

Bellman Value Decomposition for Task Logic in Safe Optimal Control

William Sharpless^{*,1}, Oswin So^{*,2}, Dylan Hirsch¹, Sylvia Herbert¹, Chuchu Fan²
¹UCSD, ²MIT, ^{*}Equal contribution, wsharpless@ucsd.edu

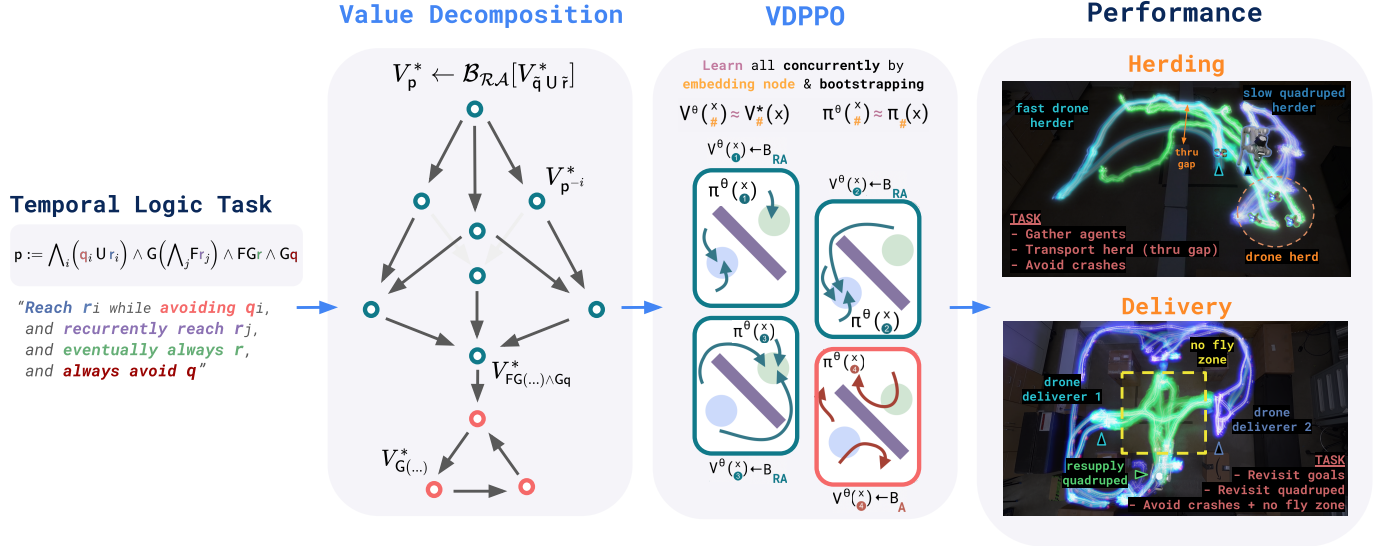


Fig. 1: **Value-Decomposition and VDPPO.** The Bellman Value for a range of temporal logic (e.g., multi-goal, recurrence, staying, safety) decomposes into a Value graph connected by atomic Bellman equations (Thms. 1–4). We propose VDPPO, an algorithm that exploits this structure to learn policies for complex, high-dimensional tasks. Our approach is validated on hardware with Herding and Delivery, two complex tasks involving a heterogeneous team of drones and a quadruped.

Abstract—Real-world tasks involve nuanced combinations of goal and safety specifications. In high dimensions, the challenge is exacerbated: formal automata become cumbersome, and the combination of sparse rewards tends to require laborious tuning. In this work, we consider the innate structure of the Bellman Value as a means to naturally organize the problem for improved automatic performance. Namely, we prove the Value for a complex task defined in temporal logic can be decomposed into a graph of Values, connected by a set of well-known Bellman equations (BEs): the Reach-Avoid BE, the Avoid BE, and a novel type, the Reach-Avoid-Loop BE. To solve the Value and optimal policy, we propose VDPPO, which embeds the decomposed Value graph into a two-layer neural net, bootstrapping the implicit dependencies. We conduct a variety of simulated and hardware experiments to test our method on complex, high-dimensional tasks involving heterogeneous teams and nonlinear dynamics. Ultimately, we find this approach greatly improves performance over existing baselines, balancing safety and liveness automatically. <https://willsharpless.github.io/valdec-site/>

I. INTRODUCTION AND RELATED WORK

Popular modern approaches to robotic learning, like Reinforcement Learning (RL), typically optimize expected cumulative reward [1], making them ill-suited for safety-critical or temporally structured tasks that require worst or best-case satisfaction over long horizons. Such objectives are naturally expressed using Temporal Logic (TL) [2], but TL itself does not prescribe how to act. Moreover, efforts to combine RL & TL tend to face

a trade-off between sparse binary rewards that slow learning and hand-crafted dense rewards that can misalign objectives.

Hamilton–Jacobi Reachability (HJR) [3, 4] provides optimal controllers for basic safety and liveness tasks via max–min Bellman equations, yielding dense and informative learning signals. Recent work showed that certain TL tasks can be solved exactly by decomposing their Value functions into sequences of simple HJR problems [5]. As depicted in Fig. 1 we generalize this idea to a broad class of TL specifications by introducing a Value function decomposition algebra and a corresponding algorithm VDPPO, and demonstrate effectiveness in simulation and real-world drone and quadruped experiments.

Constrained, Multi-Objective, and Goal-Conditioned RL. Constrained Markov Decision Processes (CMDPs) maximize discounted rewards subject to constraints, typically via Lagrangian relaxation [6–19], but require careful tuning and are ill-suited to general TL objectives. Multi-objective RL instead Pareto-optimizes multiple reward sums [20–26], yet does not naturally encode TL structure. Goal-conditioned RL learns policies over a family of goals [27–33, 30, 31, 34], but differs fundamentally from TL settings, where all specifications must be jointly satisfied.

RL with TL specifications. A large body of works studies RL with TL specifications [35–40], including approaches based

on Non-Markovian Reward Decision Processes [41–44, 2], approximated quantitative semantics [45–47], modified Bellman equations [48–50], or multiple discounted rewards [51–53]. In contrast, our method exactly decomposes TL Value functions into simpler Values for which explicit Bellman equations exist, avoiding semantic approximation and long-horizon reward sparsity. See Appendix A and S for additional discussion.

Hamilton–Jacobi Reachability. HJR was originally developed to compute Value functions for Reach, Avoid, and Reach-Avoid problems in continuous time and space [3, 4], corresponding to the quantitative semantics of Eventually, Always, and Until predicates [54]. Recent work has successfully integrated HJR into RL frameworks [55–60]. Our work builds on these results by decomposing Value functions for complex TL objectives into sequences of simpler HJR problems.

II. CONTRIBUTIONS

- 1) We establish a formal connection between Temporal Logic and the Bellman Value by deriving algebraic rules for admissible operations, dubbed the **Value Decomposition Rules (VDR)** (Thm. 1).
- 2) We use the **VDR** to prove that, for a broad class of predicates, the Value decomposes into a graph of atomic BE (Thms. 2, 3): the Avoid BE, Reach–Avoid BE, and a novel Reach–Avoid–Loop BE for recurrence specifications (Def. 6, Thm. 4).
- 3) We introduce **VDPPPO**, an algorithm that embeds the decomposed Value graph (DVG) in two *two-layer neural nets*, and demonstrate its success through extensive simulation and real-world hardware experiments, improving speed and achievement over baselines.

III. PRELIMINARIES

Given a discrete-time system $x_{t+1} = f(x_t, a_t)$ with finite states $x_t \in \mathcal{X} \subseteq \mathbb{R}^n$ and action $a_t \in \mathcal{A} \subseteq \mathbb{R}^m$, a trajectory beginning at x is a sequence of states $\xi_x^\alpha := (x, \dots) \in \mathbb{X} := \mathcal{X}^\mathbb{N}$ arising from actions $\alpha = (a, \dots) \in \mathbb{A} := \mathcal{A}^\mathbb{N}$. We let $\xi_x(t)$ and $\alpha(t)$ be the state and action at time t .

To specify desired properties of a trajectory, let an atomic proposition (AP) $r : \mathbb{R}^n \rightarrow \{\text{true}, \text{false}\}$ be defined by a bounded predicate function $r : \mathbb{R}^n \rightarrow \mathbb{R}$, also known as a target or reward function in HJR or RL. Given a trajectory and time (ξ_x, t) , r is satisfied, written $(\xi_x, t) \models r$, iff $r(\xi_x(t)) \geq 0$, and thus, r represents the arrival of a trajectory at a region (defined by the 0-level-set of r). Let q with q denote an AP to maintain, s.t. the complement represents an unsafe region.

