* `Go` *and begin passing the obstruction through the oncoming lane.*

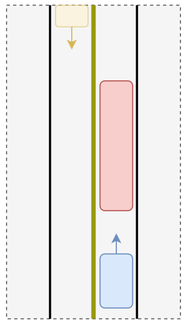


Figure 2: Illustration of Example 1.

3.1. Domain Model

The *domain model* describes the environment in which the agent operates and the dynamics of the agent's actions within that environment. We model this as a *stochastic shortest path* (SSP) problem, a commonly used form of *Markov decision process* (MDP) for reasoning in fully-observable, stochastic environments where the objective is to find the least-cost path from a start state to a goal state [9]. For the purposes of this paper, we consider goal-oriented cost-minimizing problems as they align more naturally with the problem domains that are considered in our experiments. On the other hand, extending the theory to mixed-observable and partially-observable MDPs introduces additional sources of uncertainty, particularly with respect to human interaction, that are non-trivial to handle in our model. A discussion of these challenges can be found later in Section 7.3.

Definition 1. *A* ***domain model****, $\mathcal{D}$, is an SSP represented by the tuple $\langle S, A, T, C, s_0, G\rangle$ where:*

- *S is a finite set of states,*
- *A is a finite set of actions,*
- *$T : S \times A \to \Delta^{|S|}$ is a transition function where $T(s, a)$ describes the probability distribution over successor states when taking an action $a \in A$ in state $s \in S$,*
- *$C : S \times A \to \mathbb{R}^+$ is a cost function where $C(s, a)$ describes the cost of taking action $a \in A$ in state $s \in S$,*
- *$s_0 \in S$ is the initial state, and*
- *$G \subset S$ is the finite set of goal states.*

A solution to an SSP is a mapping $\pi : S \to A$, called a *policy*, that indicates that action $\pi(s)$ is taken by the agent in state s. A policy π induces the state–value function $V^\pi : S \to \mathbb{R}$

$$V^\pi(s) = C(s, \pi(s)) + \sum_{s' \in S} T(s, \pi(s), s')V^\pi(s') \tag{1}$$

which represents the expected cumulative *cost* of reaching any $s_g \in G$ from state $s \in S$ following the policy π. Any policy that minimizes this function is referred to as an optimal policy and denoted π^*; formally:

$$\pi^* := \operatorname*{argmin}_{\pi \in \Pi} V^\pi \tag{2}$$

However, the existence of an optimal solution to the SSP is guaranteed only under the condition that there exists a *proper policy*, i.e. a policy under which a goal state is reachable from all states with probability 1, and that all *improper policies* generate infinite cost when starting from at least one state; under this assumption, the optimal value function is also unique.

	Levels of Autonomy	Human Involvement
l_0	No Autonomy	Human driver fully in control of vehicle.
l_1	Verified Autonomy	Autonomous agent in control of vehicle conditioned on explicit approval from human for maneuver prior to execution.
l_2	Supervised Autonomy	Autonomous agent in control of vehicle conditioned on a human driver supervising the system ready and capable of taking control.
l_3	Unsupervised Autonomy	Autonomous agent in unconditional control of vehicle, *possibly* with (but not requiring) a human who can take over control.

Table 1: Levels of autonomy with $\mathcal{L} = \{l_0, l_1, l_2, l_3\}$ where $l_0 \to l_1 \to l_2 \to l_3$.

3.2. Autonomy Model

The *autonomy model* describes the levels of autonomy that the agent can operate in, restrictions on the situations under which each level is allowed, the utilities of each level, and a set of system sub-competencies.

Definition 2. *An **autonomy model**, $\mathcal{A}$, is represented by the tuple $\langle \mathcal{L}, \kappa, \mu \rangle$ where:*

- *$\mathcal{L}$ is the finite, partially ordered set of levels of autonomy where each level $l \in \mathcal{L}$ corresponds to some set of constraints on the system's autonomy,*
- *$\kappa : S \times \mathcal{L} \times A \to \mathcal{P}(\mathcal{L})$ is the autonomy profile where $\kappa(s, l, a)$ returns the subset of levels of autonomy $L \subseteq \mathcal{L}$ allowed when performing action $a \in A$ in state $s \in S$ given that the agent just acted in level $l \in \mathcal{L}$, and*
- *$\mu : S \times \mathcal{L} \times A \times \mathcal{L} \to \mathbb{R}^+$ is the cost of autonomy where $\mu(s, l, a, l')$ describes the cost of taking action $a \in A$ in level $l' \in \mathcal{L}$ in state $s \in S$ given that the agent just acted in level $l \in \mathcal{L}$.*

While most interpretations of levels of autonomy, as discussed in Section 1, are presented as ordered sets of increasing autonomy, in general this need not be the case. In fact, in some cases different levels of autonomy may be directly compared. Hence, we choose to model ours more generally as a partially ordered set[1] where $l_i \leq l_j$ if and only if, given any task (s_0, G), $V^{l_i}(s_0) \leq V^{l_j}(s_0)$ where V^{l_i} is the value function induced by the optimal policy when the level of autonomy is fixed at l_i. Note that we consider two levels, l_i and l_j, to be *adjacent* if $l_i < l_j \land \nexists l_k \in \mathcal{L} \,|\, l_i < l_k < l_j$. The constraints corresponding to each level of autonomy can be technical in nature, i.e., internally imposed constraints such as requiring human supervision in poor weather conditions that may be known *a priori* to cause errors, as well as externally imposed constraints such as ethical or legal requirements. Each constraint is associated with a corresponding form of human assistance or involvement. Intuitively, the higher the level of autonomy, the lower the cost of human involvement, although this is not a requirement of the model. An example of a set of levels of autonomy can be seen in Table 1.

Additionally, κ can be defined to not only reflect hard constraints such as ethical, legal, or technical constraints [40, 55, 57, 88] that are fixed throughout the system's deployment, but also tentative constraints that can be updated over time. Tentative constraints allow for a period of learning or adjustment early in the deployment of the system as the human familiarizes themselves with the system, or the system learns to act appropriately in its environment. An example of different constraints on autonomy can be seen in Table 2.

The cost of autonomy, μ, is the cost associated with the act of operating in a given level of autonomy and is distinct from the base domain cost of the action's execution. For example, in a level of autonomy that requires tele-operation from an off-site human to provide verification to a waiting autonomous vehicle, there may be an additional cost of operating in that level corresponding to the amount of time waiting to reach an available tele-operator and receive feedback. In a system with a finite energy supply that can perform sensing and perception at different levels of fidelity (corresponding to different levels of autonomy), each level may utilize a different amount of energy.

[1] $\mathcal{L}$ could be structured as a polytree or an arbitrary directed acyclic graph, however, for the sake of clarity we do not consider such levels of autonomy in this paper.

	Constraints on Autonomy
Ethical	The AV may not be allowed to initiate a transfer of control to a human that is drowsy or otherwise deemed unfit to operate the vehicle safely.
Legal	The AV may not be allowed to operate autonomously inside of a school zone.
Technical	The AV may be disallowed from operating autonomously in snowy weather due to the interference of perception and object detection systems.
Tentative	The AV may be initialized to drive in l_1 when it has no visibility, but may learn to perform the action Edge in l_3 as it introduces an allowable level of risk by the human in the car.

Table 2: Examples of different types of constraints on autonomy.

3.3. Feedback Model

The *feedback model* describes the agent's knowledge about and predictions of its interactions with the human, including the types of feedback it can receive from the human, how likely each possible type of feedback is at any given time, and the expected cost to the human for assisting the agent.

Definition 3. *A **feedback model**, $\mathcal{F}$, is represented by the tuple $\langle \Sigma, \lambda, \rho, \tau_{\mathcal{H}} \rangle$, where:*

- *Σ is the finite set of feedback signals that the agent can receive from the human,*
- *$\lambda : S \times \mathcal{L} \times A \times \mathcal{L} \to \Delta^{|\Sigma|}$ is the feedback profile where $\lambda(s, l, a, l')$ represents the probability distribution over feedback signals that the agent will receive when performing action $a \in A$ in level $l' \in \mathcal{L}$ in state $s \in S$ given that the agent just operated in level $l \in \mathcal{L}$,*
- *$\rho : S \times \mathcal{L} \times A \times \mathcal{L} \to \mathbb{R}^+$ is the human cost function where $\rho(s, l, a, l')$ represents the cost to the human when the agent performs action $a \in A$ in level $l' \in \mathcal{L}$ in state $s \in S$ given that the agent just operated in level $l \in \mathcal{L}$, and*
- *$\tau_{\mathcal{H}} : S \times A \to \Delta^{|S|}$ is the human state transition function where $\tau_{\mathcal{H}}(s, a)$ represents the probability distribution over successors states $s' \in S$ when the human takes control of the system when the agent attempts to perform action $a \in A$ in state $s \in S$.*

Although there are many forms of human feedback that have been studied, we limit our focus specifically to *feedback signals* which are represented as discrete tokens of feedback that the human can provide to the autonomous agent, either implicitly (e.g. facial gestures or body posture), or explicitly (e.g., verbal responses or physical control), as opposed to real-valued reward signals [50, 51] or full demonstrations [24, 70, 72]. The primary reason is to keep the feedback signals semantically simple in the sense that they are represented compactly by the system while still being easily and unambiguously associated with the human's intentions. This reduces the overhead associated with the human-agent interactions. Each feedback signal may be associated with a distinct level, or subset of levels, of autonomy and a corresponding form of human involvement. An example of this can be seen in Table 3. Future directions of research may investigate extending these feedback signals to address such questions as how to learn from feedback when there is a *degree* of severity associated with it, how to handle *proactive feedback* which is intended by the human to be for inferred future states or trajectories, or feedback in the form of direct action commands.

The human cost function, ρ, is the cost *to the human* when operating in a given level and hence is separate from the costs incurred directly by the autonomous agent. This cost may often be related to the human's opportunity cost for being unable to engage in other activities while assisting the autonomous agent. However, it may additionally capture other costs to the human, such as additional stress or work added to them in addition to the time they spend assisting (assisting two different actions which take the same time may require different levels of exertion from the human, for example supervising an autonomous action making a left turn, or manually making the left turn). In practice, the human's cost function may be non-Markovian; for instance becoming fatigued after repeatedly performing manual control, or becoming frustrated after extended periods of oscillating between different levels of autonomy, constantly shifting the demand on the human. While this can be coarsely approximated by conditioning the cost on the previous

	Feedback Signal	Interaction	Levels of Autonomy
$\emptyset$	No feedback	N/A	$\{l_0, l_2, l_3\}$
$\oplus$	Approval	Verbal or Tactile Response	$\{l_1\}$
$\ominus$	Disapproval	Verbal or Tactile Response	$\{l_1\}$
$\oslash$	Override	Arrested Control	$\{l_2, l_3\}$

Table 3: Each feedback signal is provided via a fixed and known interaction; for instance, the feedback signal *approval* may be provided either by a verbal "Yes" from the human, or via a tactile response such as pressing a button on a touchscreen, similarly for *disapproval*. *Override* may be recognized by any form of arrested control by the human during autonomous operation, for instance braking, accelerating, or steering while the AV is in control. Each signal is only recognized when the AV is operating at the corresponding level of autonomy.

level of autonomy (as done here), one can improve this by maintaining a model of the human's state, similar to what is done by Costen et al. [26].

If λ and $\tau_{\mathcal{H}}$ are known exactly *a priori* then the system's true competence (Definition 10) can be immediately computed exactly under any κ, and the problem reduces to a straightforward planning problem. Furthermore, in some problem instances where the feedback model is known exactly there may be no need to even constrain the policy space at all (i.e. $\kappa(s, a) = \mathcal{L}$ for every $(s, a) \in S \times A$). This is the case when the feedback mechanisms are sufficient to prevent the agent from taking actions that would violate hard constraint; for example, if the human authority always overrides an action at a level that would violate an ethical, legal, or technical constraint. This introduces a trade-off in distributing the burden of effort between the designers of the system and the operator of the system to ensure safe and reliable operation in all cases.

However, in this work we are primarily concerned with systems where λ and $\tau_{\mathcal{H}}$, and by consequence the system's true competence, are unknown *a priori*. In this case, they must be estimated by functions $\hat{\lambda}$ and $\hat{\tau}_{\mathcal{H}}$, which are based on observed data collected online through interactions with the human at various levels of autonomy that can generate feedback signals. These feedback signals can be analogously treated as labels in a labeled data set where the data is the state, action, and level that generated the feedback signal. In Section 5, we address situations where the human's model of the world does not align with that of the autonomous agent, leading to feedback that is poorly discriminated by the agent, which reduces its ability to learn from the signals it receives from the human.

Note that, in many real-world problems, the process of acquiring feedback signals may not be instantaneous, and in some cases could require a complex process of fully or partially transferring control to and from a human over an indefinite amount of time, where each element of the transfer process, such as the communication interface, is important. The problem of transfer of control in semi-autonomous systems has been separately studied [81, 99]; however, for the sake of clarity, we do not model this process explicitly in this work as we focus on the orthogonal problem of modeling levels of autonomy and competence.

3.4. Competence-Aware Systems

A *competence-aware system* (CAS) represents a planning problem that accounts for the different levels of autonomy available to the agent and factors in the agent's expectations regarding the likelihood and cost of human feedback (e.g., assistance, queries, intervention, etc.). The objective of a solution to a CAS planning problem is to create a plan that best balances the cost of reaching the goal with the cost of human assistance to achieve the most cost-effective strategy given the constraints of the problem. Hence, the CAS uses the autonomy model to proactively generate plans that operate across multiple levels of autonomy by leveraging the feedback model to predict the likelihood of different feedback signals in order to optimize the level of autonomy and minimize the reliance on humans. To this end, we represent a CAS as a *multi-objective* planning problem.

Example 2. *A competence-aware system with four levels of autonomy—verified, supervised, unsupervised, and no autonomy—and four type of feedback signals—approval, disapproval, override, and no feedback. The policy, π, constrained by the autonomy profile κ, produces an action a at a level l to be performed in state $\bar{s}$. The level l determines the execution process of the action a, as depicted in the lower section of the figure. Certain levels may*

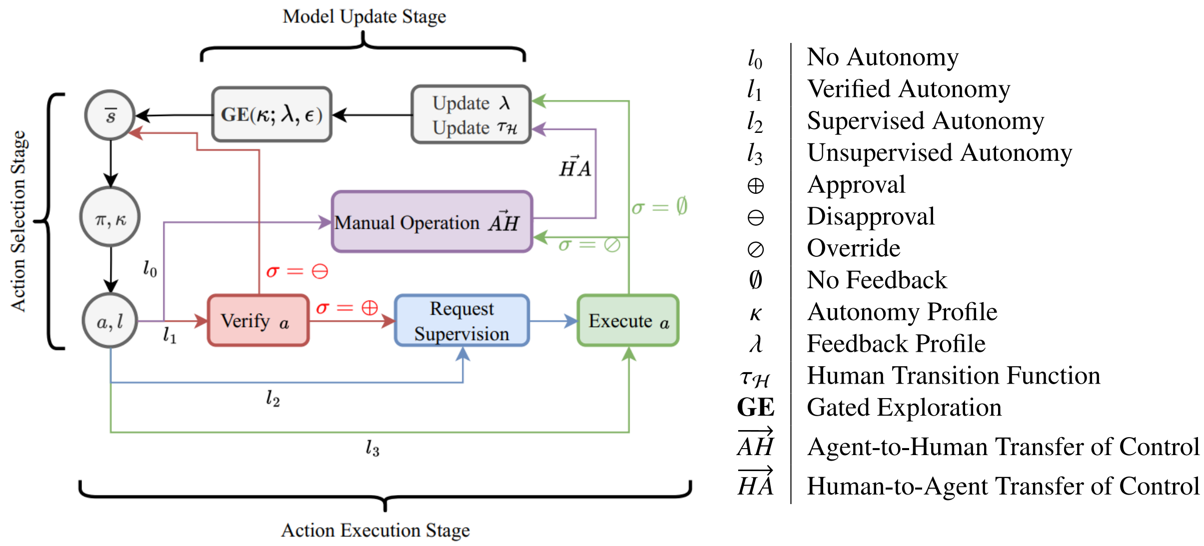


Figure 3: Illustration of Example 2

prompt the human for feedback, with a possibility of complete transfer of control from the autonomous agent to the human. After the action is executed and data is collected, internal model parameters, λ and $\tau_{\mathcal{H}}$, are updated. Finally, the agent may perform gated exploration (Definition 8) to update the autonomy profile κ, although in practice this would be performed on a less frequent basis.