To represent complex tasks, TL defines a language for the composition of predicates [61]. Predicates may be composed via negation (NOT, $\neg$), conjunction (AND, $\wedge$), the **Until** operator (U) and the **next** operator (NEXT, X). With these operations, one may also define disjunction (OR, $\vee$), **finally/eventually** (F), and **globally/always** (G). All operators may be defined via a robustness function $\rho : \mathbb{R}^n \rightarrow \mathbb{R}$ [62], a scalar measure of satisfaction, equivalent to the payoff in HJB optimal control [3, 4].

Definition 1. For any predicate p , let the robustness score $\rho[p] : \mathbb{X} \times \mathbb{N} \rightarrow \mathbb{R}$ be defined inductively with the following rules.

Satisfaction via ρ

$$(\xi_x, t) \models p \iff \rho[p](\xi_x, t) \geq 0$$

Atomic prop.

$$\begin{aligned} \rho[r](\xi_x, t) &:= r(\xi_x(t)) \\ \rho[q](\xi_x, t) &:= q(\xi_x(t)) \end{aligned}$$

NOT

$$\rho[\neg p](\xi_x, t) := -\rho[p](\xi_x, t)$$

NEXT

$$\rho[Xp](\xi_x, t) := \rho[p](\xi_x, t+1)$$

AND

$$\begin{aligned} \rho[p \wedge p'](\xi_x, t) &:= \\ \min \{ \rho[p](\xi_x, t), \rho[p'](\xi_x, t) \} \end{aligned}$$

OR

$$\begin{aligned} \rho[p \vee p'](\xi_x, t) &:= \\ \max \{ \rho[p](\xi_x, t), \rho[p'](\xi_x, t) \} \end{aligned}$$

Finally/Eventually

$$\rho[Fp](\xi_x, t) := \max_{\tau \geq t} \rho[p](\xi_x, \tau)$$

Globally/Always

$$\rho[Gp](\xi_x, t) := \min_{\tau \geq t} \rho[p](\xi_x, \tau)$$

Until

$$\rho[p \text{ U } p'](\xi_x, t) := \max_{\tau \geq t} \min \left\{ \rho[p'](\xi_x, \tau), \min_{\kappa \in [t, \tau]} \rho[p](\xi_x, \kappa) \right\}$$

With this syntax, one may express the satisfaction of complex specifications over trajectories formally and succinctly¹. Importantly, we use the robustness score ρ because this is the payoff used in the corresponding HJR optimal control problem [3, 4] (see Appendix B for details).

IV. PROBLEM FORMULATION

In this work, we consider the problem of synthesizing optimal actions α , such that for any initial state x the resulting trajectory ξ_x^α maximizes the payoff ρ for a given predicate. We later extend this to a policy $\pi : \mathcal{X} \rightarrow \mathcal{A}$ (Appendix D). We assume the system begins at $t = 0$ and evolves indefinitely. For brevity, we let $\rho[p](\xi) := \rho[p](\xi, 0)$. This leads to the following infinite-horizon Optimal Control Problem (OCP),

$$\begin{aligned} \text{maximize}_\alpha \quad & \rho[p](\xi_x^\alpha), \\ \text{s.t.} \quad & \xi_x^\alpha(t+1) = f(\xi_x^\alpha(t), \alpha(t)). \end{aligned}$$

Note, because ρ is defined by temporal extrema (max/min over time), this program induces behavior characterized by its *outlying performance*, in contrast with a sum-based OCP (in canonical RL [11]) which selects for average behavior. This objective is explicitly captured by the Bellman Value function V^* , the “high score” function for the given infinite-horizon OCP.

Definition 2. For a predicate p , we aim to solve

$$V^*[p](x) := \max_\alpha \rho[p](\xi_x^\alpha), \quad (1)$$

which gives the optimal ρ over infinite action sequences from any given initial state.

¹Note that $F r = T \text{ U } r$, where T is true, and thus it often suffices to consider only U and G in analysis. Similarly, $G q \wedge G q' = G(q \wedge q') = G q''$ so we write always specifications succinctly. Also note, U is defined per HJR convention to include the arrival time s.t. $\kappa \in [t, \tau]$, not $\kappa \in [t, \tau)$.

We define the Value for a general TL predicate p (V_p^* for brevity), but for basic operations, this object has been studied in-depth [3, 4, 63]. Namely, the Reach ($\mathcal{R}$), Avoid ($\mathcal{A}$) and Reach-Avoid ($\mathcal{RA}$) Values – for respectively (1) reaching a target, (2) avoiding an obstacle, and (3) avoiding an obstacle until a target is reached – correspond to the Values for F , G , and U .

Reach Value

$$V^*[Fr](x) = \max_{\alpha} \max_t r(\xi_x^{\alpha}(t))$$

Avoid Value

$$V^*[Gq](x) = \max_{\alpha} \min_t q(\xi_x^{\alpha}(t))$$

Reach-Avoid Value

$$V^*[qU r](x) = \max_{\alpha} \max_t \min \left\{ r(\xi_x^{\alpha}(t)), \min_{\kappa \leq t} q(\xi_x^{\alpha}(\kappa)) \right\}$$

In this context, the following Bellman operations for these extrema-based Values have been derived [57, 56] (note, $\mathcal{R}$ is a special case of $\mathcal{RA}$).

Definition 3. Let the $\mathcal{A}$ and $\mathcal{RA}$ Bellman operators be

$$\mathcal{B}_{\mathcal{A}}^{\gamma}[V] := (1 - \gamma)q + \gamma \min\{V^+, q\},$$

$$\mathcal{B}_{\mathcal{RA}}^{\gamma}[V] := (1 - \gamma) \min\{r, q\} + \gamma \min\{\max\{r, V^+\}, q\},$$

where $V^+(x) := \max_a V(f(x, a))$.

These operators are of crucial importance as they are contractive and yield the namesake Value in the limit of discounting [57, 56]. In other words, for the $\mathcal{A}$ -Value $V^*[Gq]$ and the $\mathcal{RA}$ -Value $V^*[qU r]$ defined by [1], we have that

$$V^{\gamma}[Gq] \leftarrow \mathcal{B}_{\mathcal{A}}^{\gamma}[V^{\gamma}[Gq]], \quad V^{\gamma}[qU r] \leftarrow \mathcal{B}_{\mathcal{RA}}^{\gamma}[V^{\gamma}[qU r]],$$

are unique fixed points, satisfying $\lim_{\gamma \rightarrow 1} V^{\gamma} = V^*$ [57, 56]. Unlike canonical RL [1], these operators propagate extrema of r and q , encouraging behavior defined by outlying performance.

In a recent work [5], it was demonstrated that for simple conjunctions $Fr \wedge Gq$ (reach a goal and avoid obstacles) and $Fr_1 \wedge Fr_2$ (reach two goals), one may decompose the corresponding Values into the “atomic” BE, which in some ways resembles the base case for what follows. In this work, we generalize this principle, demonstrating that the $\mathcal{A}$ -BE and $\mathcal{RA}$ -BE, along with the novel Reach-Avoid-Loop BE (Lem. 4), serve as a set of “atomic” building blocks to decompose the Value of complex TL predicates.

V. OPTIMALITY VERSUS SATISFACTION

The decomposition of formal logic is well-studied in several contexts, including formal verification [64], automata theory [65], and temporal logic trees (TLT) [66]. This body of work has established a rich framework for understanding the structure and properties of temporal logic formulas, and has led to performant decompositional learning methods for complex tasks [67]. However, the algebra of TL, which is equivalent to the algebra over the robustness score, is fundamentally distinct from that of the Value function due to the presence of the maximum over action sequences or control policies in [2]. This distinction is not only relevant to theoretical analysis but can lead to safety failures and sub-optimality in real world applications. We illustrate this with the following remark and offer concrete counterexamples in Appendix C.

Remark 1. The following TL identity always holds:

$$\rho[Fr \wedge Gq](\xi_x^{\alpha}) = \min\{\rho[Fr](\xi_x^{\alpha}), \rho[Gq](\xi_x^{\alpha})\}.$$

By contrast, for the corresponding Value, we have

$$V^*[Fr \wedge Gq](x) \leq \min\{V^*[Fr](x), V^*[Gq](x)\},$$

where the inequality is indeed strict when no single choice of action sequence can both reach r and avoid $\neg q$.

VI. RESULTS

We present our main results regarding the decomposition of the Value for complex TL predicates. We begin by discussing the relationship between the Value and TL algebras, and then present the decompositional results. In general, we seek to express the Value for a complex predicate in terms of simpler components that are themselves composed with the fundamental Bellman equations. All proofs of the results are in Appendix D.

A. Agreeable Algebra

We begin by clarifying the similarity between the decomposition of the Value and TL algebra. Although not immediately apparent, these rules will serve as the foundation of complex decompositions presented later. The presence of the $\max_{\alpha}$ in [1] does not always distinguish the Value algebra from that of TL, for example, when the TL is also defined by maxima, as with OR and a “right”-U (for which, F is a special case). In any case, when $\max_{\alpha}$ commutes over the given ρ , a decomposition follows that mirrors that of the TL. To state these rules, it is first important consider the atomic proposition associated with a given Value.

Definition 4. Let $v[p]$ be the AP for $V^*[p]$, s.t.

$$\rho[v[p]](\xi_x, t) := V^*[p](\xi_x(t)).$$

We now give the algebraic properties of Value decomposition.

Theorem 1. Value Decomposition Rules (VDR)

For general p and p' , the following holds:

- 1) $V^*[pU p'](x) = V^*[pU v[p']](x)$ (right-U)
- 2) $V^*[p \vee p'](x) = V^*[v[p] \vee v[p']](x)$ (OR)
- 3) if $\forall \alpha, \beta \in \mathcal{A}^{\mathbb{N}}, \rho[p](\xi_x^{\alpha}) = \rho[p](\xi_x^{\beta})$,
 $V^*[p \wedge p'](x) = V^*[v[p] \wedge v[p']](x)$ (weak-AND)
- 4) $V^*[Xp](x) = V^*[Xv[p]](x)$ (NEXT)

Intuitively, the VDR offer four ways to decompose the Value by “pushing it” through the logic. In fact, each symbolic manipulation corresponds to distributing the infinite action sequence argument to the logical components of the total predicate. Then by definition, the residual objects are the component Values (see Appendix D.1 for proof).

Together, this set of operations makes some compositions simple to consider. For example, one may see the Value for FG, i.e. the “Reach-Stay” task, readily decomposes by VDR-1,

$$V^*[FGq](x) = V^*[F\tilde{r}](x), \quad \tilde{r} := v[Gq]. \quad (2)$$

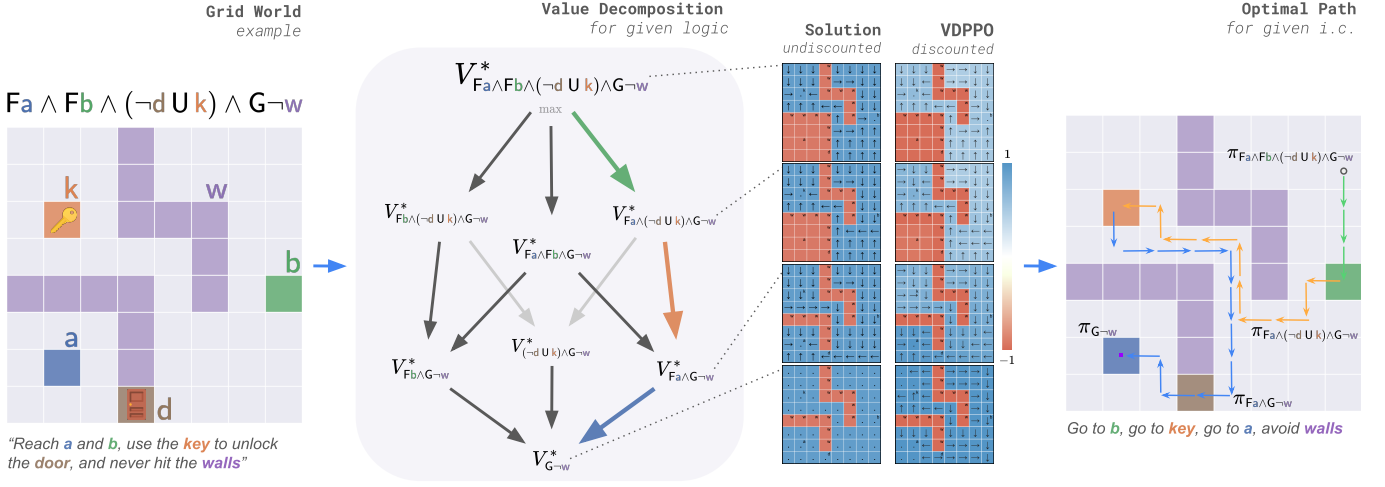


Fig. 2: **Multi-Reach-Avoid Value Decomposition.** Here we illustrate the primary decomposition result (Thm. 2), with a GridWorld example (left) for a given specification. The corresponding DVG is shown (center left) with each node representing a decomposed Value, and edges representing dependencies. In the center right, a subset of decomposed Values solved with dynamic programming are shown, along with the discounted solution produced by VDPPO. On the right, the optimal path for a given initial condition is shown.

Hence, this Value may be solved by, first, solving the $\mathcal{A}$ -Value for q , $V^*[Gq]$, and then, second, using this to define the target $\tilde{r} := \mathcal{V}[Gq]$ of a $\mathcal{R}$ -Value, $V^*[F\tilde{r}]$. More generally, we may know that the Value for a series of Until predicates is equivalent to a chain of $\mathcal{RA}$ Values (Cor. 2 in Appendix D.1).

This example illustrates that decomposition via the $\mathcal{VDR}$ transforms a Value with no apparent Bellman equation into a hierarchy of Values composed of “atomic” Bellman equations (the DVG), amounting to a *macro-scale* dynamic programming process. These results, however, do not directly apply when the TL is defined by min as with AND, and thus are insufficient for many common predicates. The solution will be to transform the logic into a special, amenable form. We show several examples of this transformation in the following sections.

B. Multi-Reach-Avoid Value Decomposition

We next present the first major result of the work concerning the decomposition of the Value for the AND of N Until predicates with a safety spec., or equivalently, the N - $\mathcal{RA}$ -A Value. This is a generalization of the dual-objective base cases that preceded this work [5]. Akin to later results, the decomposition is achieved by transforming the logic into an amenable form, and then applying the algebraic rules of Thm. 1 to decompose the Value into a single $\mathcal{RA}$ -Value.

Theorem 2. For the predicate $p := \bigwedge_{i \in \mathcal{I}} (q_i \cup r_i) \wedge Gq$ with $N = |\mathcal{I}|$, the corresponding Value satisfies

$$\begin{aligned} V^* \left[\bigwedge_i (q_i \cup r_i) \wedge Gq \right] (x) \\ = V^* \left[\underbrace{\left(\bigwedge_i q_i \wedge q \right)}_{\tilde{q}} \cup \underbrace{\left(\bigvee_i (r_i \wedge \mathcal{V}[p_{-i}]) \right)}_{\tilde{r}} \right] (x) \end{aligned}$$

where $p_{-i} := \bigwedge_{j \in \mathcal{I} \setminus \{i\}} q_j \cup r_j \wedge Gq$.

Proof. We first transform p into an amenable form (Lem. 4),

$$p := \bigwedge_i (q_i \cup r_i) \wedge Gq \equiv \underbrace{\left(\bigwedge_i q_i \wedge q \right)}_{\tilde{q}} \cup \left(\bigvee_i (r_i \wedge p_{-i}) \right).$$

We may then simply apply $\mathcal{VDR}$ 1, $\mathcal{VDR}$ 2, and $\mathcal{VDR}$ 3

$$\begin{aligned} V^*[p](x) &= V^* \left[\tilde{q} \cup \mathcal{V}[\bigvee_i (r_i \wedge p_{-i})] \right] (x), \\ &= V^* \left[\tilde{q} \cup \left(\bigvee_i \mathcal{V}[r_i \wedge p_{-i}] \right) \right] (x), \\ &= V^* \left[\tilde{q} \cup \left(\bigvee_i (r_i \wedge \mathcal{V}[p_{-i}]) \right) \right] (x). \quad \square \end{aligned}$$

This result gives an equivalence between the N - $\mathcal{RA}$ Value and the Value function of a single $\mathcal{RA}$ Value, which has abstract reach and avoid predicates $\tilde{r}$ & $\tilde{q}$ in the sense that they no longer represent physical goals or obstacles. Instead, the new reach predicate $\tilde{r}$ is defined by the OR of N AND’s that each correspond to reaching one of the predicates r_i and being able to satisfy the remaining logic p_{-i} , i.e. having $V^*[p_{-i}] \geq 0$. The new avoid predicate $\tilde{q}$ is defined by the AND of all N avoid predicates and hence implies that we need to avoid all q_i . Intuitively, Thm. 2 breaks down the optimal Value into the goal of safely reaching any of the predicates while being able to satisfy the rest of the predicate of $N-1$ Until’s, denoted p_{-i} , where r_i & q_i have been “popped off” the original predicate.