Definition 4. *A **competence-aware system** $\mathcal{S}$ is represented by the tuple $\langle \overline{S}, \overline{A}, \overline{T}, \overline{C}, \overline{s}_0, \overline{G} \rangle$, where:*

- *$\overline{S} = S \times \mathcal{L}$ is a set of factored states, each comprised of a domain state $s \in S$ and a level of autonomy $l \in \mathcal{L}$.*
- *$\overline{A} = A \times \mathcal{L}$ is a set of factored actions, each comprised of a domain action $a \in A$ and a level of autonomy $l \in \mathcal{L}$.*
- *$\overline{T} : \overline{S} \times \overline{A} \to \Delta^{|\overline{S}|}$ is a transition function where $\overline{T}(\overline{s}, \overline{a})$ represents the distribution over successor states when taking action $\overline{a} \in \overline{A}$ in state $\overline{s} \in \overline{S}$.*
- *$\overline{C} = \begin{bmatrix} C & \mu & \rho \end{bmatrix}^T$ is a vector of cost functions.*
- *$\overline{s}_0 \in \overline{S}$ is the initial state where $\overline{s}_0 = \langle s_0, l \rangle$ for some $l \in \mathcal{L}$.*
- *$\overline{G} \subset \overline{S}$ is the set of goal states.*

A CAS state $\overline{s} \in \overline{S}$ represents the CAS's current domain state s and the level of autonomy, l, that the CAS performed its last action in. The purpose of including the previous level of autonomy in the state representation is to capture the fact that human feedback can vary depending on the level of autonomy that the agent was just operating in (for instance, a human may be less likely to override the system if they were previously engaged in supervising the system); additionally, we may want to discourage the system from oscillating between levels of autonomy by imposing a small cost every time the system changes levels. Note that, one can set $\overline{G} = \hat{S} \times \mathcal{L}$ for some $\hat{S} \subseteq S$ to indicate that the level of autonomy does not impact the goal condition or state, for instance setting $\overline{G} = G \times \mathcal{L}$.

A CAS action $\overline{a} \in \overline{A}$ represents a domain action a to be performed at a given level of autonomy l which may alter both the mechanics of how the action is executed, the form and degree of involvement by the human authority in the execution of the action, and the types of feedback that the agent can receive from the human authority.

$\overline{T}$ is a transition function that represents the probability distribution over both how the state will change and which feedback signal, if any, the agent will receive from the human when performing an action conditioned on the level the action is being performed in, the current state, and the previous level that the agent had operated in (i.e. the timestep prior to the current one). For example, the likelihood of a human override may decrease if the system had already been acting under supervision than if they had been acting without supervision, as the human may have a better understanding of what the system is doing.

Example 3. *Given $\mathcal{L}$ and Σ, we can specify the* ***state transition function*** *of this CAS. Given $\overline{s} = (s, l)$, $\overline{s'} = (s', l')$, and $\overline{a} = (a, l')$, we define $\overline{T}$ as follows:*

$$\overline{T}(\overline{s}, \overline{a}, \overline{s}') = \begin{cases} \tau_{\mathcal{H}}(s, a, s'), & \text{if } l = l_0, \\ \lambda(\oplus|\overline{s}, \overline{a})\overline{T}(\overline{s}, (a, l_2), \overline{s'}) + \lambda(\ominus|\overline{s}, \overline{a})[s = s'], & \text{if } l = l_1, \\ \lambda(\emptyset|\overline{s}, \overline{a})T(s, a, s') + \lambda(\oslash|\overline{s}, \overline{a})\tau_{\mathcal{H}}(s, a, s'), & \text{if } l \in \{l_2, l_3\}, \end{cases} \tag{3}$$

where $[\cdot]$ denotes Iverson brackets. Intuitively, Equation 3 states that when the agent operates in l_0, it follows the transition dynamics of the human who takes control. When operating in l_1, the probability it arrives in state s' is the probability it is approved to take the action times the probability of the state change following $\overline{T}$ under level l_2, plus the probability that it is disapproved and the state is the same. In levels l_2 and l_3, the probability it arrives in state s' is the probability it succeeds following T without any human intervention plus the probability that the human overrides it and takes it to that state. In general, we expect the probability of an override to be lower (or even 0) in l_3 as supervision is not required.

A solution to a given CAS is a policy π that maps states and levels $\overline{s} \in \overline{S}$ to actions and levels $\overline{a} \in \overline{A}$. Multi-objective decision making has been well-studied [73], and for our purposes we assume a scalarized approach [73] with a scalarization function f parameterized by a weight vector $\mathbf{w}$. A common approach is simply based on a linear combination of the cost functions in $\overline{C}$, e.g., $\overline{C} = \mathbf{w}\begin{bmatrix} C & \mu & \rho \end{bmatrix}^T$. With some modifications, the problem could be extended to handle both lexicographic models [100] and constrained models [2]. However, the properties that we derive for the scalarized model may not necessarily hold for arbitrary multi-objective models, and would need to be re-examined in those contexts. Additionally, we restrict the CAS to only consider policies that are allowed under the autonomy profile κ in the following way.

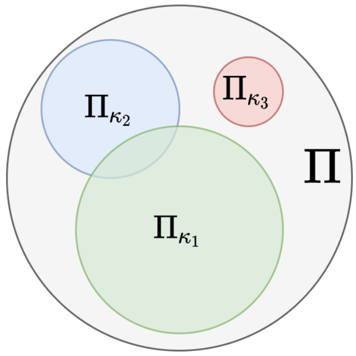


Figure 4: Illustration of a policy space, Π, constrained by three different autonomy profiles, κ_1, κ_2, and κ_3.

Definition 5. *Let $\overline{a} = \langle a, l \rangle$. Given $\overline{s} = \langle s, l' \rangle \in \overline{S}$, we say that $(\overline{s}, \overline{a})$ is* allowed *if $l \in \kappa(s, a)$, and a policy π is* allowed *if for every $\overline{s} \in \overline{S}$, $(\overline{s}, \pi(\overline{s}))$ is allowed.*

We denote the set of allowable policies given κ as Π_κ and require that the policy returned by solving the CAS, π^*, is always taken from $\text{argmin}_{\pi \in \Pi_\kappa} V^\pi(s_0)$. An illustration of how different autonomy profiles can constrain the full policy space, Π, can be seen in Figure 4.

In general, a competence-aware system planning model is not guaranteed to be a valid stochastic shortest path problem (see Proposition 1) due to the possible effects that κ and λ can have on the existence of a proper policy, although in some cases they may only induce dead-ends away from the initial state for which there is existing work on how to handle [52]. However, one can ensure that there is a proper policy with the inclusion of a level of autonomy with a property similar to level l_0 in Table 1 which allows for (at potentially high cost) the deterministic completion of any action or task, guaranteeing the existence of a proper policy. Note that we do not need to worry about the possibility of ρ or μ inducing zero-cost cycles as they are non-negative cost functions, and the domain model is, by assumption, a valid SSP.

4. Properties of a Competence-Aware System

In this section, we will discuss the central properties of a CAS that will allow us to prove several key results of competence-aware systems. Henceforth, we will assume that there exists a singular human authority that the semi-autonomous system in a CAS interacts with, and we will use the notation $\mathcal{H}$ to refer to them.

Definition 6. *The* human authority, *$\mathcal{H}$ is represented by the tuple $\langle F^{\mathcal{H}}, \lambda^{\mathcal{H}}, \kappa^{H} \rangle$ where:*

- *$F^{\mathcal{H}}$ is the set of features used by $\mathcal{H}$ when providing feedback,*
- *$\lambda^{\mathcal{H}} : \overline{S} \times \overline{A} \rightarrow \Delta^{|\Sigma|}$ is a stationary distribution of feedback signals that $\mathcal{H}$ follows, and*
- *$\kappa^{\mathcal{H}} : \overline{S} \times A \rightarrow \mathcal{P}(\mathcal{L})$ is the fixed mapping from state-action pairs to sets of autonomy levels that $\mathcal{H}$ will allow the autonomous agent to operate in with nonzero probability.*

Intuitively, $\kappa^{\mathcal{H}}$ represents the human authority's belief of the agent's competence; by definition any level not contained in the image of $\kappa^{\mathcal{H}}$ will never be allowed by $\mathcal{H}$.

First, we begin with a simple proof that a CAS model is, in general, not guaranteed to be a valid stochastic shortest path problem due to the lack of a proper policy.

Proposition 1. *There exists a competence-aware system $\mathcal{S}$ that does not admit a proper policy.*

Proof. Let $\mathcal{S}$ be a CAS with exactly one level of autonomy, l, where the level of autonomy works as follows: when the agent attempts to execute action a, they must first query the human to obtain a binary `yes` or `no` feedback signal. If the signal is `yes` then the agent may attempt to execute the action according to its model. If the signal is `no` then the agent may not attempt to execute the action in its current state. Let $(s_0, l) \in \overline{S}$ denote the initial state and assume $(s_0, l) \notin \overline{G}$, where $\overline{S}$ is the state space of $\mathcal{S}$ and $\overline{G}$ is the set of goals. Let $\lambda^{\mathcal{H}}(\texttt{yes}|(s_0, l), (a, l)) = 0.0$ for every action $a \in A$ (where A is the action set). As the agent will never be able to transition out of its state which is not a goal state by assumption, it is clear that there exists no proper policy. □

Second, a fundamental component of the CAS model is the ability to adjust its autonomy profile over time using what it has learned in order to optimize its autonomy by reducing unnecessary reliance on human assistance. However, before operating in a new level of autonomy, the system may have no knowledge of how the human will interact with it in that level, i.e., the feedback profile in that new level may be initialized by default to some baseline distribution. As a result it is necessary that the system *explore* levels of autonomy that it predicts are more cost effective than its current allowed levels, so that it may learn whether or not it is competent to act in those levels.

Allowing the system to alter its own autonomy profile, however, can lead to severe consequences in the real world if not done carefully, mitigating the risk-awareness we aim to endow via the competence modeling. Therefore, we propose two notions to ensure a measure of safety and risk-sensitivity in a competence-aware system. The first is *level-safety* which is a notion of the safety of the level of autonomy that the system is using and is conditioned on both the agent and the human; intuitively, a CAS is level-safe if it cannot act in levels that the human authority would not allow. Second is *gated exploration* which is a simply extension to standard exploration methods used in reinforcement learning in which the system must obtain permission from a human before exploring a new (disallowed) level of autonomy, ensuring that level-safety is never violated.

Example 4. *An autonomous vehicle is initialized to only use levels $\{l_0, l_1, l_2\}$ when executing the overtaking maneuver, but learns that there is a very low likelihood of an override by the human authority during the day with clear visibility and sparse traffic. Hence, it expects based on estimated costs that its competence is in fact l_3 which is initially disallowed to ensure safety at initial deployment. It therefore queries the human to approve it to update its autonomy profile κ by adding level l_3 under the stated conditions.*

Definition 7. *A CAS $\mathcal{S}$ is* level-safe *under κ if $\kappa(\overline{s}, a) \subseteq \kappa^{\mathcal{H}}(\overline{s}, a)$ for every $(\overline{s}, a) \in \overline{S} \times A$.*

Definition 8. *We define the* gated-exploration *strategy for $(\overline{s}, a) \in \overline{S} \times A$ as follows: let $adj(l, l') = 1$ if $l = l'$ or l and l' are adjacent in $\mathcal{L}$ and 0 otherwise, and let $adj(\kappa(\overline{s}, a), l') = 1$ if $l' \in \kappa(\overline{s}, a)$ or $adj(l, l') == 1$ for some $l \in \kappa(\overline{s}, a)$. Let $P_l(\mathcal{L})$ be a distribution over $\mathcal{L}$ such that $P_l(l') = 0$ if $adj(l, l') == 0$, and let $l^* \sim P_l(\mathcal{L})$. If $l^* \in \kappa(\overline{s}, a)$ do nothing, otherwise, query the human authority $\mathcal{H}$ to allow for the level exploration. If the query returns a positive response, set $\kappa(\overline{s}, a) \leftarrow \kappa(\overline{s}, a) \cup \{l^*\}$, and otherwise do nothing.*

Proposition 2. *Let $\mathcal{S}$ be a CAS with initial autonomy profile κ_0. If $\mathcal{S}$ is level-safe under κ_0 and follows the gated-exploration strategy, then $\mathcal{S}$ will be level-safe under κ_t for any $t \geq 0$.*

Proof. This is straightforward to observe by applications of the definitions. If $\mathcal{S}$ is level-safe under κ_0, then for all $(\bar{s}, a) \in \bar{S} \times A$, $\kappa_0(\bar{s}, a) \subseteq \kappa^{\mathcal{H}}(\bar{s}, a)$ by definition. If there exists $t > 0$ for which $\kappa_t(\bar{s}, a) \neq \kappa_0(\bar{s}, a)$ for some $(\bar{s}, a) \in \bar{S} \times A$, then there is some $l^* \in \kappa_t(\bar{s}, a) \setminus \kappa_0(\bar{s}, a)$. By the definition of gated exploration and $\kappa^{\mathcal{H}}$, it must be that $l^* \in \kappa^{\mathcal{H}}(\bar{s}, a)$, and hence $\kappa_t(\bar{s}, a) \subseteq \kappa^{\mathcal{H}}(\bar{s}, a)$. As $(\bar{s}, a)$ is arbitrary, this holds for all $(\bar{s}, a) \in \bar{S} \times A$, and hence $\mathcal{S}$ is level-safe. □

Next, we introduce a notion of *feedback consistency* which is a property of how consistent the human authority is in providing the same feedback given the same query by the acting agent.

Definition 9. *Let $F^{\mathcal{H}} = \{F_1^{\mathcal{H}}, ..., F_n^{\mathcal{H}}\}$ be the set of features used by the human authority, $\mathcal{H}$, and let $\bar{S}_{\mathcal{H}} = F_1^{\mathcal{H}} \times \cdots \times F_n^{\mathcal{H}} \times \mathcal{L}$. The **ground truth feedback function** is a deterministic mapping $f : \bar{S}_{\mathcal{H}} \times \bar{A} \to \Sigma$. $\mathcal{H}$ is **perfectly consistent** if $\lambda^{\mathcal{H}}(f(\bar{s}, \bar{a})|\bar{s}, \bar{a}) = 1$ $\forall \bar{s} \in \bar{S}, \bar{a} \in \bar{A}$. If $\lambda^{\mathcal{H}}(f(\bar{s}, \bar{a})|\bar{s}, \bar{a}) \geq \epsilon$ for $\epsilon \in (0, 1)$ $\forall \bar{s} \in \bar{S}, \bar{a} \in \bar{A}$, then $\mathcal{H}$ is **ϵ-consistent**.*

Unless otherwise stated, we assume that the human authority is ϵ–consistent henceforth. We now define three central properties of a CAS.

Definition 10. *Let $\lambda^{\mathcal{H}}$ be the stationary distribution of feedback signals that the human authority follows. The **competence** of CAS $\mathcal{S}$, denoted $\chi_{\mathcal{S}}$, is a mapping from $\bar{S} \times A$ to the optimal (least-cost) level of autonomy given perfect knowledge of $\lambda^{\mathcal{H}}$. Formally:*

$$\chi_{\mathcal{S}}(\bar{s}, a) = \operatorname*{argmin}_{l \in \mathcal{L}} q^*(\bar{s}, (a, l); \lambda^{\mathcal{H}}) \tag{4}$$

where $q^*(\bar{s}, (a, l); \lambda^{\mathcal{H}})$ is the cumulative expected cost under the optimal policy π^* when taking action $\bar{a} = (a, l)$ in state $\bar{s}$ conditioned on the human authority's feedback distribution, $\lambda^{\mathcal{H}}$.

Fundamentally, the system's competence for executing action a in state $\bar{s}$, $\chi_{\mathcal{S}}(\bar{s}, a)$, is the most beneficial (e.g. cost effective) level of autonomy were it to know the true human feedback distribution. When $\mathcal{L}$ is an ordered set, we expect this to generally be the highest level of autonomy *allowed* by the human; however, this need not be the case. In principle, the highest allowed level of autonomy could require more frequent human interventions, e.g. due to lower levels of trust by the human in the system [44], that may render it less efficient overall relative to a lower level of autonomy.

It is important to note that this definition of competence relies on $\lambda^{\mathcal{H}}$, and hence is a definition of competence on *the overall human-agent system*, and is explicitly not just a measure of the underlying agent's technical capabilities (i.e. $\mathcal{D}$). A corollary of this fact is that the CAS is only as competent as the human authority believes it to be; a human authority that has a poor understanding of the system's abilities could lead to the system having a lower competence than a human authority that knows perfectly the limitations and capabilities of the system. One reason for modeling *competence* in this manner is to avoid relying on arbitrary thresholding based on evaluative metrics to determine when a system is competent or not.

We say that a CAS $\mathcal{S}$ is λ-stationary if, in expectation, any new feedback drawn from the true distribution $\lambda^{\mathcal{H}}$ will not affect λ enough to change the optimal level of autonomy for any $\bar{s} \in \bar{S}$ and $a \in A$. We show below that, under standard assumptions, $\mathcal{S}$ will converge to λ-stationarity.