Notably, Thm. 2 can be applied recursively, and, therefore, we may reapply it iteratively to break the N - $\mathcal{RA}$ Value into N decomposable sub-Values and so forth, giving $2^N - 1$ Values in total. Crucially, as each of these Values is equivalent to a special Until Value, they may each be solved with the discounted $\mathcal{RA}$ -BE with their respective rewards. We demonstrate this result in Fig. 2 with a simple GridWorld problem, where the true solution may be solved via dynamic programming.

C. Recurrent Reach-Avoid Value Decompositions

In this section, we consider the family of recurrence relation operations corresponding to the composition of G with U (for

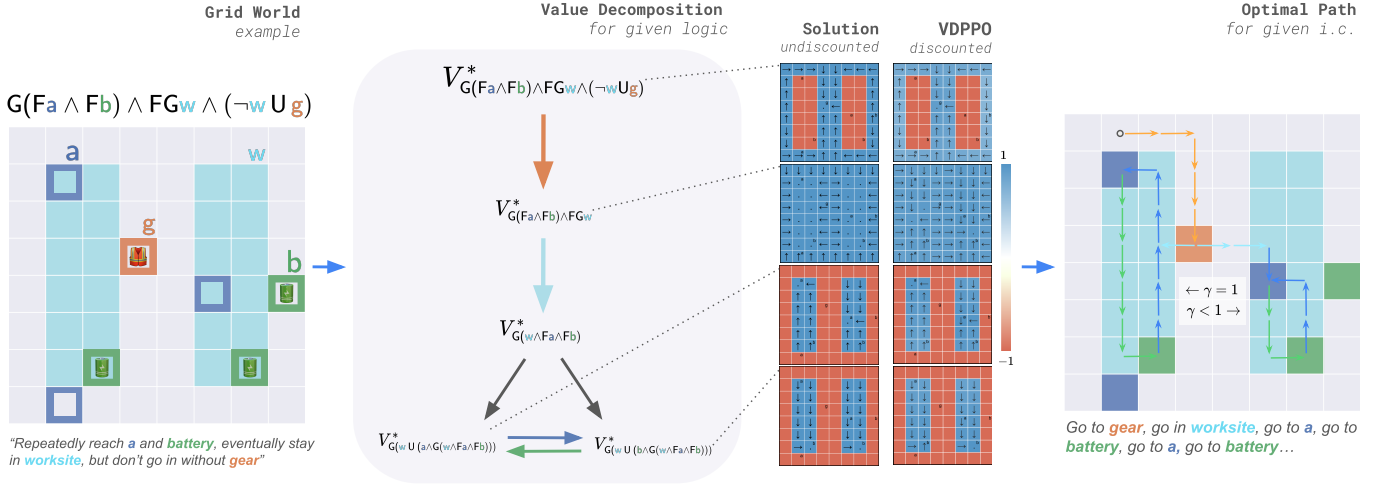


Fig. 3: **E.g. Recurrent Reach-Avoid Value Decomposition.** We illustrate the recursive decomposition result (Thm. 3), with a GridWorld example (left) for a given specification. The plots here are analogous to those of Fig. 2 with the DVG (center left), decomposed Values (center right), and optimal path (right). Note, the optimal path for the discounted case differs due to the subtle effect of discounting the Value associated with a G composition, which selects for shorter loops (Appendix F).

which GF is a special case). To always-eventually satisfy a predicate implies that a trajectory must continue to satisfy it indefinitely. These compositions are particularly important as they encompass the liveness (i.e., goal-satisfaction) property, where certain states must be revisited or regenerated in some sense. Moreover, this operation is significantly less strict than the Reach-Stay specification FG in (2), and thus more desirable, when continuous goal achievement is impossible or unknown.

The temporal coupling of the outer G with the inner TL makes the Value of these compositions behave differently. Note, by definition, the robustness function alone satisfies

$$\rho[\text{GF}r](\xi_x) = \min_{t \in \mathbb{N}} \max_{t' \geq t} \rho[r](\xi_x, t') = \limsup_{s \rightarrow \infty} \rho[r](\xi_x, s), \quad (3)$$

and, hence, the Value is constant along trajectories, regardless of when they begin. Yet, this coupling also gives rise to an implicit characterization amenable to the VDR: e.g. $p := G(q \cup r)$ satisfies $p \equiv q \cup (r \wedge Xp)$, thus, we may apply VDR-1[3][4] to have the following.

Lemma 1. For the predicate $p := G(q \cup r)$ the corresponding Value satisfies

$$V^*[p](x) = V^*[q \cup (r \wedge Xv[p])](x).$$

This result demonstrates that the Value function associated with the predicate $G(q \cup r)$ can be characterized implicitly. For intuition, one may consider this Value as a special RA Value that aims to reach an intersection of the target predicate r and its own satisfiable set (denoted by $v[p]$) at the next step, and hence, maintain the ability to satisfy it again in the future. More generally, we may expand this result to the case involving a composition of G with N-AND-Until's by considering a loop of predicates p_i^ℓ .

Definition 5. Given a predicate, $p := G(\bigwedge_{i \in \mathcal{I}} (q_i \cup r_i))$, let a loop of $N := |\mathcal{I}|$ predicates p_i^ℓ be defined² s.t. $p_{N+1}^\ell := p_1^\ell$ and

$$p_i^\ell := \left(\bigwedge_{j \in \mathcal{I}} q_j \right) \cup \left(r_i \wedge Xp_{i+1}^\ell \right). \quad (4)$$

Theorem 3. For predicate $p := G(\bigwedge_{i \in \mathcal{I}} (q_i \cup r_i))$, the corresponding Value satisfies $\forall i$,

$$\begin{aligned} V^* \left[G \left(\bigwedge_i (q_i \cup r_i) \right) \right] (x) \\ = V^* [p_i^\ell] (x) = V^* \left[\underbrace{\left(\bigwedge_{j \in \mathcal{I}} q_j \right)}_q \cup \underbrace{\left(r_i \wedge Xv[p_{i+1}^\ell] \right)}_r \right] (x). \end{aligned}$$

where p_i^ℓ and p_{i+1}^ℓ are defined in Def. 5

Proof. Here, the amenable logical form is the loop of predicates p_i^ℓ : $\forall i$, we have $p = p_i^\ell$ (Lem. 6). We may then apply VDR-1, VDR-3 and VDR-4

$$\begin{aligned} V^*[p_i^\ell](x) &= V^* \left[\bar{q} \cup v[r_i \wedge Xp_{i+1}^\ell] \right] (x), \\ &= V^* \left[\bar{q} \cup (r_i \wedge v[Xp_{i+1}^\ell]) \right] (x), \\ &= V^* \left[\bar{q} \cup (r_i \wedge Xv[p_{i+1}^\ell]) \right] (x). \quad \square \end{aligned}$$

This result allows us to consider the problem of recurrently reach-avoiding N tasks as a loop of N coupled Values. Perhaps bizarrely, the implicit nature implies that the loop order is arbitrary (discounting in Def. 6, however, gives higher Value to shorter loops, see Appendix F).

Although these results appear similar to the previous decompositions, it is important to note that they are fundamentally different due to the implicit definition. These characterizations do *not* offer a method to solve the Value,

²This depends on the inclusion of the end-point in U (Def. 1) following HJR convention, which implies $p \equiv G(\bigwedge_i Fr_i \wedge \bigwedge_i q_i)$. Otherwise, let $p_i^\ell := (\bigwedge_j q_j \wedge \tilde{w}_i) \cup (r_i \wedge \tilde{w}_i \wedge Xp_{i+1}^\ell)$ where $\tilde{w}_i := \bigwedge_{j \neq i} (q_j \wedge r_j)$.

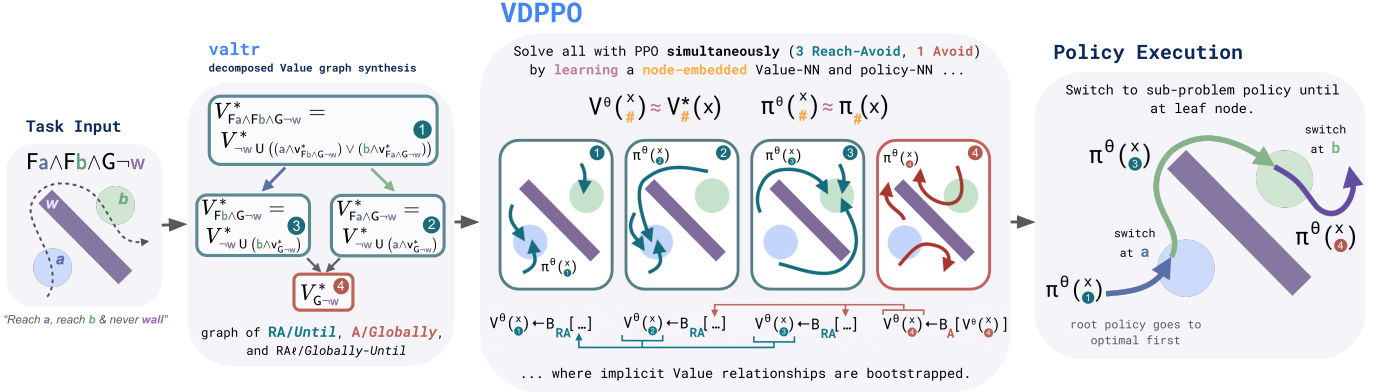


Fig. 4: **Graphical Depiction of valtr and VDPP0.** TL is inputted, e.g., reaching two goals and avoiding a wall: $Fa \wedge Fb \wedge G \neg w$ (far left). Next, valtr decomposes the logic based on the VDR, yielding a graph of $\mathcal{RA}$, $\mathcal{A}$, and $\mathcal{RA}_\ell$ nodes, e.g., 4 Values with 1 $\mathcal{A}$ leaf and 3 $\mathcal{RA}$ parents (mid-left). Next, VDPP0 learns an actor and critic for all Values simultaneously by embedding the graph. The Bellman update for each node bootstraps the corresponding children (mid-right). The policy is executed beginning at the root node and switches upon predicate satisfaction (far right).

and in some state spaces, may be meaningless. To certify the well-posedness in certain scenarios (e.g. finite state spaces), we show in Appendix C that these Values are equivalent to the limit of finite recurrence, however, this is not generally a practical procedure. Moreover, a straightforward application of the discounted $\mathcal{RA}$ -BE yields a BE that is *not necessarily contractive*, due to the appearance of the Value in both $(1 - \gamma)$ and γ terms. To address these challenges, we propose a novel contractive Bellman Equation, which we call the Reach-Avoid-Loop ($\mathcal{RA}_\ell$) BE, which is guaranteed to yield the desired Value in the limit of discounting.

Definition 6. Given a set of N values $\{V_i\}^N$, let the $\mathcal{RA}_\ell$ Bellman operator be defined for each element as

$$\mathcal{B}_{\mathcal{RA}_\ell}^\gamma[V_i] := (1 - \gamma) \min\{r_i, \min_j q_j\} + \gamma \min \left\{ \max \left\{ \min \{r_i, V_{i+1}^+\}, V_i^+ \right\}, \min_j q_j \right\}.$$

Theorem 4. The $\mathcal{RA}_\ell$ -BE given in Def. 6 is contractive such that for a set of N values $V^\gamma[p_i^\ell]$,

$$V^\gamma[p_i^\ell] \leftarrow \mathcal{B}_{\mathcal{RA}_\ell}^\gamma[V^\gamma[p_i^\ell]]$$

is a unique fixed point, satisfying $\forall i$,

$$\lim_{\gamma \rightarrow 1} V^\gamma[p_i^\ell] = V^* \left[G \left(\bigwedge_i (q_i \cup r_i) \right) \right].$$

Equipped with the $\mathcal{RA}_\ell$ -BE, we can now approximate the Value for the family of recurrence predicates effectively.

D. General Decomposition and Policy

In general, the VDR may be used to decompose the Value beyond the cases derived in Thms. 2 & 3. These were presented for their practical relevance, and because they serve as examples of a general proof strategy, however, they are not the limit of the VDR. For example, using the same techniques, it is simple to consider a combination of the two classes as presented in Fig. 1, which we give in Appendix E. Moreover, in Appendix F we demonstrate how this strategy can be used to synthesize a corresponding policy.

VII. ALGORITHM(S)

In this section, we introduce Value-Decomposition PPO, a variant of PPO that solves the Value associated with the class of TL predicates in Sec. VI using the decomposed Value graph (DVG). We also describe the tools required to generate the DVG and to solve it via dynamic programming for low-dimensional problems. A graphical overview is shown in Fig. 4.

valtr: Generating the DVG. We introduce valtr, a tool that converts a parsed temporal logic specification into the general predicate form of Thm. 5 by recursively applying standard TL rules into a form suitable for the VDR. This representation is then transformed into the DVG, where nodes correspond to predicates, negations, max, min, and Value functions, and edges encode their dependencies. Cyclic recurrence compositions are handled via a special node, enabling efficient parsing and transformation of arbitrary predicates into DVGs. See Appendix J for details.

Dynamic Programming with the DVG. With the DVG, one may compute the Value of a given predicate by performing a topological sort of the DVG and applying dynamic programming to compute the Value of each subformula in the correct order. Cyclic nodes, arising from recurrence specifications, can be approximate by a finite number of recurrences as described in Appendix G. This allows us to compute the dynamic programming solution for the low-dimensional test cases given in Figs. 2 and 3.

Policy Inference with the DVG. To execute the policy, we keep track of the current node in the DVG initialized at the root node. At each step, we execute the policy associated with the current node and check its satisfaction. If the predicate is satisfied, we switch to the child node corresponding to the satisfied predicate and repeat this process. We provide more details in Appendix I. For an explicit construction and rigorous proof of the optimality of the constructed policy, see [68].

VDPP0. Finally, we propose Value-Decomposition PPO (VDPP0), a special variant of PPO which solves the Value associated with the class of TL predicates in Sec. VI by using the DVG. VDPP0 uses one two-layer neural net (NN) for all Values in the DVG and one for all corresponding policies, by embed-

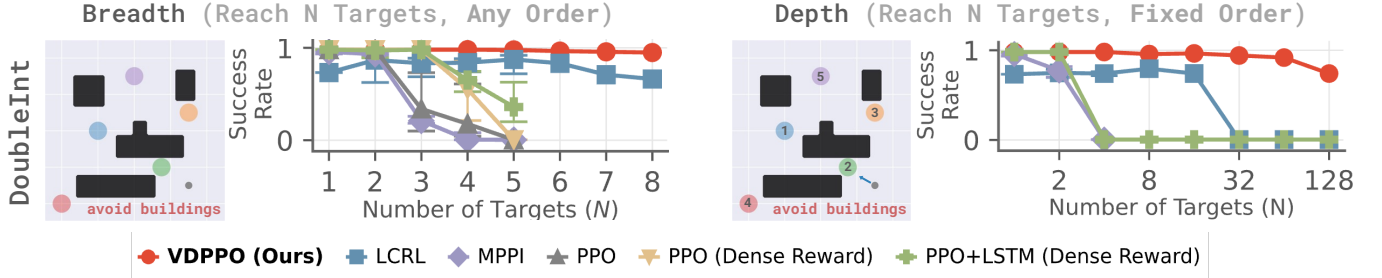


Fig. 5: **Sim. Performance Scaling with TL Complexity.** On the left, the robustness of VDPPO with respect breadth of a multi-reach spec. is assessed with a N - $\mathcal{RA}$ - $\mathcal{A}$ spec. (Thm. 2) in which the agent must reach N targets and avoid crashing into the buildings. Both the DVG and automata scale exponentially with breadth, however, standard methods with additive rewards prove the worst. On the right, depth of a multi-reach spec. is assessed with a chain of N specs. and a safety spec. Here, the linear topology of the DVG allows VDPPO to scale to higher N .