Definition 11. *Let $\mathcal{S}$ be a CAS and let $U(\lambda)$ be the q-value of $(\bar{s}, a)$ under the optimal policy given λ where $\mathcal{S}$ executed the action a in level l in state $\bar{s}$. We define the expected value of sample information (EVSI) on $\sigma \in \Sigma$ for $(\bar{s}, a)$ to be:*

$$\sum_{\sigma \in \Sigma} \max_{l \in L} \int_{\Lambda} U(l, \lambda) \lambda(\sigma | \bar{s}, a, l) p(\lambda) d\lambda - \max_{l \in L} \int_{\Lambda} U(l, \lambda) p(\lambda) d\lambda. \tag{5}$$

Definition 12. *Let $\mathcal{S}$ be a CAS. $\mathcal{S}$ is **λ-stationary** if for every state $\bar{s} = (s, l) \in \bar{S}$, and every action $a \in A$, the expected value of sample information on $\sigma \in \Sigma$ for $(\bar{s}, a)$ (Eq. 5) is less than ϵ for any ϵ greater than 0.*

Proposition 3. *Let $\lambda_t^{\bar{s},a}$ be the random variable representing $\lambda(\bar{s}, a)$ after having received t feedback signals for $(\bar{s}, a)$ where each signal is sampled from the true distribution $\lambda^{\mathcal{H}}(\bar{s}, a)$. Then, as $t \to \infty$, the sequence $\{\lambda_t^{\bar{s},a}\}$ converges in distribution to $\lambda_{\mathcal{H}}^{\bar{s},a} = \mathbb{E}[\lambda^{\mathcal{H}}(\bar{s}, a)]$.*

Proof. As each signal is drawn from $\lambda^{\mathcal{H}}(\bar{s}, a)$ i.i.d, then by a straightforward application of the law of large numbers the sequence will converge in probability to $\lambda_{\mathcal{H}}^{\bar{s},a}$, which directly implies the claim. □

Theorem 1. *Let $\mathcal{S}$ be a CAS, and let $\lambda_t^{\bar{s},a}$ be the random variable representing $\lambda(\bar{s}, a)$ after having received t feedback signals for $(\bar{s}, a)$ where each signal is sampled from the true distribution $\lambda^{\mathcal{H}}(\bar{s}, a)$. As $t \to \infty$, if no $(\bar{s}, a)$ is starved, $\mathcal{S}$ will converge to λ-stationarity.*

Proof. Let $\bar{s} \in \bar{S}$ and $a \in A$. As $\bar{s}$ and a are arbitrary and we assume that no $(\bar{s}, a)$ is starved, it is sufficient to show convergence to stationarity for $(\bar{s}, a)$ as $t \to \infty$. By Proposition 3, $\{\lambda_t^{\bar{s},a}\}$ will converge to $\lambda_{\mathcal{H}}^{\bar{s},a}$ in distribution given our assumptions. Because $\{\lambda_t^{\bar{s},a}\}$ converges in distribution, $\lim_{t\to\infty} Pr(|\lambda_t^{\bar{s},a} - \lambda_{\mathcal{H}}^{\bar{s},a}| > \epsilon) = 0 \ \forall \epsilon > 0$. Therefore, in the limit the probability that $\lambda = \lambda_{\mathcal{H}}^{\bar{s},a}$ after t steps, $p_t(\lambda)$, defines a Dirac delta function with point mass centered at $\lambda^{\mathcal{H}}$. Hence we get that, $\lim_{t\to\infty}$ EVSI (Eq. 5)

$$
\begin{aligned}
&= \Big(\lim_{t\to\infty} \sum_{\sigma\in\Sigma} \max_{l\in L} \int_\Lambda U(\lambda, l)\lambda(\sigma|s, \emptyset, a, l)p_t(\lambda)d\lambda\Big) - \Big(\lim_{t\to\infty} \max_{l\in L} \int_\Lambda U(\lambda, l)p_t(\lambda)d\lambda\Big) \\
&= \Big(\sum_{\sigma\in\sigma} \max_{l\in L} U(\lambda^{\mathcal{H}}, l)\lambda^{\mathcal{H}}(\sigma|s, \emptyset, a, l)\Big) - \Big(\max_{l\in L} U(\lambda^{\mathcal{H}}, l)\Big) \\
&= \sum_{\sigma\in\Sigma} \max_{l\in L} U(\lambda^{\mathcal{H}}, l)(1 - \lambda^{\mathcal{H}}(\sigma|s, \emptyset, a, l)) \\
&= \max_{l\in L} U(\lambda^{\mathcal{H}}, l)\Big(1 - \sum_{\sigma\in\Sigma} \lambda^{\mathcal{H}}(\sigma|s, \emptyset, a, l)\Big) \\
&= \max_{l\in L} U(\lambda^{\mathcal{H}}, l)(1 - 1) \\
&= 0.
\end{aligned}
$$

□

Second, we say that a CAS $\mathcal{S}$ is *level-optimal* in some state if, under its current optimal policy, the action it takes in that state is performed at its competence for that state-action pair.

Definition 13. *Let $\mathcal{S}$ be a CAS. $\mathcal{S}$ is **level-optimal in state $\bar{s}$** if*

$$\pi^*(\bar{s}) = (a, \chi_{\mathcal{S}}(\bar{s}, a)) \tag{6}$$

*If this holds for all states we say that $\mathcal{S}$ is **level-optimal**. Similarly, $\mathcal{S}$ is **γ-level-optimal** if this holds in $\gamma|\bar{S}|$ states for $\gamma \in (0, 1)$.*

The primary goal of a competence-aware system is to *reach level-optimality while maintaining level-safety*. As we have already shown that a CAS will maintain level-safety under the gated-exploration strategy (given an initial, level-safe autonomy profile), we therefore want to show that under certain conditions, a competence-aware system $\mathcal{S}$ will be guaranteed to reach level-optimality. In other words, that the system is guaranteed to reach a point where it operates at its competence in all situations.

To prove that a competence-aware system will reach level-optimality, we rely on the notion of *gated exploration* as detailed in Definition 8. However, we also require the following *exploitation* approach: if $\mathcal{S}$ has reached λ-stationarity then it no longer explores under the exploration strategy and instead exploits its knowledge by deterministically selecting the optimal level of autonomy at that point, i.e. for any given $(\bar{s}, a) \in \bar{S} \times A$, the system will use a level $l \in \text{argmin}_{l\in\kappa(\bar{s},a)} q(\bar{s}, (a, l); \hat{\lambda})$. However, as the theory only proves convergence to λ-stationarity (that is, an expected value of sample information of 0 over all $\sigma \in \Sigma$ for every $(\bar{s}, a) \in \bar{S} \times A$) in the *limit*, we instead simply require that for any fixed $z \in \mathbb{R}^+$, sufficiently small, the system will switch to exploitation once the expected value of sample information falls below z everywhere which will happen in finite time. We will refer to this below as *exploitation under stationarity*.

Definition 14. *Let $\mathcal{S}$ be a CAS, and let κ_t represent the autonomy profile κ at time t. Given $\bar{s} \in \bar{S}$ and $a \in A$, we say that $l \in \mathcal{L}$ is reachable from κ_t for $(\bar{s}, a)$ if there exists at least one path from $\kappa_t(\bar{s}, a)$ to $l \in \mathcal{L}$, where all levels along the path are in $\kappa^{\mathcal{H}}(\bar{s}, a)$.*

In the following text, let κ_t refer to the autonomy profile, κ, after the t^{th} feedback signal has been received.

	Human 1			Human 2		
$\hat{F}$	σ_1	σ_2	σ_3	σ_1	σ_2	σ_3
f_1	0.171	-0.146	-0.055	0.222	0.255	-0.410
f_2	0.293	-0.158	-0.209	-0.037	-0.109	0.111
f_3	-0.399	0.267	0.220	-0.212	-0.197	0.361
f_4	0.375	-0.335	-0.103	0.384	-0.170	-0.313
f_5	-0.379	0.257	0.205	-0.372	0.311	0.208
f_6	0.064	0.043	-0.141	0.045	-0.183	0.069
f_7	-0.030	0.118	-0.104	0.044	-0.019	-0.036
f_8	0.179	-0.110	-0.112	0.044	-0.019	-0.036
f_9	0.085	-0.093	-0.002	-0.062	0.027	0.051
f_{10}	0.108	-0.151	0.038	-0.237	0.104	0.193
f_{11}	0.175	-0.059	-0.168	0.325	0.295	-0.549

Table 5: The correlation matrices of each override signal with each feature.

Theorem 2. *Let $\mathcal{S}$ be a CAS that follows the gated exploration strategy and performs exploitation under stationarity, where $\chi_{\mathcal{S}}(\overline{s}, a)$ is reachable from κ_0 for all $(\overline{s}, a) \in \overline{S} \times A$. Then if no $(\overline{s}, a)$ is starved, as $t \to \infty$, $\mathcal{S}$ will converge to level-optimality.*

Proof. Fix $\overline{s} \in \overline{S}$ and threshold $z \ll 1 \in R^+$. We need to show that in the limit, $\pi^*(\overline{s}) = (a, \chi_{\mathcal{S}}(\overline{s}, a))$. By Proposition 1, $\mathcal{S}$ will converge to λ-stationarity for $(\overline{s}, a)$ for all $a \in A$. Hence there is a finite point t at which the expected value of information on Σ falls below z for $(\overline{s}, a)$ for every $a \in A$ and $\mathcal{S}$ will *exploit under stationarity* for $\overline{s}$. That is, at such time, $\pi^*(\overline{s}) = (a, \text{argmin}_{l \in \kappa_t(\overline{s}, a)}(q^*(\overline{s}, (a, l))$. By Proposition 3, this value is exactly the definition of $\chi_{\mathcal{S}}(\overline{s}, a)$ provided that $\chi_{\mathcal{S}}(\overline{s}, a) \in \kappa_t(\overline{s}, a)$. By assumption, $\chi_{\mathcal{S}}(\overline{s}, a)$ is reachable from $\kappa_0(\overline{s}, a) \subseteq \kappa^{\mathcal{H}}(\overline{s}, a)$, so given that under the gated exploration strategy, there is a nonzero probability of reaching $\chi_{\mathcal{S}}(\overline{s}, a)$, and as $\overline{s}$ is arbitrary, we are done. □

5. Improving Competence Online

As discussed in Section 1, many problems in the open world are too complex to fully specify *a priori* all features that will be relevant over the course of the system's deployment, even with expert knowledge of the domain. This is particularly prevalent with features that may not directly impact the technical functionality of the autonomous agent (e.g. its domain model) but rather are factors that influence the human's feedback which may encompass additional features that affect other elements such as comfort or social behavior [8, 58]. Preliminary analysis of override data collected on a real autonomous vehicle prototype from two different safety drivers corroborates this claim. Here, the AV could either be in *supervised autonomy*, or could defer full control to the human; overrides corresponded to braking or accelerating registered by the human driver while the AV was operating in supervised autonomy.

The results of this analysis can be seen in Table 5 where we provide the correlation matrix for each type of override with every feature used by the CAS model implemented on the AV for each human safety driver. These results demonstrate two important facts. First, the difference in correlation matrices between Human 1 and Human 2 illustrate that feedback, and the features which determine that feedback, can vary significantly between humans, meaning there is no "one-size-fits-all" feedback model. Second, the lack of any feature having a correlation coefficient greater that ±0.4 indicates that it is challenging, even with expert input, to capture all of the causal features used by all humans *a priori*. If the CAS model does not represent certain features in its model that are used by the human in deciding their feedback signals (either explicitly or implicitly), the human's feedback may appear inconsistent or even random, leading to low competence and a potentially high degree of improper reliance on the human stemming from an underspecified model. Consequently, for these systems to be most effective in the real world it is important that they are equipped with a means of updating their model online to better align with the human's model so that they can better predict the correct feedback likelihoods.

To address this shortcoming, we propose a method for providing a CAS the ability to improve its competence over time by increasing the granularity of its state representation through online model updates. The approach works by identifying states that are deemed *indiscriminate* under the system's current feedback profile, i.e. unable to predict human feedback with high confidence, and attempts to find the feature, or set of features, that is available to the system but currently unused that best discriminates human feedback, leading to a more nuanced drawing of the boundaries between regions of the state space with different levels of competence. An example of this process can be viewed in Figure 6. By exploiting the existing information available in a standard CAS model (namely, the existing human feedback) to identify where features may be missing and should be added, our approach adds no additional work to the human at all. Additionally, when the missing features impact only the human's feedback profile (and not the system's technical capabilities), or when using a CAS with levels of autonomy that involve forms of human assistance that maintain safe operation (like that which is described in the running example) we only need to modify the state space directly, and not the transition or cost functions, enabling the entire process to be performed online and fully autonomously.

Example 5. *Recall the scenario in our running example, where the AV (blue) must overtake an obstacle blocking its lane (red) by driving into the oncoming traffic's lane (yellow). Now, consider the existence of a trailing vehicle (or vehicles) waiting behind the AV (green); the existence of trailing vehicles may not be included in the state representation of the domain model as they do not affect the decision making of the AV from a technical perspective (that is, they do not influence the success or failure probabilities of each action, do not influence the safety of the actions, and short of rear-ending the AV do not directly alter the AV's state), and serve only to increase the state space of the planner. However, it may be the case that the human in the AV is actually* more likely *to override safe behavior, such as waiting if there is an oncoming vehicle, and take manual control of the vehicle due to the social pressure exerted by the trailing vehicle's existence.*

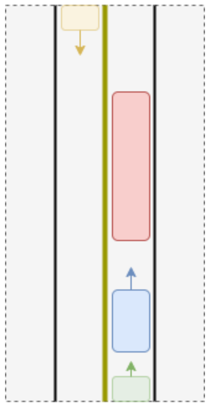


Figure 5: Illustration of Example 5

5.1. Indiscriminate States

Let $\mathcal{S}$ be a competence-aware system. In practice, when a robotic system is deployed into the open world, both the exact environment the system will operate in, and the human authority it will interact with, may not be known *a priori*. Naively including all possible features available to the system from perception or external sources in its planning model may make planning intractable without benefit in the case where many of the features do not add useful information for decision making and serve only to increase the number of states. Hence, we assume that $\mathcal{S}$ has available to it a *complete feature space* that can be partitioned into an *active feature space* that is used by $\mathcal{S}$ and an *inactive feature space* that is not yet used by $\mathcal{S}$ in its planning model. However, as $\mathcal{S}$ receives additional feedback over time, $\mathcal{S}$ will learn to exploit some of the inactive features, adding them to its state representation to more effectively align its features with those used by the human authority.

Definition 15. *Given the* ***complete feature space*** $F = \{F_1, F_2, ..., F_n\}$ *available to* $\mathcal{S}$*, the* ***active feature space*** *is denoted as* $\hat{F} \subseteq F$*, and the* ***inactive feature space*** *as* $\check{F} = F \setminus \hat{F}$.

We say that a state $\overline{s} \in \overline{S}$ is *indiscriminate* if, intuitively, the active feature space is missing features needed to properly discriminate the feedback received from the human for the state $\overline{s}$. The condition states more precisely that for at least one action there must be no feedback signal that, under the system's current feedback profile, can be predicted with high probability. The intuition is that, under the assumption of ϵ-consistency and a ground truth feedback, situations where the agent cannot predict feedback with high probability indicate that a feature may be missing from its state representation causing the probability mass to be normalized over the remaining features in its active feature space. We formalize this below.

Definition 16. *Let the human authority* $\mathcal{H}$ *be* ϵ*-consistent for* $\epsilon > \frac{1}{|\Sigma|}$*. A state* $\overline{s} \in \overline{S}$ *is* ***indiscriminate*** *if there exists at least one action,* $\overline{a} \in \overline{A}$*, where for every feedback signal* $\sigma \in \Sigma$*, we have the following:*

$$\lambda(\sigma \mid \overline{s}, \overline{a}) \leq 1 - \delta \qquad \delta \in (1 - \epsilon, 1 - \frac{1}{|\Sigma|}) \tag{7}$$

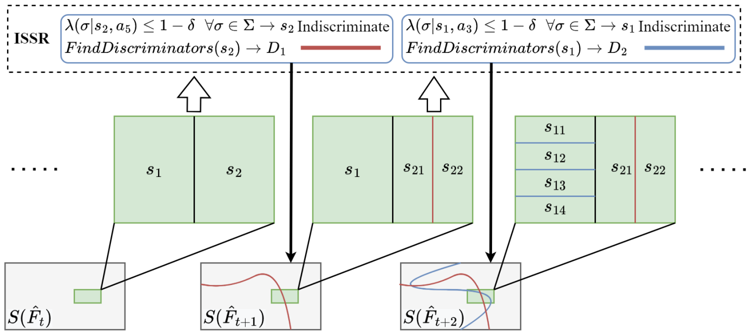


Figure 6: An illustration of *iterative state space refinement*. $S(\hat{F}_i)$ represents the state space given the active feature set $\hat{F}_i$. The middle row depicts a "zoomed in" view of a small part of the state space. We can see that originally, with active feature set $\hat{F}_1$, there are only two states in the subspace: s_1 and s_2. The top row depicts the key information found by our algorithm: first, it identifies that s_2 is an *indiscriminate state* given λ, and finds the discriminator D_1 (represented by the red line) which then partitions s_2 into two states: s_{21} and s_{22}. The process repeats once more, finding that s_1 is also an indiscriminate state, and finding discriminator D_2 which partitions s_1 into four states: s_{11}, s_{12}, s_{13}, and s_{14}.