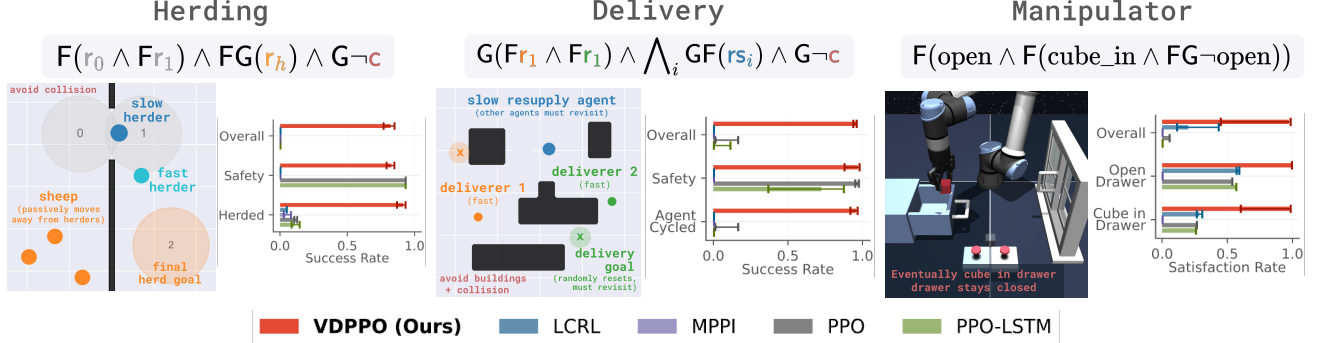


Fig. 6: **Sim. Performance on Complex, High-Dimensional Tasks.** Here, we assess VDPPO on more realistic tasks involving complex specifications, namely the Herding (30D, left), Delivery (18D, middle), and Manipulator (39D, right) environments. The first two tasks involve heterogeneous teams and a wide breadth of specifications, which leads to extraneous caution in baselines. The latter task involves a single agent with complex grasping and collision dynamics and a simpler series of tasks, yielding better baseline performance initially but poor sequential performance. In all cases, VDPPO excels, automatically breaking down complex tasks into learnable sub-tasks via the DVG and graph embedding.

ding the node representations with a one-hot vector. VDPPO solves all nodes in the DVG simultaneously by conducting rollouts for each node with the corresponding embedded policy. Depending on the embedding value, the target and advantage for each node are computed with the corresponding discounted $\mathcal{A}$ -BE, $\mathcal{RA}$ -BE or $\mathcal{RA}_\ell$ -BE. To train all nodes concurrently, at this step VDPPO uses the current child node Value estimate for the corresponding BE (“bootstraps”), which is represented by the feedback loop in Fig. 4. The policy is trained with the standard PPO objective, using the advantage estimate corresponding to the embedding. This allows us to leverage the decomposed structure of the Value functions to efficiently learn policies that satisfy complex TL specifications without sequentially approximating the Value. See Appendix K for further details.

VIII. SIMULATION RESULTS

To better understand the performance of VDPPO, we design simulation experiments to answer the following questions:

- (Q1): Does Value decomposition help satisfy & optimize a wider spec. with more options?
- (Q2): Does Value decomposition help satisfy & optimize a longer horizon spec.?
- (Q3): Can VDPPO scale to more complex dynamics?

Details on experiments are in Appendix L (sim details) and Appendix M (baselines). Additional ablation studies are provided in Appendix N.

A. Setup

Environments. We evaluate on four simulated domains: DoubleInt (toy double integrator environment to focus on TL challenges), Herding (a team of herders collaborates to herd multiple targets to a designated location while avoiding obstacles), Delivery (agents must continuously pick up and deliver packages to a special agent while avoiding collisions with each other and static obstacles), and Manipulator (a robotic arm interacts with a cube and a drawer as specified by TL formulas) from OGBench’s scene task [69].

Baselines. We compare VDPPO with other model-free methods that can solve TL specifications with black-box dynamics. These include 1) LCRL [70], a deep RL method that solves TL tasks by augmenting the state space with an automata representation of the TL formula, 2) an extension of Model Predictive Path Integral (MPPI) [71] to tackle TL problems [72], TL-MPPI, 3) PPO using the robustness metric as a reward, and 4) an extension of PPO using a Long Short-Term Memory backbone [73], PPO-LSTM. For each environment, LCRL, PPO, PPO-LSTM and VDPPO are run for the same number of update steps, while for TL-MPPI we follow the hyperparameters chosen in [72].

Evaluation criteria. Performance is measured by success rate on finite-horizon TL satisfaction; we additionally report satisfaction rates of individual subformulas. All methods are trained with three seeds and evaluated on 256 initial conditions.

B. Results

(Q1): Value decomposition improves breadth scalability with TL complexity. To answer this question, we study the effect of an increasing N - $\mathcal{RA}$ - $\mathcal{A}$ spec., i.e. a N -AND of Until in a single-agent double integrator environment. The spec. requires an agent to visit N targets in any order while maintaining safety. The problem is difficult because the number of paths grows exponentially. All methods solve the singular spec. but degrade as the number of specifications increases. VDPPPO consistently outperforms all baselines as the breadth complexity of the TL spec. increases (Fig. 5, left), demonstrating the effectiveness of Value decomposition in handling complex TL tasks.

(Q2): Value decomposition improves depth scalability with TL complexity. To answer this question, we study the effect of an increasing N -chain-of- $\mathcal{RA}$ spec., i.e. a N -chain of Until predicates in a single-agent double integrator environment. The spec. requires an agent to visit N targets in a fixed order while maintaining safety. Again, VDPPPO consistently outperforms all baselines as the depth complexity of the TL spec. increases (Fig. 5, right), achieving up to 128 targets reliably. This is likely because the probability of randomly satisfying nested TL specifications decreases exponentially with depth, making it difficult for non-decompositional methods to learn effective policies.

(Q3): VDPPPO outperforms on problems with difficult dynamics. We now consider more challenging problems, either due to complex interactions with uncontrolled agents (Herding), heterogeneous collaboration (Delivery & Herding), or complex dynamics (Manipulator), and show the results in Fig. 6. In all three tasks, VDPPPO achieves the highest success rate by a significant margin. See Appendix N for ablations.

IX. HARDWARE RESULTS

Lastly, we perform hardware experiments corresponding to the Herding and Delivery environments using a swarm of Crazyflie (CF) drones collaborating with the Unitree Go2 to demonstrate the ability of VDPPPO to solve complex task specifications in high-dimensional real-world settings with heterogeneous collaboration. See Fig. 7 for an overview.

A. Herding

In this experiment, we consider a team of one CF and the Go2 tasked with herding three “sheep” CFs through a narrow gap to a target location while avoiding obstacles and collisions. The sheep CFs have a fixed nominal policy, using the softmin to drive them away from the nearest obstacle or agent, and thus will move only when approached.

The TL specification for the task is given by,

$$\mathbf{P}_{\text{herding}} := \mathbf{F}(r_0 \wedge \mathbf{F}r_1) \wedge \mathbf{FG}(r_h) \wedge \mathbf{G}\neg\mathbf{c}, \quad (5)$$

where $\mathbf{c}$ denotes collisions, r_0 the herd reaching the pre-gate region, r_1 passage through the gate, and r_h arrival at the target. This encodes a sequence of reach–avoid objectives followed by a reach–stabilize objective requiring indefinite herding.

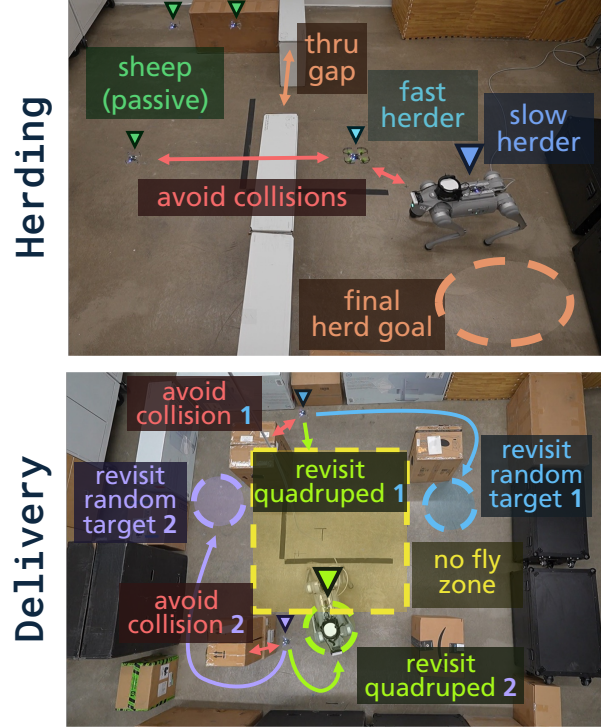


Fig. 7: Hardware Overview for Herding and Delivery Tasks

The herders (CF and Go2) are initialized opposite the gap from the herd and have asymmetric dynamics, with the Go2 moving more slowly. To satisfy the specification, the herders must coordinate to pass through the gap, collect the herd, and guide it to the target while avoiding obstacles. We train a VDPPPO policy using the DVG and deploy it on hardware, where the agents adapt online to real-time state feedback.

Ultimately, we observe that the CF and Go2 **learn to divide the labor** of the task such that the CF passes through the gap to gather the agents (Fig. 8B), while the Go2 waits to receive on the herding side (Fig. 8C). When the herd passes through the narrow gap, the Go2 initially moves out of the way (Fig. 8C) and then transitions to providing support, rapidly shifting position to block the Herd from distributing across the new space (Fig. 8E). This behavior is entirely emergent and demonstrates the wide-ranging ability of VDPPPO to solve complex tasks automatically.

B. Delivery

In this experiment, we consider a team of two CFs and the Go2 tasked with recurrently visiting agent-specific target locations and recurrently revisiting the Go2 agent (to model package delivery and resupply), while avoiding building obstacles, collisions, and a “no fly zone” (for the CFs).

The TL specification for the task is given by,

$\mathbf{P}_{\text{delivery}} := \mathbf{G}(\mathbf{F}r_1 \wedge \mathbf{F}r_2) \wedge \bigwedge_i \mathbf{GF}(rs_i) \wedge \mathbf{G}\neg(\mathbf{ac} \vee \mathbf{ob} \vee \mathbf{nf})$ where the predicate r_1 & r_2 capture CF i visiting target i , rs_i captures CF i visiting the Go2 (resupplying), $\mathbf{ac}$ captures aerial collision, $\mathbf{ob}$ captures obstacle collision, and $\mathbf{nf}$ captures the no-fly-zone (for the CFs only). Here, the task logic is dominated by GF, and hence is largely solved with the $\mathcal{RA}_\ell$ -BE.

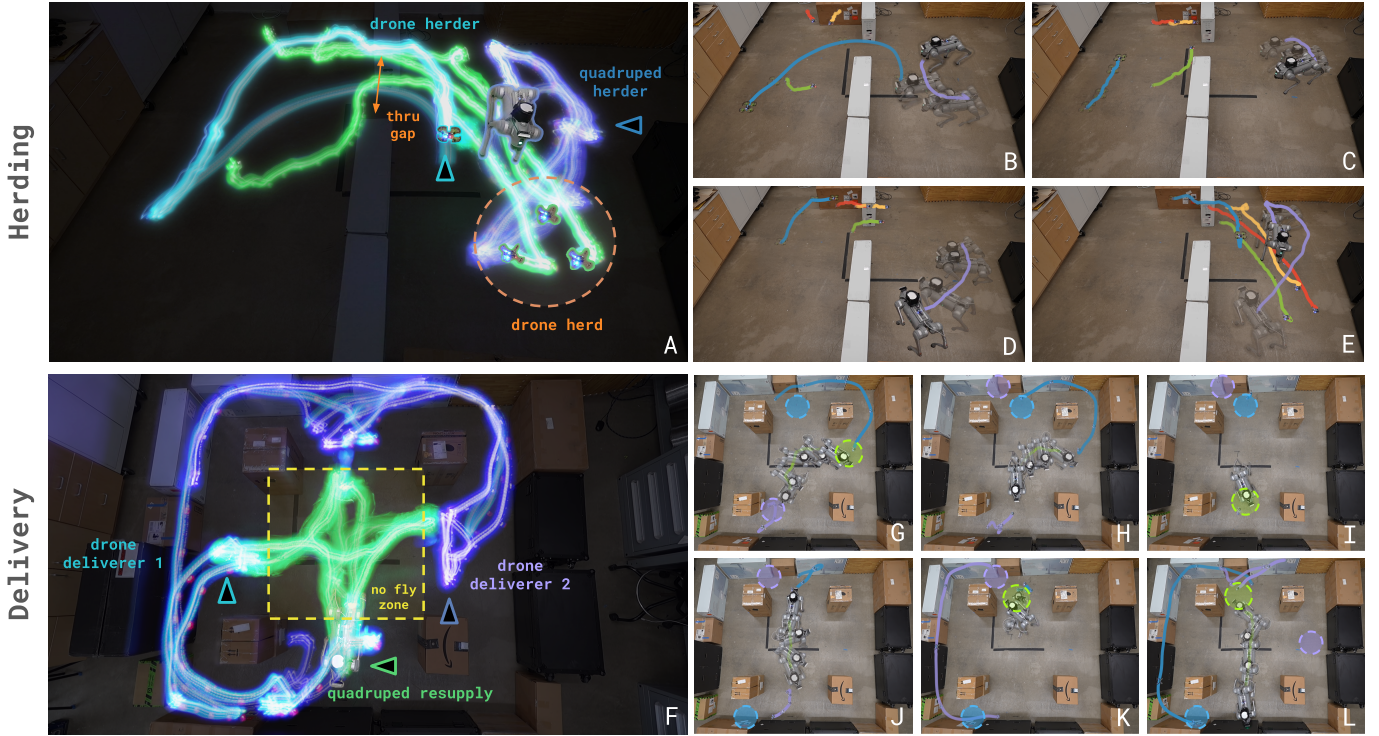


Fig. 8: **Trajectory snapshots from Herding and Delivery hardware tasks.** We show a long-exposure photo (left), and stills from independent times (right), with depictions corresponding to Fig. 7. Videos can be seen at <https://willsharpless.github.io/valdec-site/>

In this environment, the CF targets jump to a new random location after an agent has visited it, requiring a policy that is conditioned to various target locations. The real difficulty of this problem arises in the tightness of the layout; the obstacles confine the Go2 to the central area where the CFs are not allowed to fly (modeling a busy intersection), yet the CFs must visit the Go2 to “resupply.” We again implement VDPP0 to learn a policy to solve the complex task and deploy it live.

Ultimately, we observe sophisticated coordination between the three agents to **distribute the difficulty of the task fairly**. Namely, as the CFs move around the outskirts of the arena, avoiding one another carefully but not slowly (Fig. 8L), the Go2 anticipates their movements, moving between each of the agents (Fig. 8G-I) to be in position to resupply them as close to their target as possible. This complex collaboration generated by VDPP0 allows the agents to rapidly meet deliveries and resupply without crashing at all.

X. DISCUSSION AND LIMITATIONS

We briefly discuss some limitations of the proposed work and future directions.

1. *Any-Order Task Scaling:* Thm. 2 yields a graph of 2^N Values, i.e. the compute for a simultaneous any-order task grows exponentially. This Traveling Salesman Problem like growth is fundamental and also arises in automaton construction [74]. Thus, LCRL and other RL-automata methods share this limitation. Approximations that only consider a subgraph could mitigate this, for example, but we leave this to future work.

2. *Running Cost Minimization:* VDPP0 does not minimize a running cost, but can be extended to do so (i.e., a constrained

optimization problem formulation) via Lagrangian [75] or epigraph formulations [76, 55]. Similarly, VDPP0 might be extended to generate Lyapunov/Barrier functions with an anti-discounting term [77]. We leave this to future work.

3. *Optimizing Robustness, Tradeoffs:* Dense robustness scores can induce unintended behaviors while maintaining satisfaction (e.g., reaching the target then moving far away to increase safety margin). However, the presented theory is amenable to any predicate function with the same level set, e.g. allowing for tuning like $r' := 10r$ without sacrificing “priority” of r . Conveniently, this allows for both dense & sparse reward (and penalty) functions; in our results, we mostly use sparse/binary predicate functions and employ pseudo-binary dense functions for rare tasks.

4. *Stochasticity:* The presented decompositions depend on max/min algebra which does not hold when wrapped with the Expectation operator (all equivalences become inequalities). As discussed in [55], a policy gradient result is still derivable, however, and has been demonstrated to work in practice [5].