Here, δ is referred to as the *discrimination slack*, and determines the predictive confidence needed for a state to be declared indiscriminate; the lower the slack is set, the higher the confidence needed. The discrimination slack serves to provide a formal trade-off mechanism between increasing the complexity of the underlying planning model, and the completeness of the competence-aware model. The determination of how to set δ may be done via expert knowledge, offline evaluations, or could even be tuned online in a dynamic fashion. To avoid considering states that have a very small amount of data (and hence may be deemed "indiscriminate" due to chance), we consider only states for which the system has collected a sufficient amount of data (which may be determined simply via a fixed threshold, or based on some statistical analysis).

Given the notion of an indiscriminate state, we can now define the central concept of this approach. A *discriminator* is, intuitively, any *subset* of the inactive feature space that could help the agent to better discriminate feedback from $\mathcal{H}$ for an indiscriminate state. For example, consider the autonomous vehicle agent in Running Example 5 that initially does not consider the existing of a trailing vehicle in its active feature set. Suppose that the human always overrides the vehicle and takes manual control when there is a trailing vehicle if the AV waits for too long before proceeding around the obstruction to maintain safe operation. Without this additional feature in its model, the agent may perceive having received "noisy", or even seemingly random, feedback from the human authority, leading to a feedback profile with low predictive capabilities and a poor competence model, resulting in the AV conservatively transferring control to the human when performing an overtake in situations where it was actually competent to act autonomously. By providing the agent with the ability to add these features to its active feature space, the agent's new feedback profile will be able to predict the correct feedback signal in more situations with higher probability.

5.2. Iterative State Space Refinement

Definition 17. *A **discriminator** is any subset of $\check{F}$ which, if added to $\hat{F}$, will improve the performance of λ by at least α, for some $\alpha \in (0, 1)$.*

The larger that α is set, the stricter the requirement is on including a new feature. Determining α can be as simple as setting it to be a fixed threshold, or can be via more sophisticated means such as based on the value of information or other information-theoretic metrics. The methodology for selecting discriminators is well explored in the feature selection literature and not the focus of this contribution; standard approaches include mRMR [65], JMI [17], and correlation-based methods [82]. We define a discriminator as a subset because there may be causal features which if added individually do not help to discriminate the human's feedback, but when added together do (i.e. they are only meaningful in the context of each other). The size of feature subsets to consider when selecting potential discriminators is therefore an important parameter of the approach, but we note that, if desired, one could also take an iterative approach, running the algorithm with increasing size until a discriminator is found.

Algorithm 1: Single–Step State Space Refinement

Input: A CAS $\mathcal{S}$, dataset $\mathcal{D}$, slack δ, and threshold M
Result: An updated CAS $\mathcal{S}$

1 $\overline{S}^* \leftarrow \{\}$
2 **for** $\overline{s} \in \mathcal{S}.GetStates()$ **do**
3   **for** $\overline{a} \in \mathcal{S}.GetActions()$ **do**
4   **if** $\max_{\sigma\in\Sigma} \lambda(\sigma|\overline{s}, \overline{a}) \leq 1 - \delta$ and
5         $\max_{\sigma\in\Sigma} \Pr[Obs(\mathcal{D}(\overline{s}, \overline{a}))|\sigma \text{ is ground truth}] < p_\epsilon$
6             $\overline{S}^* \leftarrow \overline{S}^* \cup \{\overline{s}\}$
7   **end**
8   **end**
9 **end**
10 **if** $\overline{S}^* = \emptyset$
11   **return** $\mathcal{S}$
12 **end**
13 $\overline{s}^* \sim \overline{S}^*$
14 $\mathcal{D}_{train}, \mathcal{D}_{val} \leftarrow Split(\mathcal{D})$
15 $D \leftarrow FindDiscriminators(\mathcal{D}_{train}, \breve{F}, \overline{s})$
16 **for** $d \in D$ **do**
17     $\lambda_d \leftarrow train(\hat{F}_1 \times \cdots \times \hat{F}_{|\hat{F}|} \times d, \mathcal{D}_{train})$
18 **end**
19 $d^* = \text{argmax}_{d\in D}\, Evaluate(\lambda_d, \mathcal{D}_{val})$
20 **if** $Validate(d^*, \mathcal{S})$ **is** $True$
21     $\hat{F} \leftarrow \hat{F} \cup d^*$
22     $\mathcal{S}' \leftarrow Update(\mathcal{S})$
23 **end**
24 **return** $\mathcal{S}'$

Algorithm 1 presents the pseudocode of our approach for improving the competence of a CAS via iterative partitioning of the state space by adding new features to the state representation over time. The algorithm first identifies the current set of indiscriminate states (Lines 1–9). To avoid labeling sparsely sampled state-action pairs as indiscriminate through chance, we limit the process to only consider certain state-action pairs. In particular, only those where the probability of having observed all labeled instances of that element in the existing dataset $\mathcal{D}$, referred to in Algorithm 1 as $Obs(\mathcal{D}(\overline{s}, \overline{a}))$, is at least some threshold p_ϵ conditioned on the assumption that there exists a true correct feedback signal returned with probability at least ϵ by the human for every state-action pair (Line 5). Next, the algorithm samples an indiscriminate state from the set (Line 13) and identifies the most likely discriminators for that state using any standard feature selection technique (in our case, we used mRMR [65] with the FCQ methodology [102]) (Line 15). For each potential discriminator, a new feedback profile is trained using a portion of the full dataset with the discriminator temporarily added to the active feature set (Lines 16–18). The discriminator that leads to the best performing feedback profile, in our case the highest Matthews correlation coefficient, is selected for validation (Line 19). If validation is successful, the discriminator is added to the active feature set and the system is updated (Lines 20–23).

In the design and usage of Algorithm 1, we make two key assumptions. First, we assume that the initial transition function provided in the domain model is *sufficiently correct* for any scenario where the agent is allowed, under $\kappa^{\mathcal{H}}$, to act autonomously. We aim to improve the robustness of deployed systems where accounting for every scenario *a priori* is infeasible, but where the scenarios that are considered *a priori* are well-designed.

Second, we assume that the human authority has a sufficient understanding of the agent's capabilities to both prevent the execution of an action that the agent cannot perform successfully and also provide consistent feedback. We make this assumption for two reasons. First, there are different ways to improve the human authority's understanding of the system's capabilities so that it has the appropriate trust [45], or reliance, on the system. These include pre-deployment training, standardized feedback criteria, and expert knowledge of the system. Second, recognizing potential failures and handling fault recovery are separate areas of active research [7, 27, 94] that are orthogonal to what we examine here.

Critically, under these assumptions, *we do not need to update the domain model's transition or reward functions directly at any point.* It suffices for the agent to be able to discriminate between actions that it has the competence to perform autonomously and actions that require human involvement because, under the first assumption, T is correct when the agent is allowed to execute an action autonomously. Consequently, the only elements of the CAS transition function, $\overline{T}$, that are marginally dependent on features added to the state representation are λ and $\tau_{\mathcal{H}}$. As λ and $\tau_{\mathcal{H}}$ are learned online from observed feedback, we can directly compute the respective new distributions over $\hat{F}'$ from the current dataset which in turn updates the transition function as λ and $\tau_{\mathcal{H}}$ are both parameters of $\overline{T}$. We suggest that when one or both of these assumptions do not hold it is possible to use our approach as a means of identifying the missing features and subsequently improving the system's competence by directly updating the transition and cost functions (e.g. via software updates).

A natural question is whether in the process of adding a discriminator to make some indiscriminate states discriminate, we will, as an unintended by-product, make some discriminate state indiscriminate.

Remark 1. *Adding a discriminator will never cause a discriminate state to become indiscriminate.*

While possibly not obvious a priori, this remark is trivially true. Observe that any given discriminate state will either be affected by the discriminator or it will not. If it is not affected, the feedback profile for the state will not change. If the state is affected, then the initial state in question by definition no longer exists. More importantly, we want to ensure that every state is eventually properly discriminated given a sufficient set of features.

The following proposition states that if every feature that the human uses to determine their feedback is available to the robot, then there must be a point in time at which the robot has fully discriminated all states, and no state will become indiscriminate past that point.

Proposition 4. *Let I_t be the number of indiscriminate states at time t, and let $\lambda_t^{\overline{s},a}$ be the random variable representing $\lambda(\overline{s}, a)$ after having received t feedback signals for $(\overline{s}, a)$ where each signal is sampled from the true distribution $\lambda^{\mathcal{H}}(\overline{s}, a)$. If $F^{\mathcal{H}} \subseteq F$, $\mathcal{H}$ is ϵ-consistent, $\delta > 0$ and no $(\overline{s}, \overline{a}) \in \overline{S} \times \overline{A}$ is starved, then there exists some $t^* > 0$ for which $I_{t'} = 0$ for all $t' > t^*$.*

Proof. First, observe that as $F^{\mathcal{H}} \subseteq F$, if there is a point at which $F^{\mathcal{H}} \subseteq \hat{F}$, then because the sequence $\{\lambda_t^{\overline{s},a}\}$ converges in distribution by Proposition 3, $\lim_{t\to\infty} \Pr(|\lambda_t^{\overline{s},a} - \lambda_{\mathcal{H}}^{\overline{s},a}| > \gamma) = 0\ \forall\gamma > 0, (\overline{s}, a) \in \overline{A} \times A$. Hence, there exists some $t^* > 0$ for which $\Pr(|\lambda_t^{\overline{s},a} - \lambda_{\mathcal{H}}^{\overline{s},a}| > \delta) = 0$ at which point it is clear that no state will be indiscriminate under δ. Consequently, for the claim to not hold, it must be the case that for every $t > 0$, $F^{\mathcal{H}} \setminus (F^{\mathcal{H}} \cap \hat{F}) \neq \emptyset$. Pick such a t, sufficiently large, for which there is an indiscriminate state $\overline{s} \in \overline{S}$. There is some subset, $G \subseteq F^{\mathcal{H}} \setminus (F^{\mathcal{H}} \cap \hat{F})$, which is a discriminator of $\overline{s}$. As this holds for all $t > 0$ and $\overline{s} \in \overline{S}$, we either reach a satisficing t^* where $F^{\mathcal{H}} \setminus (F^{\mathcal{H}} \cap \hat{F}) \neq \emptyset$, and hence are done, or where $F^{\mathcal{H}} \subseteq \hat{F}$ which contradicts our assumption. □

6. Empirical Evaluations

To test the competence-aware system, we implemented the CAS model in two simulated autonomous vehicle domains at different levels of decision-making abstraction. The first domain is a high-level navigation problem in which an autonomous vehicle must plan (and execute) the optimal route to take between two locations conditioned on its knowledge about different intersections and streets and its own competence in performing different maneuvers at the various locations. The second takes a more fine-grained look at one of the maneuvers that can be performed in the first domain, namely passing an obstacle that is blocking its lane, and is modeled after the domain depicted in Example 1.

We evaluated our iterative state space refinement approach (Algorithm 1) on both of these domains as well, where the key difference is that the CAS model is missing features in its initial active feature space that do not impact its transition model (that is, what it is technically capable of doing), but impact the human's feedback signal likelihoods regardless. We test our approach for multiple different simulated humans, each of whom uses different auxiliary features in determining their feedback. We describe an overview of the domains below, and include additional experimental details in Appendix A.

6.1. Autonomous Vehicle Navigation

6.1.1. Domain Description

In this domain, an autonomous vehicle operates in a known map represented by a directed graph $G = (V, E)$ where each vertex $v \in V$ represents an intersection and each edge $e \in E$ represents a road; the graph used can be seen in Figure 7 and is modeled after locations in the area of Amherst, Massachusetts. The autonomous vehicle is tasked with navigating the map safely from a start vertex to a goal vertex.

Each vertex (intersection) state is represented by an ID for the vertex, a boolean indicator of the presence of pedestrians, a boolean indicator of the presence of an occlusion limiting or blocking visibility, the number of other

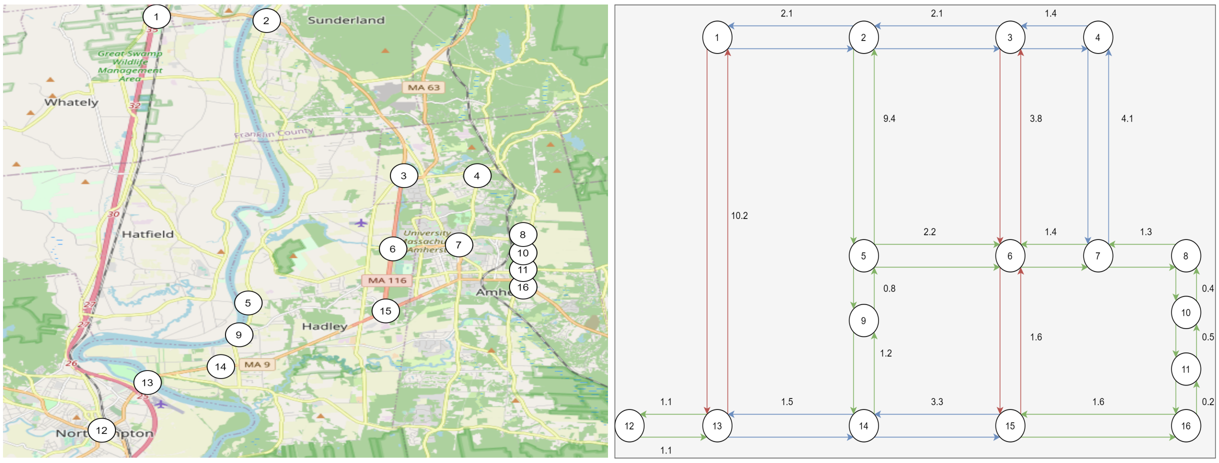


Figure 7: A depiction of the map used for our simulated navigation domain with actual locations from OpenStreetMap (left) and the abstracted representation of the navigation graph (right).

vehicles at the intersection (0-4), and the vehicle's heading. Each edge (road) state is represented by a start vertex ID, a destination vertex ID, the number of drivable lanes on the current road segment, the direction of travel, and a boolean indicator of the presence of an obstruction blocking the agent's lane. Additionally, each edge is associated with a known length and speed of travel. Model parameters dictating the probabilities of each state variable (e.g. the probability of a pedestrian being at a given intersection upon reaching it) are assumed to be known offline and given as part of the model input.

In vertex states, the agent can either `Go Straight`, `Turn Right`, `Turn Left`, `U-Turn`, each of which has a cost of 10.0, or `Wait`, which has a cost of 1.0. All maneuvers succeed deterministically. In edge states, the agent can either `Continue` or `Overtake` an obstruction, each with unit cost. `Overtake` is assumed to succeed with probabilities $[0.2, 0.5, 0.8]$ depending on the number of lanes. `Continue` fails deterministically in the presence of an obstruction, and if there is no obstruction transitions the agent to the end-vertex of the edge with probability $p \propto speed(e)\ /\ length(e)$ or otherwise to the same edge with some probability of an obstruction occurring. We model the expected duration as part of the transition function, rather than the cost function, to allow for the development of an obstruction in the AV's lane while traversing an edge segment which may be very long in real life.

We consider the following levels of autonomy, $\mathcal{L} = \{l_0, l_1, l_2, l_3\}$ where l_3 does not require any involvement from the human at all (i.e. we assume the probability of an override is 0), l_2 allows the agent to execute an action under supervision, during which the human may override the action if they deem it unsafe, l_1 which requires explicit approval from the human for an action prior to its execution during which, if approval is received, the agent may attempt to execute the action under supervision, and if the action is disapproved by the human the agent must select a different action to perform, and l_0 which requires full transfer of control to the human to complete the action.

The autonomy profile, κ, is initialized to $\mathcal{L}$ in edge states without an obstruction and otherwise to $\{l_0, l_1, l_2\}$. The feedback profile, λ, is initialized to be uniformly random over the possible feedback signals. There is an associated cost of 10.0 to the human for operating in l_0, as the human is required to manually control the vehicle, a cost of 2.0 for operating in l_1, a cost of 1.0 in level l_2, and no additional cost to the human when operating in l_3. The system incurs a cost of 3.0 when receiving a negative response in l_1 and a cost of 10.0 when receiving an override in l_2 as we assume that the human completes the intended action.

6.1.2. Results

To validate the CAS model in the AV navigation domain, we randomly selected a start node and goal node each episode to ensure that the system had the ability to visit the entirety of the graph. We repeated this for four different human authorities where we varied their consistency: 0.8, 0.9, 1.0 (i.e. perfectly consistent), and, in the final case, a human who starts with a very low consistency (0.6) to reflect their poor understanding of the capabilities of the system, but increases their consistency by a small amount (0.1) each episode to reflect their improved understanding of the capabilities of the system over time as they interact with it. Figures 8, 9 and 10 report the results from the experiment conducted in the autonomous vehicle navigation domain.

Figure 8 depicts the results on a fixed route (node 12 to node 7 in Figure 7). The top graph shows the expected cost of the route and the bottom graph shows the actual mean cost (averaged over 100 simulations) of the CAS (blue) compared against an agent just using the domain model agnostic to its competence, with a human overriding as necessary (i.e. effectively always operating in level l_2) (red). These results demonstrate that by learning an accurate competence model and incorporating that into the planning model, a CAS can efficiently (< 40 feedback signals) improve both its average performance and expected performance, significantly outperforming a system that is agnostic to its competence and the dynamics of human interaction. These experiments were taken from the human with consistency $\epsilon = 0.9$ but we note that very similar results were obtained in all cases.

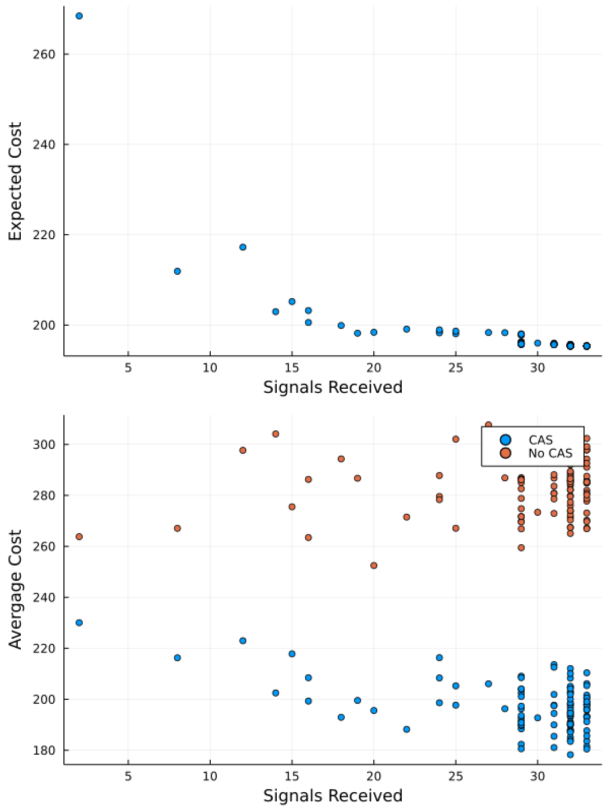
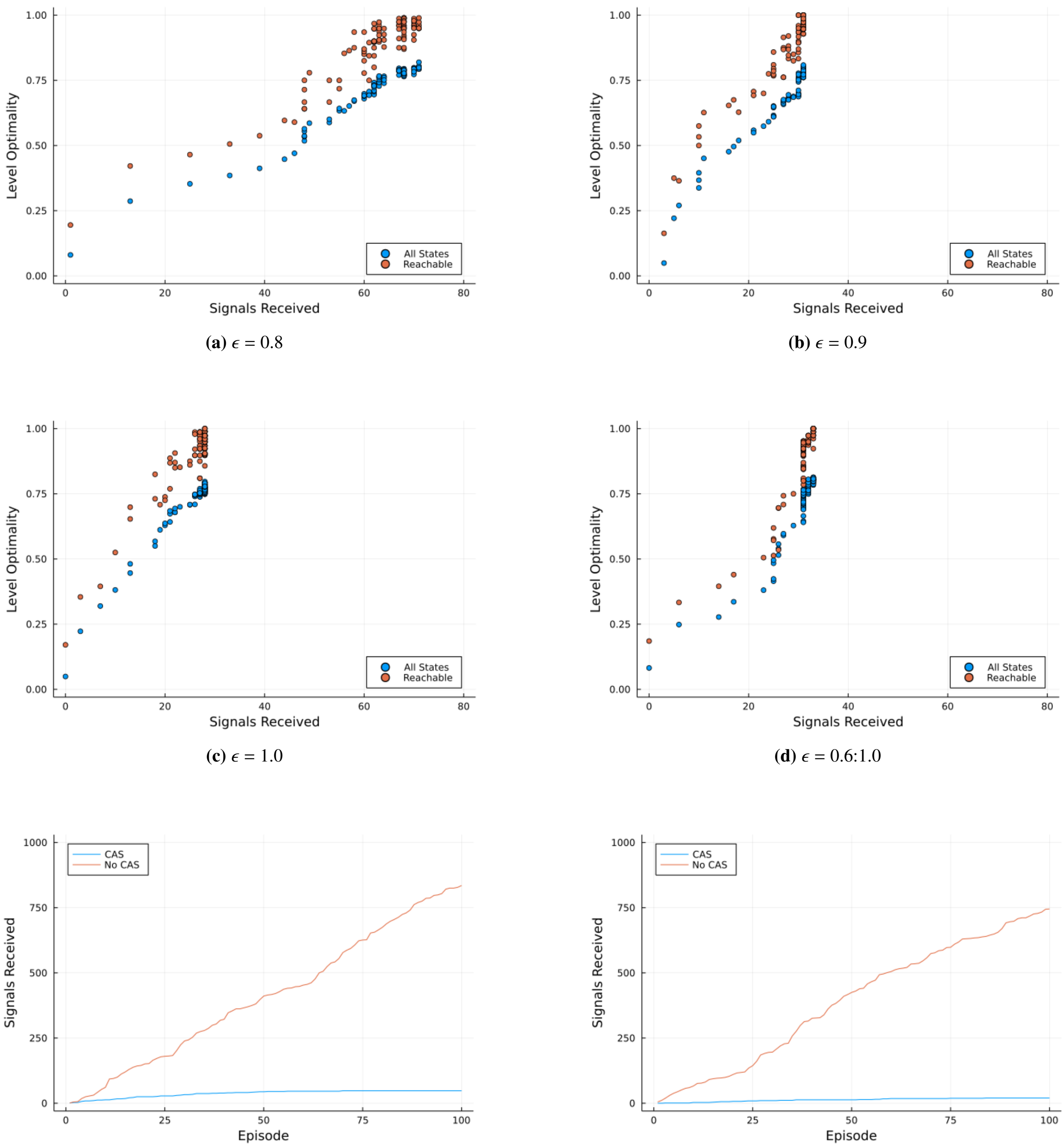


Figure 8: Empirical results from simulations of a fixed route ($12 \rightarrow 7$) showing the expected cost (top) to goal of a CAS and the average cost (bottom) over 100 trials with a CAS (blue) and without a CAS (red) as a function of the number of signals received.

Figure 9 depicts in the top two rows the convergence of the level-optimality of the competence-aware system as a function of the number of feedback signals received, and in the bottom row the number of signals received over the course of 100 episodes (where each episode is a random route) for a system with a CAS (blue) and a system without a CAS (red). Each graph corresponds to a human authority with a different consistency, ϵ, as detailed above. In all cases, the level optimality reaches 100% over all reachable states in the domain. Interestingly, in Figure 9d, the results are more comparable to a human with a fixed consistency of 0.9 or 1.0 in the level-optimality convergence rate than they are to a human with a fixed consistency of 0.8 which requires roughly twice as many feedback signals to converge to level-optimality. This demonstrates that even a CAS with a human who starts with an initially poor understanding of the system's capabilities, and consequently low consistency, can efficiently reach level-optimality if the human's understanding and consistency improves at a consistent rate. The figures in the bottom row illustrate that without a CAS the number of feedback signals provided by the human grows linearly, demonstrating the significant disparity in burden placed upon the human in a system without a CAS model compared to a system with a CAS model. We only depict the results for 0.8 and 1.0 for the sake of space, but the results look very similar for all ϵ-consistencies considered. Overall these results demonstrate the primary goal of the CAS model which is that it enables a system to efficiently reach level-optimality, optimizing the trade-off between autonomous performance and human assistance, thereby reducing the net burden placed on the human over the course of the system's operation.

Figure 10 depicts the change in routes taken between the first episode and the 100th episode for the CAS model for four fixed routes. Here, purple denotes parts of the route taken that are the same, red denotes parts of the route that are taken in the first episode but not the 100th, and blue denotes parts of the route that are taken in the 100th episode but not the first. This figure illustrates the *macro* policy changes made as the CAS learns its competence—namely altering its route to avoid states or trajectories of low competence which would require excessive human assistance—in addition to the *micro* changes of selecting which level of autonomy to use in any given situation. In general, we find that the AV's behavior changes to avoid areas densely populated with pedestrians, occlusions, and single lane roads, such as downtown Amherst (nodes 8-11) and University of Massachusetts Amherst (nodes 6-8).



(a) $\epsilon = 0.8$

(b) $\epsilon = 0.9$

(c) $\epsilon = 1.0$

(d) $\epsilon = 0.6{:}1.0$

(e) $\epsilon = 0.8$

(f) $\epsilon = 1.0$

Figure 9: Empirical results from the autonomous vehicle navigation domain with varying levels of human consistency showing the level-optimality as a function of the number of feedback signals received (9a – 9d) and the number of feedback signals received over the first 100 routes executed (9e - 9f). In Figure 9d, the human consistency increases after each route is executed, mimicking a human whose consistency improves the more it interacts with the system.

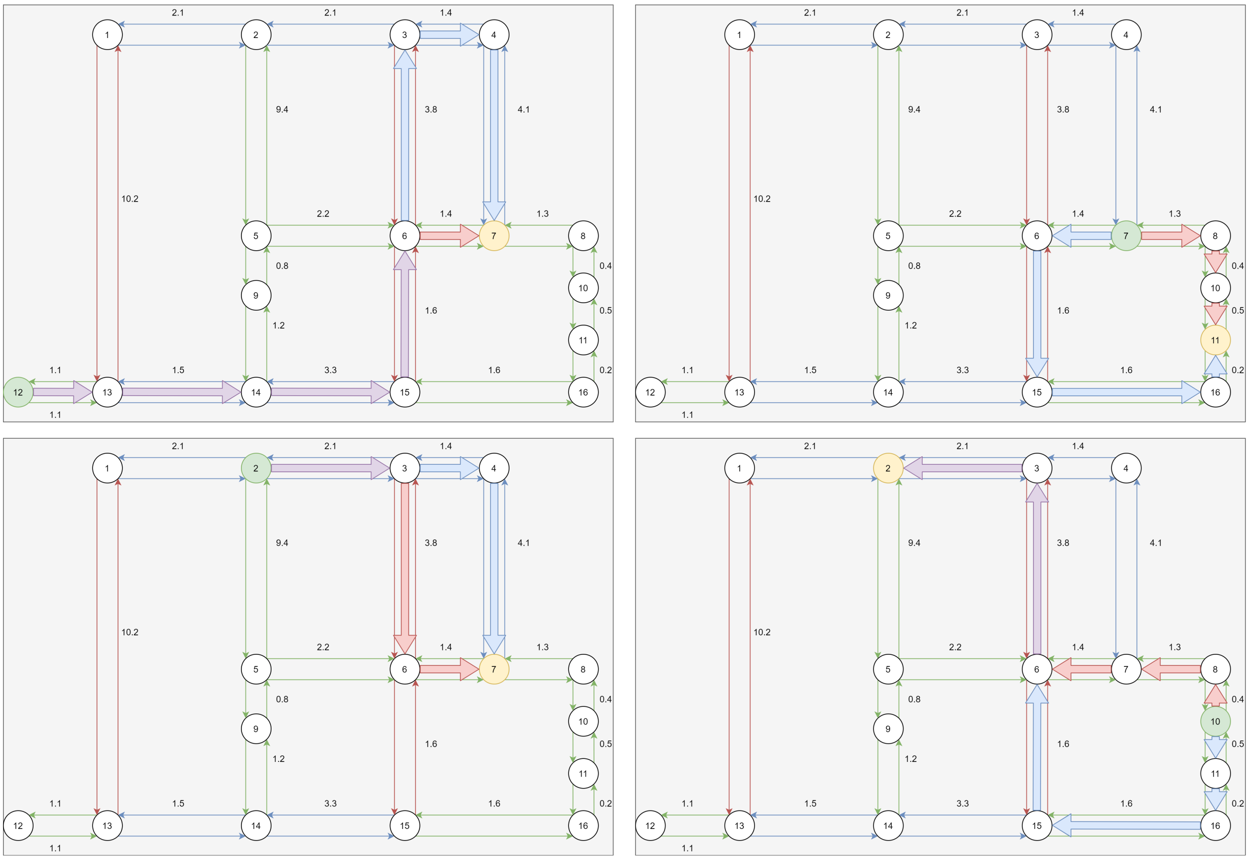


Figure 10: Comparison of routes taken before and after the CAS learns its competence. Purple indicates shared route, red indicates route taken by starting model alone, blue indicates route taken by ending model alone. Green and yellow circles denote start and end nodes respectively.

6.2. Autonomous Vehicle Obstacle Passing

6.2.1. Domain Description

In this domain, modeled after the problem depicted in Example 1, an autonomous vehicle must overtake an obstacle that is blocking its lane on a one-lane road. Importantly, this maneuver required that the AV drive into the oncoming traffic's lane in order to overtake the obstacle, a potentially dangerous maneuver.

Each state is represented by the vehicle's position (0-4), the position of an oncoming vehicle (0-3, or unknown), and whether the oncoming vehicle has given priority to the AV to attempt its overtake. Model parameters dictating the behavior of oncoming vehicles is assumed to be known offline and given as part of the model input.

The autonomous vehicle can perform the following actions: `Wait`, `Edge`, and `Go`. `Edge` provides visibility of oncoming traffic to the AV if unknown and otherwise advances the AV's position with probability 0.5. `Go` deterministically advances the AV's position, which results in a crash if the AV and an oncoming vehicle share the same position. `Stop` holds the AV's position, during which time the oncoming vehicles position may change (or become empty), or the oncoming vehicle may give priority to the AV. If the AV has priority it is assumed that the oncoming traffic will stay stopped until the AV has finished its overtake. All actions have unit cost, and crashing incurs a very high cost.

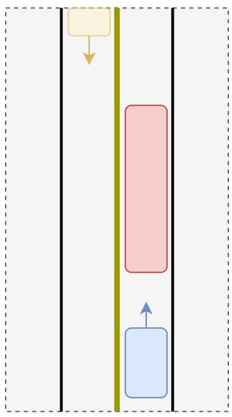
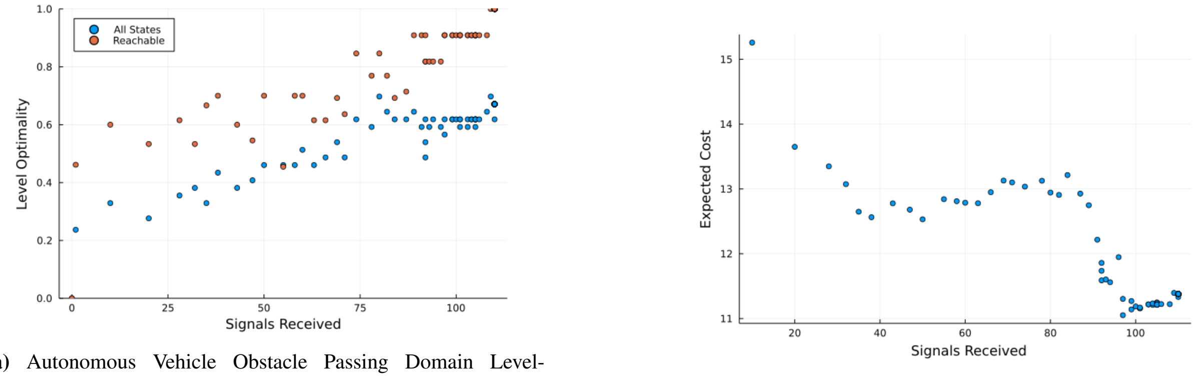


Figure 11: Illustration of the AV obstacle passing domain.



(a) Autonomous Vehicle Obstacle Passing Domain Level-Optimality

(b) Autonomous Vehicle Obstacle Passing Average Cost

Figure 12: Empirical results from the autonomous vehicle obstacle passing domain depicting the level-optimality (left) over all reachable states (red) and the full state space (blue), and the average cost (right) over 1000 simulations, as a function of the number of feedback signals received.

We consider the following levels of autonomy, $\mathcal{L} = \{l_0, l_1, l_2\}$ where l_2 does not involve the human at all, l_1 allows the agent to execute an action under supervision, during which the human may override the action if they deem it unsafe, and l_0 which requires full transfer of control to the human to complete the action. Note that we do not include the level l_1 from the prior domain (referred to earlier as "verified autonomy" in Table 1) due to the second-to-second nature of decision making in this safety-critical domain, where prompting the human for explicit approval before every action may be impractical or even dangerous.

The autonomy profile, κ, is initialized to $\{l_0, l_1\}$ in all cases; i.e., in such a safety critical domain it is expected that, initially, the human is always aware and ready to override the system. As above, the feedback profile λ is initialized to be uniformly random. The human incurs a cost of 10.0 when the CAS operates in l_0 but is assumed to complete the maneuver successfully (i.e., the human does not give back control part way through passing the obstacle), a cost of 1.0 when supervising in l_2, and no cost in l_3. The system receives a penalty of 10.0 when being overridden by the human.

6.2.2. Results

In the AV obstacle passing domain, the problem—i.e., the initial state and goal state—stayed fixed each episode. Figures 12a and 12b report the results from the experiment conducted in the autonomous vehicle obstacle passing domain. Figure 12a shows the level-optimality of the CAS over all states in the domain and all reachable states (each episode) plotted against the number of feedback signals received from the human, in this case consisting only of overrides. The figure illustrates that the CAS is able to converge to level-optimality on all reachable states in the domain with slightly more than 100 feedback signals. The slower convergence rate is due to a stricter requirement on gated exploration due to the more safety-critical nature of the domain (see Appendix A for details). 100% Level-optimality is not reached on the whole state space due to the absence of a portion of the state space ever being visited (or even reachable), preventing the human authority from providing any feedback for actions taken in those states. Figure 12b reports the expected cost of overtaking the obstacle and illustrates that the expected cost decreases as the level-optimality increases, corroborating the results from the previous domain. This also demonstrates that, in certain domains, performance may be improved to near optimal performance without even needing to converge to full level-optimality across the entire state space due to variations in state reachability trends.

6.3. Iterative State Space Refinement

To validate the *iterative state space refinement* method, we implemented Algorithm 1 and compared the performance of a CAS with Algorithm 1 and a CAS without it on both of the domains defined above (autonomous vehicle navigation and autonomous vehicle obstacle passing). In both experiments we considered different human users of the autonomous vehicle system, each of whose feedback was conditioned not just on the features already used by the CAS model that directly impacted the CAS's technical performance (i.e., the existence of a pedestrian, an occlusion, etc.) but additionally on auxiliary features which are tracked by the autonomous vehicle but not included in its *a priori* planning model, as the features in question are different for each person, and do not (directly) impact the transition and cost dynamics of the system.

In the AV navigation domain, the inactive feature set included the following features: whether the AV has a trailing vehicle, a vehicle to its left, or a vehicle to its right, whether the AV has been "waiting" to move, whether it is daytime or nighttime, and whether it is sunny, rainy, or snowy. In the AV obstacle passing domain, we consider the same inactive features except whether there is a vehicle to the AV's left or right, as the problem is for single lane roads.

In the AV navigation domain, we consider two "people" implemented as software agents: the first person is cautious with low trust in letting the AV operate in challenging environmental conditions (even though they do not impact the AV in simulation), for instance taking over control when the system attempts an overtake on a road segment when it is either snowing or rainy and night time. The intuition here is that the weather conditions impacts the human's ability to fully assess the situation and hence the veracity of the AV's actions, prompting them to take control of the vehicle themselves. We refer to them as "Cautious". The second person is motivated by more social factors, and is more likely to take control of the vehicle when there is a trailing vehicle the AV is blocking, and or when the AV has been stopped for too long (either on a road segment behind an obstruction, or at an intersection). We refer to them as "Conscientious".

In the AV obstacle passing domain, we consider three "people" implemented as software agents (see Appendix A for more details): the first is motivated by the same features as the first person above; we again refer to them as "Cautious". The second person is motivated by whether there is a trailing vehicle that they are blocking, prompting them to take control if the AV waits to long to attempt its overtake; we also refer to them as "Conscientious". The third person is in a rush and takes over control if the AV is waiting too long or doesn't go when it has priority; we refer to them as "Rushed". Each simulated person is perfectly consistent up to some fixed noise ϵ, within which they return uniformly random feedback.

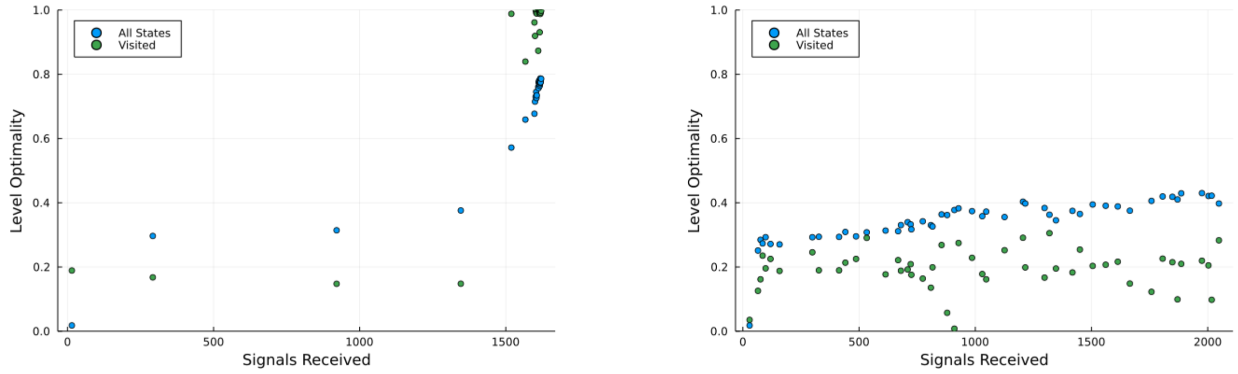
We note that in both domains, some inactive features are never used by any of the humans simulated, and hence we aim to show that our approach does not simply "pick all features" in the inactive feature space. Additionally, one important distinction between the two domains is that the additional inactive features may change at each new state in the AV navigation domain, but are fixed in the AV obstacle passing domain at the beginning of each episode due to the short time horizon of the problem. Details of the simulated humans can be found in Appendix A.

6.3.1. Results

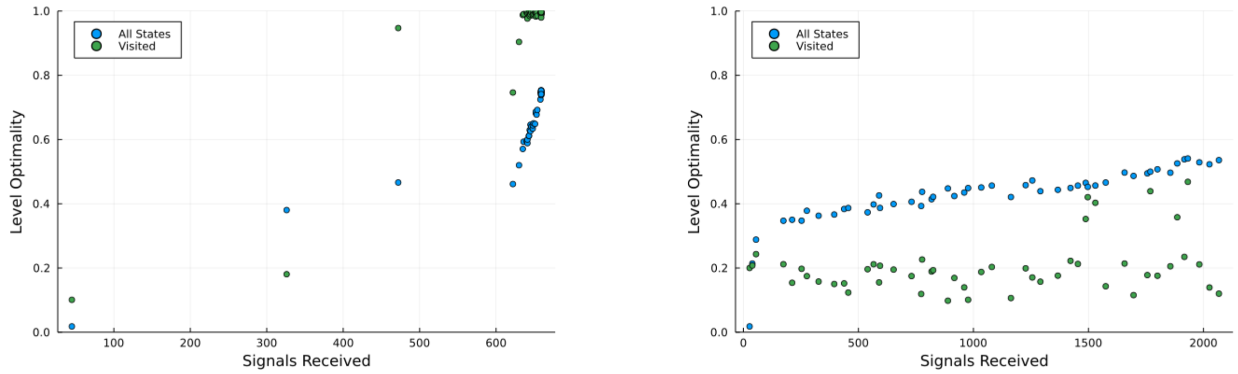
Figure 13 shows the results of our experiment, comparing the performance of a CAS with and without the iterative state space refinement (ISSR) approach (Algorithm 1) implemented, on the AV navigation domain with random routes each episode. Figure 14 shows the results for the AV obstacle passing domain. In Figure 13, we can see that the CAS with the ISSR implemented converges to higher level-optimality on all state in the domain, and 100% level-optimality on all states visited each episode, leading to far fewer feedback signals from the human, for both human authorities. Additionally, in both cases, the only features added to the active feature space where the features in the inactive feature space that were actually used by the humans in determining their feedback.

Figure 14 shows the results for the AV obstacle passing domain. Note that we include results on all reachable states here because the additional features stay fixed through each episode, whereas in the AV Navigation domain, they can change throughout an episode and the transition dynamics are (by design) not modeled by the agent.

There are several key takeaways from these graphs. First, if we consider the level-optimality over all states in the domain, it is higher for the ISSR-CAS in the cases of all three human authorities, than for the CAS without ISSR active, indicating that our approach is enabling the CAS to generalize its competence model to a larger portion of the (unvisited) state space. We remark that by adding features in order to refine the state space, the number of states increases multiplicatively with each feature added, meaning that not only is the ISSR-CAS level-optimal in a larger portion of the state space, that directly translates to being level-optimal in a larger number of unique situations. More important are the results depicting the level-optimality over all visited states each episode; here, we see that this reaches 100%, or near 100%, for all 3 human authorities with fewer than 50 feedback signals. However, we observe an interesting phenomenon for the CAS without ISSR active; namely, we see *several* clusters of green at the far right (at which point no additional feedback signals were received). This phenomenon is due to the fact that the CAS learns to operate in l_0, that is, full human control, in a large portion of the statespace because it cannot properly discriminate the feedback received from the human conditioned on features in the inactive feature space, which is



(a) Person 1 (Cautious)



(b) Person 2 (Conscientious)

Figure 13: Iterative state space refinement results for two human authorities in the autonomous vehicle navigation domain, showing the level optimality after each episode as a function of the number of feedback signals with (left) and without (right) Algorithm 1 implemented. Colors indicate the level-optimality over states visited during each episode (green) and the full state space (blue).

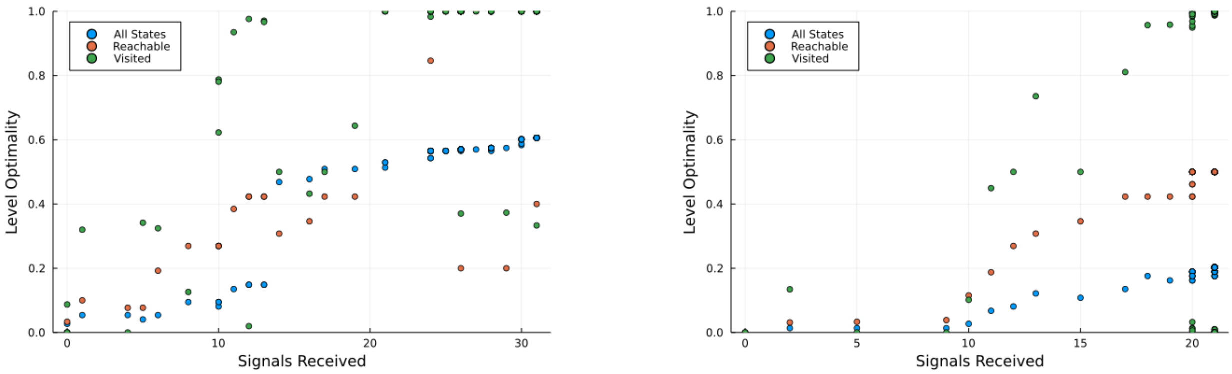
correct for certain settings of these features (which, to reiterate, are set and fixed at the start of each episode), but not for others. However, because the state space is not refined enough to consider these decision boundaries, the CAS learns to operate at the incorrect level of autonomy (relative to the full feature space) in certain conditions.

These results demonstrate that the ISSR method is effective at enabling a competence-aware system to improve its competence online when missing from its active feature space features used by its human authority.

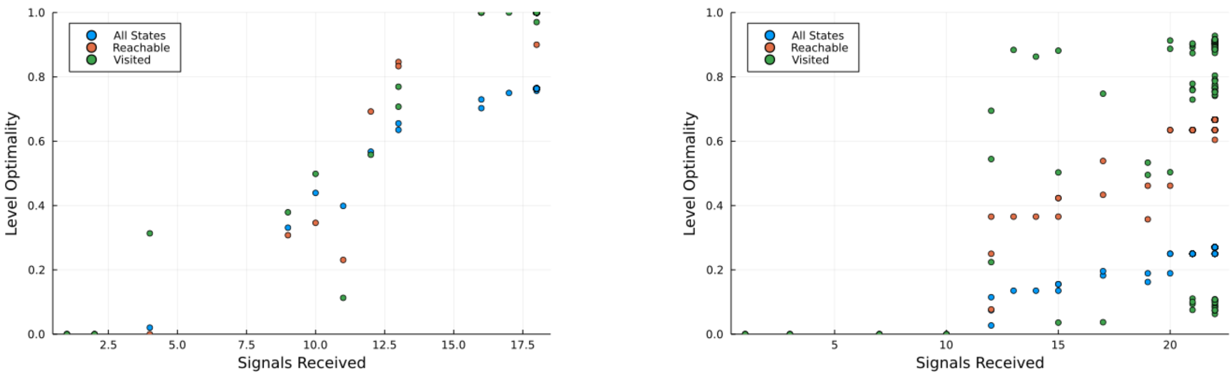
7. Discussion and Future Work

7.1. Autonomy Profile Initialization

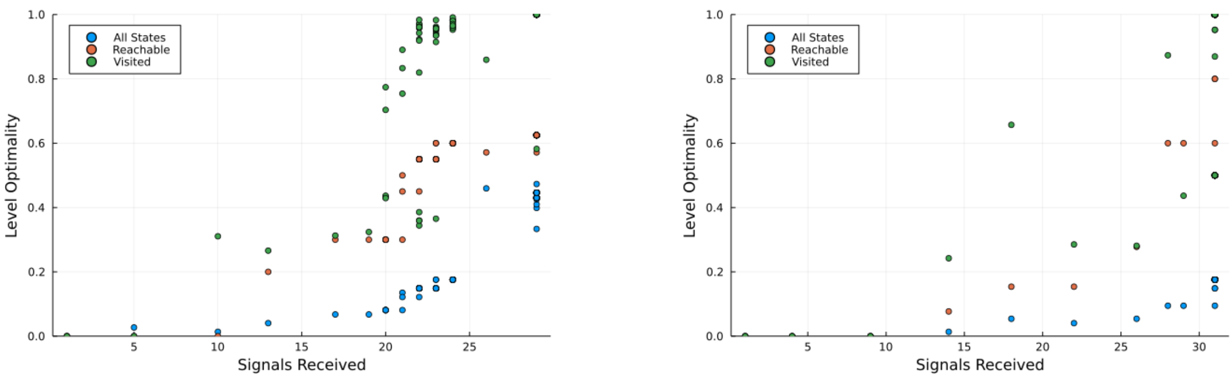
Because we restrict the system to choose policies from Π_κ, if the autonomy profile κ is altered, so too is the space of allowed policies. Hence, there is a trade-off when setting the initial constraints on the allowed autonomy of the system, i.e., κ. One can take a conservative approach and constrain the system significantly, for instance setting $|\kappa(s, a)| = 1$ so that a single level is deterministically selected for every $(s, a) \in S \times A$, reducing the problem complexity to solving the underlying domain model. However, doing so risks a globally sub-optimal policy with respect to $\mathcal{L}$ and may, depending on the initial κ, make reaching the globally optimal policy impossible. On the other extreme, one can take a risky approach and not constrain the system at all a priori, leaving the decision of choosing the level of autonomy completely up to the system when solving its model. This approach, while necessarily containing the optimal policy (subject to the agent's model) is naturally slower due to the larger policy space and inherently less safe as the agent



(a) Person 1 (Cautious)



(b) Person 2 (Conscientious)



(c) Person 3 (Rushed)

Figure 14: Iterative state space refinement results for three human authorities in the autonomous vehicle obstacle passing domain, showing the level optimality after each episode as a function of the number of feedback signals with (left) and without (right) Algorithm 1 implemented. Colors indicate the level-optimality over states visited during each episode (green), all reachable states each episode (red), and the full state space (blue).

can take actions in undesirable levels. Figure 4 illustrates different partitionings of the policy space under different autonomy profiles.

We propose that in practice, the desired initialization is somewhere in the middle; κ should be less constraining in situations where the expected cost of failure is relatively low, and more constraining in situations where it is high. While the model makes no such requirements, in many practical settings such information may be at least partially known *a priori* for a specific domain. For instance, in an autonomous vehicle, κ should be more constraining initially in situations involving pedestrians, poor visibility, or chaotic environments such as large intersections with multiple vehicles; however, initial testing may indicate that driving along a highway is low-risk and may not require a highly constraining κ.

7.2. Model Assumptions

We now discuss the practical considerations of the two main assumptions made in Section 3.4: (1) the human authority, $\mathcal{H}$, provides consistent feedback and (2) the human authority's feedback comes from a stationary, Markovian distribution.

Implicit in Assumption (1) is that humans respond appropriately to each situation, possibly with some noise representing the likelihood of human error. However, because of the limited scope of the system's domain model, it could be that perfectly consistent feedback from $\mathcal{H}$'s perspective is *perceived* to be random by the system, particularly when it is not aware of the domain features that explain the human feedback. As an example, consider a robot that can open 'push' doors and cannot open 'pull' doors, but does not model this discriminating feature. If the robot cannot discriminate between these types of doors, consistent and correct human feedback (approving autonomously opening 'push' doors only) may be perceived by the robot to be arbitrary or random. Although in practice one may wish to avoid such situations, we emphasize that the system *will still converge to its competence for the state features it uses*—possibly a low competence—when the feedback distribution appears to be random.

Assumption (2)—the human feedback distribution $\lambda^{\mathcal{H}}$ is stationary and Markovian from the start—implies that the human has good knowledge of the system from the start. That may not be realistic in certain domains. It is more likely that the feedback signals may vary based upon the observed performance of the system over time. However, as the human authority observes the system's performance, it is reasonable to assume that their feedback distribution will eventually reach a stationary point as long as the system's underlying capabilities stay fixed. Therefore, even if there are erroneous feedback signals provided early in this process, in the limit the system will still converge to its competence. Two possible means of expediting this is to introduce a training phase at the beginning of the system's deployment to allow the human to observe the system's performance and develop accurate expectations regarding the system's capabilities, and to introduce standardized feedback criteria that is made known to the human *a priori*.

7.3. Partially Observable Models

As stated in Section 3, the CAS is designed to handle fully-observable sequential decision-making models like SSPs and, more generally, MDPs, but is not immediately compatible with partially observable models (or mixed-observability models) despite partial observability and other limitations on state observability being a natural contributor to limitations on system competence. The two main barriers in directly applying the CAS to models like a POMDP are (1) the challenge of appropriately associating feedback signals with domain states for learning purposes when the system only has access to a belief state at any given time, and (2) the challenge in defining the competence of a belief-state, where the system implicitly does not know its true state. Future work will consider ways in which we can extend both the representation of feedback signals and the definition of competence, and consequently the CAS model, to such domains in a well-defined manner, for instance by changing the definition of competence from a function on states to a function on observations.

8. Conclusion

We introduce a new framework for representing, learning, and reasoning with self-competence models in semi-autonomous systems. Competence in our approach represents the level of autonomy that the system can handle reliably based on human feedback. More precisely, we define competence as the *optimal level of autonomy* in any given situation, consistent with perfect human feedback. We present a novel decision-making framework, *competence-aware systems*, that enables a semi-autonomous system to learn its own competence over time through interactions with a human authority. The result is a system that can handle risky scenarios by relying on the human authority to compensate for limitations or constraints on its autonomous abilities, while simultaneously optimizing its autonomous operation to reduce *unnecessary* reliance on humans.

We illustrate the operation of a competence-aware system with a running example and prove several theoretical properties of the CAS model. In particular, we prove that under standard convergence assumptions the model will converge to *level-optimality*, guaranteeing that the system consistently operates at its competence. We test the efficacy of our model empirically on two simulated autonomous vehicle domains, at different levels of reasoning abstraction, and demonstrate that the competence-aware system can efficiently reach high level-optimality, optimizing the trade-off between its own autonomous operation and human assistance, and leading to less burden on the human and a more cost-effective overall plan.

Preliminary internal testing on an autonomous vehicle prototype suggests that designing a perfectly specified CAS model for real-world, highly-unstructured domains is a non-trivial task. Even with expert domain knowledge, an initial model may be missing features used by the human in determining their feedback for the CAS. To avoid solving this naively with the inclusion of all possible system features in the CAS's domain model (many of which would serve no functional purpose but would cause the state space to explode and render planning intractable), we devise the *iterative state space refinement* approach. Described in Algorithm 1, the approach provides a competence-aware system the means to gradually refine its state representation online, enabling it to better identify the boundaries between state-action pairs with difference competences. This ability is particularly relevant in the context of systems deployed in the real world where human feedback may be conditioned on features that are unspecified or unknown *a priori*. Such features may not impact the original stated objectives of the system, but could influence unstated human preferences, trust, safety, and social conscientiousness. We prove that, when possible, this approach is guaranteed to reach a point where all states are discriminated, and demonstrate empirically that a CAS with this approach implemented far outperforms a CAS without it when the CAS cannot properly learn from human feedback due to missing state features. In particular, the modified CAS requires both fewer total feedback signals from the human, placing less burden on the human, and is more sample efficient with the feedback it receives in learning its competence, leading to a higher level-optimality for the CAS.

The primary direction of future work lies in extending competence-aware systems to models with limited state observability, such as MOMDPs and POMDPs. This includes devising a method of associating human feedback acquired in belief-states with underlying states in the domain, when the system does not know which state is responsible for the feedback, and generalizing competence to belief-states in a well-defined way that still captures the risk-sensitive semantics of the current approach. We are also interested in extending our model of human feedback to account for temporal uncertainty about the feedback signals, and to handle both proactive and retroactive feedback that is not necessarily associated with the action being currently executed.

Acknowledgements

This work was supported in part by the National Science Foundation grants IIS-1724101 and IIS-1954782 and in part by the Alliance Innovation Lab Silicon Valley.

References

[1] Allen, J.E., Guinn, C.I., Horvtz, E., 1999. Mixed-initiative interaction. IEEE Intelligent Systems and their Applications 14, 14–23. doi:10.1109/5254.796083.

[2] Altman, E., 1999. Constrained Markov decision processes: stochastic modeling. Routledge.

[3] Barto, A.G., Sutton, R.S., Anderson, C.W., 1983. Neuronlike adaptive elements that can solve difficult learning control problems. IEEE Transactions on Systems, Man, and Cybernetics SMC-13, 834–846.

[4] Basich, C., Svegliato, J., Beach, A., Wray, K.H., Witwicki, S., Zilberstein, S., 2021. Improving competence via iterative state space refinement, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE. pp. 1865–1871.

[5] Basich, C., Svegliato, J., Wray, K.H., Witwicki, S., Biswas, J., Zilberstein, S., 2020. Learning to optimize autonomy in competence-aware systems, in: International Conference on Autonomous Agents and MultiAgent Systems (AAMAS), pp. 123–131.

[6] Beal, J., Rogers, M., 2020. Levels of autonomy in synthetic biology engineering. Molecular Systems Biology 16, e10019. doi:10.15252/msb.202010019.

[7] Beck, A.B., Schwartz, A.D., Fugl, A.R., Naumann, M., Kahl, B., 2015. Skill-based exception handling and error recovery for collaborative industrial robots., in: IROS FinE-R Workshop, pp. 5–10.
[8] Belpaeme, T., Kennedy, J., Ramachandran, A., Scassellati, B., Tanaka, F., 2018. Social robots for education: A review. Science Robotics 3, eaat5954.
[9] Bertsekas, D.P., Tsitsiklis, J.N., 1991. An analysis of stochastic shortest path problems. Mathematics of Operations Research 16, 580–595.
[10] Biswas, J., Veloso, M., 2016. The 1,000-km challenge: Insights and quantitative and qualitative results. IEEE Intelligent Systems 31, 86–96. doi:10.1109/MIS.2016.53.
[11] Blaom, A.D., Kiraly, F., Lienart, T., Simillides, Y., Arenas, D., Vollmer, S.J., 2020. MLJ: A julia package for composable machine learning. Journal of Open Source Software 5, 2704. doi:10.21105/joss.02704.
[12] Bonet, B., Geffner, H., 2003. Labeled rtdp: Improving the convergence of real-time dynamic programming., in: International Conference on Automated Planning and Scheduling (ICAPS), pp. 12–21.
[13] Bradshaw, J.M., Jung, H., Kulkarni, S., Johnson, M., Feltovich, P., Allen, J., Bunch, L., Chambers, N., Galescu, L., Jeffers, R., et al., 2005. Kaa: Policy-based explorations of a richer model for adjustable autonomy, in: International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 214–221.
[14] Bresina, J., Jónsson, A., Morris, P., Rajan, K., 2005. Mixed-initiative activity planning for Mars rovers, in: International Joint Conference on Artificial Intelligence (IJCAI), pp. 1709–1710.
[15] Broggi, A., Bertozzi, M., Fascioli, A., Bianco, C.G.L., Piazzi, A., 1999. The ARGO autonomous vehicle's vision and control systems. International Journal of Intelligent Control and Systems 3, 409–441.
[16] Broggi, A., Cerri, P., Felisa, M., Laghi, M.C., Mazzei, L., Porta, P.P., 2012. The VisLab intercontinental autonomous challenge: An extensive test for a platoon of intelligent vehicles. International Journal of Vehicle Autonomous Systems 10, 147–164. doi:10.1504/IJVAS.2012.051250.
[17] Brown, G., Pocock, A., Zhao, M.J., Luján, M., 2012. Conditional likelihood maximisation: A unifying framework for information theoretic feature selection. The Journal of Machine Learning research 13, 27–66.
[18] Bruemmer, D.J., Few, D.A., Boring, R.L., Marble, J.L., Walton, M.C., Nielsen, C.W., 2005. Shared understanding for collaborative control. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 35, 494–504.
[19] Capobianco, R., Gemignani, G., Iocchi, L., Nardi, D., Riccio, F., Vanzo, A., 2016. Contexts for symbiotic autonomy: Semantic mapping, task teaching and social robotics, in: AAAI Workshop on Symbiotic Cognitive Systems.
[20] Cashmore, M., Fox, M., Larkworthy, T., Long, D., Magazzeni, D., 2014. AUV mission control via temporal planning, in: IEEE International Conference on Robotics and Automation (ICRA), pp. 6535–6541.
[21] Chernova, S., Veloso, M., 2009. Interactive policy learning through confidence-based autonomy. Journal of Artificial Intelligence Research 34, 1–25. doi:10.1613/jair.2584.
[22] Chiou, M., Hawes, N., Stolkin, R., 2021. Mixed-initiative variable autonomy for remotely operated mobile robots. ACM Transactions on Human-Robot Interaction (THRI) 10, 1–34.
[23] Chiou, M., Hawes, N., Stolkin, R., Shapiro, K.L., Kerlin, J.R., Clouter, A., 2015. Towards the principled study of variable autonomy in mobile robots, in: IEEE International Conference on Systems, Man, and Cybernetics (ICSMC), IEEE. pp. 1053–1059.
[24] Clouse, J.A., 1996. On integrating apprentice learning and reinforcement learning. Ph.D. thesis. University of Massachusetts.
[25] Coradeschi, S., Saffiotti, A., 2006. Symbiotic robotic systems: Humans, robots, and smart environments. IEEE Intelligent Systems 21, 82–84. doi:10.1109/MIS.2006.59.
[26] Costen, C., Rigter, M., Lacerda, B., Hawes, N., 2022. Shared autonomy systems with stochastic operator models, in: International Joint Conferences on Artificial Intelligence Organization (IJCAI), pp. 4614–4620.
[27] Das, D., Banerjee, S., Chernova, S., 2021. Explainable ai for robot failures: Generating explanations that improve user assistance in fault recovery, in: ACM/IEEE International Conference on Human-Robot Interaction (HRI), pp. 351–360.
[28] Dickmanns, E.D., 2007. Dynamic vision for perception and control of motion. Springer Science & Business Media.
[29] Dorais, G., Bonasso, R.P., Kortenkamp, D., Pell, B., Schreckenghost, D., 1999. Adjustable autonomy for human-centered autonomous systems, in: IJCAI Workshop on Adjustable Autonomy Systems, pp. 16–35.
[30] Dubois, D.D., 1998. The competency casebook: Twelve studies in competency-based performance improvement. Human Resource Development.
[31] Eliot, L., 2020a. Legal judgment prediction (ljp) amid the advent of autonomous ai legal reasoning. arXiv preprint arXiv:2009.14620 .
[32] Eliot, L., 2020b. An ontological AI-and-law framework for the autonomous levels of AI legal reasoning. arXiv preprint arXiv:2008.07328 .
[33] Ferguson, G., Allen, J.F., Miller, B.W., et al., 1996. TRAINS-95: Towards a mixed-initiative planning assistant, in: AAAI International Conference on Artificial Intelligence Planning Systems (AIPS), pp. 70–77.
[34] Ficuciello, F., Tamburrini, G., Arezzo, A., Villani, L., Siciliano, B., 2019. Autonomy in surgical robots and its meaningful human control. Journal of Behavioral Robotics 10, 30–43. doi:10.1515/pjbr-2019-0002.
[35] Fong, T., Thorpe, C., Baur, C., 2003. Multi-robot remote driving with collaborative control. IEEE Transactions on Industrial Electronics 50, 699–704.
[36] Gao, Y., Chien, S., 2017. Review on space robotics: Toward top-level science through space exploration. Science Robotics 2. doi:10.1126/scirobotics.aan5074.
[37] Ghalamzan, E.A.M., Abi-Farraj, F., Giordano, P.R., Stolkin, R., 2017. Human-in-the-loop optimisation: mixed initiative grasping for optimally facilitating post-grasp manipulative actions, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE. pp. 3386–3393.
[38] Ghallab, M., Nau, D., Traverso, P., 2016. Automated planning and acting. Cambridge University Press.
[39] Gilbert, T.F., 1996. Human Competence: Engineering Worthy Performance.
[40] Greenblatt, N.A., 2016. Self-driving cars and the law. IEEE Spectrum 53, 46–51.
[41] Griffith, S., Subramanian, K., Scholz, J., Isbell, C.L., Thomaz, A.L., 2013. Policy shaping: Integrating human feedback with reinforcement learning. Advances in Neural Information Processing Systems 26.

[42] Hager, P., Gonczi, A., 1996. What is competence? Medical Teacher 18, 15–18. doi:10.3109/01421599609040255.

[43] Hawes, N., Burbridge, C., Jovan, F., Kunze, L., Lacerda, B., Mudrova, L., Young, J., Wyatt, J., Hebesberger, D., Kortner, T., et al., 2017. The STRANDS project: Long-term autonomy in everyday environments. IEEE Robotics & Automation Magazine 24, 146–156. doi:10.1109/MRA.2016.2636359.

[44] Hewitt, C., Politis, I., Amanatidis, T., Sarkar, A., 2019. Assessing public perception of self-driving cars: The autonomous vehicle acceptance model, in: International Conference on Intelligent User Interfaces, pp. 518–527.

[45] Hoff, K.A., Bashir, M., 2015. Trust in automation: Integrating empirical evidence on factors that influence trust. Human Factors 57, 407–434. doi:10.1177/0018720814547570.

[46] Holtz, J., Guha, A., Biswas, J., 2018. Interactive robot transition repair with smt, in: International Joint Conference on Artificial Intelligence (IJCAI), pp. 4905–4911. URL: https://joydeepb.com/Publications/ijcai2018_srtr.pdf, doi:10.24963/ijcai.2018/681.

[47] Huenupán, F., Yoma, N.B., Molina, C., Garretón, C., 2008. Confidence based multiple classifier fusion in speaker verification. Pattern Recognition Letters 29, 957–966. doi:10.1016/j.patrec.2008.01.015.

[48] Jiang, S., Arkin, R.C., 2015. Mixed-initiative human-robot interaction: definition, taxonomy, and survey, in: IEEE International Conference on Systems, Man, and Cybernetics (ICSMC), IEEE. pp. 954–961.

[49] Kaelbling, L.P., Littman, M.L., Moore, A.W., 1996. Reinforcement learning: A survey. Journal of Artificial Intelligence Research (JAIR) 4, 237–285.

[50] Knox, W.B., Breazeal, C., Stone, P., 2012. Learning from feedback on actions past and intended, in: ACM/IEEE International Conference on Human-Robot Interaction, Late-Breaking Reports Session (HRI), Citeseer.

[51] Knox, W.B., Stone, P., Breazeal, C., 2013. Training a robot via human feedback: A case study, in: International Conference on Social Robotics (ICSR), Springer. pp. 460–470.

[52] Kolobov, A., Mausam, Weld, D.S., 2012. A theory of goal-oriented MDPs with dead ends, in: Conference on Uncertainty in Artificial Intelligence (UAI), pp. 438–447.

[53] Kuncheva, L.I., 2004. Combining Pattern Classifiers: Methods and Algorithms. John Wiley & Sons.

[54] Kunz, C., Murphy, C., Singh, H., Pontbriand, C., Sohn, R.A., Singh, S., Sato, T., Roman, C., Nakamura, K., Jakuba, M., et al., 2009. Toward extraplanetary under-ice exploration: Robotic steps in the Arctic. Journal of Field Robotics 26, 411–429. doi:10.1002/rob.20288.

[55] Lin, P., 2016. Why ethics matters for autonomous cars, in: Autonomous driving. Springer, pp. 69–85.

[56] Lysiak, R., Kurzynski, M., Woloszynski, T., 2014. Optimal selection of ensemble classifiers using measures of competence and diversity of base classifiers. Neurocomputing 126, 29–35. doi:10.1016/j.neucom.2013.01.052.

[57] Maurer, M., Gerdes, J.C., Lenz, B., Winner, H., 2016. Autonomous driving: technical, legal and social aspects. Springer.

[58] McQuillin, E., Churamani, N., Gunes, H., 2022. Learning socially appropriate robo-waiter behaviours through real-time user feedback, in: ACM/IEEE International Conference on Human-Robot Interaction (HRI), pp. 541–550.

[59] Meeussen, W., Marder-Eppstein, E., Watts, K., Gerkey, B.P., 2011. Long term autonomy in office environments, in: Robotics: Science and Systems (RSS) Alone Workshop.

[60] Moffitt, V.Z., Franke, J.L., Lomas, M., 2006. Mixed-initiative adjustable autonomy in multi-vehicle operations. Association for Unmanned Vehicle Systems International .

[61] Moreira, I., Rivas, J., Cruz, F., Dazeley, R., Ayala, A., Fernandes, B., 2020. Deep reinforcement learning with interactive feedback in a human–robot environment. Applied Sciences 10, 5574.

[62] Mostafa, S.A., Ahmad, M.S., Mustapha, A., 2019. Adjustable autonomy: A systematic literature review. Artificial Intelligence Review 51, 149–186. doi:10.1007/s10462-017-9560-8.

[63] Mustard, J.F., Beaty, D.W., Bass, D.S., 2013. Mars 2020 science rover: Science goals and mission concept, in: AAS Division for Planetary Sciences Annual Meeting, pp. 211–217.

[64] Parasuraman, R., Sheridan, T.B., Wickens, C.D., 2000. A model for types and levels of human interaction with automation. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 30, 286–297.

[65] Peng, H., Long, F., Ding, C., 2005. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence 27, 1226–1238. doi:10.1109/TPAMI.2005.159.

[66] Petousakis, G., Chiou, M., Nikolaou, G., Stolkin, R., 2020. Human operator cognitive availability aware mixed-initiative control, in: IEEE International Conference on Human-Machine Systems (ICHMS), IEEE. pp. 1–4.

[67] Platanios, E.A., Stretcu, O., Neubig, G., Poczos, B., Mitchell, T.M., 2019. Competence-based curriculum learning for neural machine translation. arXiv preprint arXiv:1903.09848 .

[68] Rabiee, S., Basich, C., Wray, K.H., Zilberstein, S., Biswas, J., 2022. Competence-aware path planning via introspective perception. IEEE Robotics and Automation Letters doi:10.1109/LRA.2022.3145517.

[69] Rabiee, S., Biswas, J., 2019. IVOA: Introspective vision for obstacle avoidance, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1230–1235.

[70] Ramakrishnan, R., Kamar, E., Nushi, B., Dey, D., Shah, J., Horvitz, E., 2019. Overcoming blind spots in the real world: Leveraging complementary abilities for joint execution, in: AAAI conference on Artificial Intelligence (AAAI), AAAI Press. pp. 6137–6145. doi:10.1609/aaai.v33i01.33016137.

[71] Rastrigin, L.A., Erenstein, R.H., 1981. Method of collective recognition. Energoizdat 595, 37.

[72] Rigter, M., Lacerda, B., Hawes, N., 2020. A framework for learning from demonstration with minimal human effort. IEEE Robotics and Automation Letters 5, 2023–2030. doi:10.1109/LRA.2020.2970619.

[73] Roijers, D.M., Vamplew, P., Whiteson, S., Dazeley, R., 2013. A survey of multi-objective sequential decision-making. Journal of Artificial Intelligence Research 48, 67–113.

[74] Rosenstein, M.T., Barto, A.G., 2004. Supervised actor-critic reinforcement learning, in: Si, J., Barto, A.G., Powell, W.B., Wunsch, D. (Eds.), Handbook of Learning and Approximate Dynamic Programming. IEEE Press, pp. 359–380.

[75] Rosenthal, S., Biswas, J., Veloso, M.M., 2010. An effective personal mobile robot agent through symbiotic human-robot interaction., in: International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), pp. 915–922.

[76] SAE On-Road Automated Vehicle Standards Committee, 2014. Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems. SAE Standards Journal 3016, 1–16.

[77] Saffiotti, A., Broxvall, M., Gritti, M., LeBlanc, K., Lundh, R., Rashid, J., Seo, B., Cho, Y.J., 2008. The PEIS-ecology project: Vision and results, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2329–2335.

[78] Saisubramanian, S., Zilbertsein, S., 2019. Adaptive outcome selection for planning with reduced models, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1655–1660.

[79] Sampson, D., Fytros, D., 2008. Competence models in technology-enhanced competence-based learning, in: Handbook on Information Technologies for Education and Training. Springer, pp. 155–177.

[80] Scerri, P., Pynadath, D.V., Tambe, M., 2001. Adjustable autonomy in real-world multi-agent environments, in: International Conference on Autonomous Agents AGENTS, pp. 300–307.

[81] Scerri, P., Pynadath, D.V., Tambe, M., 2002. Towards adjustable autonomy for the real world. Journal of Artificial Intelligence Research 17, 171–228. doi:`10.1613/jair.1037`.

[82] Senliol, B., Gulgezen, G., Yu, L., Cataltepe, Z., 2008. Fast correlation based filter with a different search strategy, in: IEEE International Symposium on Circuits and Systems (ISCIS), pp. 1–4.

[83] Sheridan, T.B., 1992. Telerobotics, Automation, and Human supervisory control. Cambridge, MA: MIT Press.

[84] Smyth, B., McKenna, E., 2001. Competence models and the maintenance problem. Computational Intelligence 17, 235–249. doi:`10.1111/0824-7935.00142`.

[85] Sousa, A., Madureira, L., Coelho, J., Pinto, J., Pereira, J., Sousa, J.B., Dias, P., 2012. LAUV: The man-portable autonomous underwater vehicle. International Federation of Automatic Control 45, 268–274. doi:`10.3182/20120410-3-PT-4028.00045`.

[86] Sternberg, R.J., Kolligian Jr., J., 1990. Competence considered. Yale University Press.

[87] Sutton, R.S., Barto, A.G., 2018. Reinforcement learning: An introduction. MIT press.

[88] Svegliato, J., Nashed, S.B., Zilberstein, S., 2021. Ethically compliant sequential decision making, in: AAAI Conference on Artificial Intelligence (AAAI), pp. 11657–11665.

[89] Svegliato, J., Wray, K., Witwicki, S., Biswas, J., Zilberstein, S., 2019. Belief space metareasoning for exception recovery, in: IEEE/RSJ International Conference on Intelligent Robotics and Systems (IROS), pp. 1224–1229.

[90] Unit, F.E., 1984. Towards a competence-based system: An FEU view.

[91] Vecht, B., 2009. Adjustable autonomy: Controlling influences on decision making. Utrecht University.

[92] Veloso, M., Biswas, J., Coltin, B., Rosenthal, S., Kollar, T., Mericli, C., Samadi, M., Brandao, S., Ventura, R., 2012. Cobots: Collaborative robots servicing multi-floor buildings, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5446–5447.

[93] Veloso, M.M., Biswas, J., Coltin, B., Rosenthal, S., 2015. CoBots: Robust symbiotic autonomous mobile service robots, in: International Joint Conference on Artificial Intelligence (IJCAI), pp. 4423–4429.

[94] Visinsky, M.L., Cavallaro, J.R., Walker, I.D., 1994. Robotic fault detection and fault tolerance: A survey. Reliability Engineering & System Safety 46, 139–158.

[95] Woloszynski, T., Kurzynski, M., 2009. On a new measure of classifier competence applied to the design of multiclassifier systems, in: International Conference on Image Analysis and Processing (ICIAP), pp. 995–1004.

[96] Woloszynski, T., Kurzynski, M., 2011. A probabilistic model of classifier competence for dynamic ensemble selection. Pattern Recognition 44, 2656–2668. doi:`10.1016/j.patcog.2011.03.020`.

[97] Woloszynski, T., Kurzynski, M., Podsiadlo, P., Stachowiak, G.W., 2012. A measure of competence based on random classification for dynamic ensemble selection. Information Fusion 13, 207–213. doi:`10.1016/j.inffus.2011.03.007`.

[98] Woods, K., Kegelmeyer, W.P., Bowyer, K., 1997. Combination of multiple classifiers using local accuracy estimates. IEEE Transactions on Pattern Analysis and Machine Intelligence 19, 405–410. doi:`10.1109/34.588027`.

[99] Wray, K.H., Pineda, L., Zilberstein, S., 2016. Hierarchical approach to transfer of control in semi-autonomous systems, in: International Joint Conference on Artificial Intelligence (IJCAI), pp. 517–523.

[100] Wray, K.H., Zilberstein, S., Mouaddib, A.I., 2015. Multi-objective MDPs with conditional lexicographic reward preferences, in: AAAI conference on Artificial Intelligence (AAAI), pp. 3418–3424.

[101] Yang, G.Z., Cambias, J., Cleary, K., Daimler, E., Drake, J., Dupont, P.E., Hata, N., Kazanzides, P., Martel, S., Patel, R.V., et al., 2017. Medical robotics—regulatory, ethical, and legal considerations for increasing levels of autonomy. Science Robotics 2, 8638. doi:`10.1126/scirobotics.aam8638`.

[102] Zhao, Z., Anand, R., Wang, M., 2019. Maximum relevance and minimum redundancy feature selection methods for a marketing machine learning platform, in: IEEE International Conference on Data Science and Advanced Analytics (DSAA), IEEE. pp. 442–452.

[103] Zieba, S., Polet, P., Vanderhaegen, F., Debernard, S., 2010. Principles of adjustable autonomy: A framework for resilient human-machine cooperation. Cognition, Technology & Work 12, 193–203. doi:`10.1007/s10111-009-0134-7`.

[104] Zilberstein, S., 2015. Building strong semi-autonomous systems, in: AAAI Conference on Artificial Intelligence, pp. 4088–4092.

Appendix A.

In this section we describe additional details of our experimentation. In all of our experiments, our models were solved using LRTDP [12], and the feedback profiles were implemented as random forests using the Julia package DecisionTree.jl in the Julia MLJ framework [11] with default parameters. In our implementation of Algorithm 1, our validation step simply required a Matthews correlation coefficient that was (1) positive (i.e. better than random) on the validation data set and (2) better than the Matthews correlation coefficient of the current feedback profile on the same validation data set (with the discriminator masked out) by at least 0.2.

Gated Exploration

In all experiments, we used the gated exploration strategy as defined in Definition 8. While a variety of different distributions could be used for the exploration strategy, we use an extension of the standard Boltzmann softmax distribution [49] over q-values in the adjacency set of $l \in \mathcal{L}$:

$$P(l') = adj(\kappa(\overline{s}, a), l') \frac{\exp(-q(\overline{s}, (a, l'); \hat{\lambda}))}{\sum_{l'' \in \mathcal{L}} adj(\kappa(\overline{s}, a), l'') \exp(-q(\overline{s}, (a, l''); \hat{\lambda}))} \tag{A.1}$$

where $q(\overline{s}, (a, l); \hat{\lambda}) = \overline{C}(\overline{s}, (a, l)) + \sum_{\overline{s}' \in \overline{S}} \overline{T}(\overline{s}, (a, l), \overline{s}') V(\overline{s}'; \hat{\lambda})$ is the expected cumulative reward when taking action $(a, l) \in \overline{A}$ in state $\overline{s} \in \overline{S}$ conditioned on the current feedback profile $\hat{\lambda}$.

To improve exploration efficiency, we introduce a potential-based mechanism in our experiments in which, for each $\overline{s} \in \overline{S}$ and $a \in A$, we maintain a *potential* for each level $l \in \mathcal{L}$, $\gamma_{\overline{s},a,l}$, which is updated at each level-exploration step, defined as

$$\gamma^{t+1}_{\overline{s},a,l} \leftarrow \begin{cases} 0 & l' \text{ is chosen} \\ \min\left(\gamma^t_{\overline{s},a,l} + P(l), 1\right) & \text{otherwise} \end{cases} \tag{A.2}$$

where γ^t_l is the potential at time t and $P(l)$ is defined in Equation A.1. For readability purposes, define $\gamma^t(\overline{s}, a, l) := \gamma^t_{\overline{s},a,l}$; given this potential function we can slightly alter Equation A.1 to be

$$\hat{P}(l') = adj(l, l') \frac{\exp(\gamma^t(\overline{s}, a, l'))}{\sum_{l'' \in \mathcal{L}} adj(l, l'') \exp(\gamma^t(s, a, l''))} \tag{A.3}$$

which defines a new distribution from which to sample new levels of autonomy to explore.

In our experiments, a potential matrix was initialized for the CAS model and updated each time the autonomy profile was updated via gated exploration. Gated exploration was implemented by sampling from the above distribution to update the autonomy profile for each $(\overline{s}, a)$ input by including the sampled level if not in $\kappa(\overline{s}, a)$ already, and otherwise doing nothing. The "gated" element was simulated in all experiments by observing the likelihood of an override, and adding the highest level (the only level disallowed initially) if sampled if the likelihood is below 0.15 for the AV navigation domain or below 0.05 for the AV obstacle passing domain.

Simulated feedback

All human feedback in our experiments is fully simulated; the feedback of each simulated agent is determined by set rules based on the state and action up to their consistency ϵ. In the other $1 - \epsilon$ part of the time we return a random feedback signal drawn uniformly from the possible feedback signals for the given level of autonomy. Below, we describe the rules behind the simulated feedback in our experiments. The first two cases refer to feedback rules present across all simulated humans for the base domain. The rest of the cases refer to feedback rules present for specific simulated humans. Note that all feedback rules mentioned directly correspond to competences of no autonomy when the human would override or disapprove an action, and unsupervised autonomy otherwise; there is no situation in our domain where the optimal action to perform is in verified or supervised autonomy given a perfect model of the human's feedback.

Autonomous Vehicle Navigation The human overrides or disapproves overtaking an obstruction in an edge state when there is only a single lane, preferring to it themselves. When making a right turn at an intersection, which is considered a generally safe maneuver, the human overrides the maneuver if there is on occlusion, a pedesrtrian, and at least one other vehicle, indicating the presence of numerous other actors in a potentially chaotic environment. When going straight, making a left turn, or making a U-turn at an intersection, which are considered more challenging maneuvers as all potential cross-traffic must be considered, the human will override if there is an occlusion limiting visibility and a pedestrian or more than one vehicle, or if there is a pedestrian and more than two vehicles even without an occlusion limiting visibility. In all other cases, the human approves or does not override the system's behavior.

Autonomous Vehicle Obstacle Passing The human overrides the action `Stop` if the AV is fully in the oncoming lane or if they can see that there is no oncoming vehicle. The human overrides the action `Edge` if the AV has visibility of oncoming traffic as the AV should either commit to the overtake (if safe to do so) or stop and wait until the overtake is safe. Finally, the human overrides the action `Go` if there is no visibility of oncoming traffic, or if there *is* oncoming traffic and the AV does not have priority to go.

AV Navigation – Cautious The human overrides or disapproves the vehicle from acting at all if the weather is snowy and it is night time, preferring to drive the whole way in these conditions. The human overrides or disapproves the overtake of a vehicle if it is snowy, or if it is rainy, nighttime, and a two-lane road. At intersections, the human also prefers to take control if it is rainy and nighttime.

AV Navigation – Conscientious The human overrides or disapproves the vehicle's maneuver if there is a trailing vehicle when overtaking an obstruction, or if there is a trailing vehicle when the AV is at an intersection and either takes the `Wait` action or otherwise if there is at least one additional vehicle at the intersection, to hurry the AV through the intersection.

AV Obstacle Passing – Cautious The human overrides the vehicle if it is either snowy or rainy and nighttime, as the human does not trust the AV to handle the potentially dangerous maneuver in these conditions where the human feels less sure of what the AV can detect.

AV Obstacle Passing – Conscientious The human overrides the vehicle if there is a trailing vehicle and the vehicle takes the action `Stop`, or is stuck waiting with a trailing vehicle and takes the action `Edge`, as they feel socially pressured to execute the overtake expediently by the presence of the trailing vehicle.

AV Obstacle Passing – Rushed The human overrides the vehicle if it is stuck waiting, takes the action `Stop`, or if the vehicle has priority but does not take the action `Go`.