XI. CONCLUSION

In this work, we propose a novel approach to solving the Bellman Value function associated with complex temporal specifications via decomposition. Namely, we demonstrate that for a large class of TL predicates, the corresponding Value may be decomposed into a graph of Values connected by a set of “atomic” Bellman equations. With this perspective, we propose VDPP0 that is shown to solve optimal policies in complex tasks well beyond existing methods. This work highlights a novel and powerful approach to tackling complex task logic for real-world autonomy.

ACKNOWLEDGEMENTS

The authors would like to thank Sander Tonkens, Gioele Zardini, and Jaime Fisac for the valuable discussions that lead to the development of the work. This work was supported by the National Science Foundation (NSF) CAREER Award #CCF-2238030 and Office of Naval Research Award (ONR) ONR N000142412661.

REFERENCES

- [1] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018.
- [2] A. Camacho, R. Toro Icarte, T. Q. Klassen, R. Valenzano, and S. A. McIlraith, “LTL and beyond: Formal languages for reward function specification in reinforcement learning,” in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. California: International Joint Conferences on Artificial Intelligence Organization, 1 Aug. 2019, pp. 6065–6073.
- [3] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, “A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games,” *IEEE Transactions on automatic control*, vol. 50, no. 7, pp. 947–957, 2005.
- [4] J. F. Fisac, M. Chen, C. J. Tomlin, and S. S. Sastry, “Reach-avoid problems with time-varying dynamics, targets and constraints,” in *Hybrid Systems: Computation and Control*. ACM, 2015.
- [5] W. Sharpless, D. Hirsch, S. Tonkens, N. Shinde, and S. Herbert, “Dual-objective rl ...” *ICLR*, 2026.
- [6] E. Altman, *Constrained Markov decision processes: Stochastic modeling*. Boca Raton: Routledge, 13 Dec. 2021.
- [7] J. Achiam, D. Held, A. Tamar, and P. Abbeel, “Constrained policy optimization,” *ICML*, vol. abs/1705.10528, pp. 22–31, 30 May 2017.
- [8] A. Wachi and Y. Sui, “Safe reinforcement learning in constrained Markov decision processes,” *ICML*, vol. 119, pp. 9797–9806, 12 Jul. 2020.
- [9] A. Stooke, J. Achiam, and P. Abbeel, “Responsive safety in reinforcement learning by PID lagrangian methods,” *ICML*, vol. 119, pp. 9133–9143, 8 Jul. 2020.
- [10] T. Li, Z. Guan, S. Zou, T. Xu, Y. Liang, and G. Lan, “Faster algorithm and sharper analysis for constrained Markov decision process,” *Oper. Res. Lett.*, vol. 54, no. 107107, p. 107107, May 2024.
- [11] Y. Chen, J. Dong, and Z. Wang, “A primal-dual approach to constrained Markov decision processes,” *arXiv [math.OC]*, 26 Jan. 2021.
- [12] S. Miryoosefi and C. Jin, “A simple reward-free approach to constrained reinforcement learning,” *ICML*, vol. abs/2107.05216, pp. 15 666–15 698, 12 Jul. 2021.
- [13] T.-Y. Yang, J. Rosca, K. Narasimhan, and P. J. Ramadge, “Projection-based constrained policy optimization,” *arXiv [cs.LG]*, 7 Oct. 2020.
- [14] D. Ding, K. Zhang, T. Başar, and M. Jovanović, “Natural policy gradient primal-dual method for constrained Markov decision processes,” *Neural Inf Process Syst*, vol. 33, pp. 8378–8390, 2020.
- [15] C. Tessler, D. J. Mankowitz, and S. Mannor, “Reward constrained policy optimization,” *arXiv [cs.LG]*, 28 May 2018.
- [16] A. Gattami, Q. Bai, and V. Aggarwal, “Reinforcement learning for constrained Markov decision processes,” *AISTATS*, vol. 130, pp. 2656–2664, 2021.
- [17] H. Satija, P. Amortila, and J. Pineau, “Constrained Markov decision processes via backward value functions,” *ICML*, vol. 119, pp. 8502–8511, 12 Jul. 2020.
- [18] A. Castellano, H. Min, E. Mallada, and J. A. Bazerque, “Reinforcement learning with almost sure constraints,” in *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*, ser. Proceedings of Machine Learning Research, vol. 168. PMLR, 2022, pp. 559–570.
- [19] J. McMahan and X. Zhu, “Anytime-constrained reinforcement learning,” in *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, S. Dasgupta, S. Mandt, and Y. Li, Eds., vol. 238. PMLR, 02–04 May 2024, pp. 4321–4329.
- [20] M. A. Wiering, M. Withagen, and M. M. Drugan, “Model-based multi-objective reinforcement learning,” in *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*. IEEE, Dec. 2014, pp. 1–6.
- [21] M. K. Van and A. Nowé, “Multi-objective reinforcement learning using sets of Pareto dominating policies,” *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3483–3512, 2014.
- [22] X.-Q. Cai, P. Zhang, L. Zhao, J. Bian, M. Sugiyama, and A. Llorens, “Distributional Pareto-optimal multi-objective reinforcement learning,” *Neural Inf Process Syst*, vol. 36, pp. 15 593–15 613, 2023.
- [23] H. Mossalam, Y. M. Assael, D. M. Roijers, and S. Whiteson, “Multi-objective deep reinforcement learning,” *arXiv [cs.AI]*, 9 Oct. 2016.
- [24] A. Abels, D. Roijers, T. Lenaerts, A. Nowé, and D. Steckelmacher, “Dynamic weights in multi-objective deep reinforcement learning,” in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 2019, pp. 11–20.
- [25] R. Yang, X. Sun, and K. Narasimhan, “A generalized algorithm for multi-objective reinforcement learning and policy adaptation,” in *Advances in Neural Information Processing Systems*. proceedings.neurips.cc, 2019.
- [26] E. Liu, Y.-C. Wu, X. Huang, C. Gao, R.-J. Wang, K. Xue, and C. Qian, “Pareto set learning for multi-objective reinforcement learning,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 18, 2025.
- [27] M. Liu, M. Zhu, and W. Zhang, “Goal-conditioned reinforcement learning: Problems and solutions,” *arXiv [cs.AI]*, 20 Jan. 2022.
- [28] M. Plappert, M. Andrychowicz, A. Ray, B. McGrew,

- B. Baker, G. Powell, J. Schneider, J. Tobin, M. Chocie, P. Welinder, V. Kumar, and W. Zaremba, "Multi-goal reinforcement learning: Challenging robotics environments and request for research," *arXiv [cs.LG]*, 26 Feb. 2018.
- [29] Z. Ren, K. Dong, Y. Zhou, Q. Liu, and J. Peng, "Exploration via hindsight goal generation," *Neural Inf Process Syst*, vol. 32, pp. 13 464–13 474, 1 Jun. 2019.
- [30] J. Y. Ma, J. Yan, D. Jayaraman, and O. Bastani, "Offline goal-conditioned reinforcement learning via f-advantage regression," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 310–323.
- [31] A. Campero, R. Raileanu, H. Küttler, J. B. Tenenbaum, T. Rocktäschel, and E. Grefenstette, "Learning with AMIGo: Adversarially motivated intrinsic goals," *arXiv [cs.LG]*, 22 Jun. 2020.
- [32] A. R. Trott, S. Zheng, C. Xiong, and R. Socher, "Keeping your distance: Solving sparse reward tasks using self-balancing shaped rewards," *Neural Inf Process Syst*, vol. abs/1911.01417, 4 Nov. 2019.
- [33] B. Eysenbach, T. Zhang, R. Salakhutdinov, and S. Levine, "Contrastive learning as goal-conditioned reinforcement learning," *Neural Inf Process Syst*, vol. abs/2206.07568, pp. 35 603–35 620, 15 Jun. 2022.
- [34] E. Chane-Sane, C. Schmid, and I. Laptev, "Goal-conditioned reinforcement learning with imagined subgoals," *ICML*, vol. abs/2107.00541, pp. 1430–1440, 1 Jul. 2021.
- [35] M. H. Cohen, Z. Serlin, K. Leahy, and C. Belta, "Temporal logic guided safe model-based reinforcement learning: A hybrid systems approach," *Nonlinear Anal. Hybrid Syst.*, vol. 47, no. 101295, p. 101295, Feb. 2023.
- [36] W. Qiu, W. Mao, and H. Zhu, "Instructing goal-conditioned reinforcement learning agents with temporal logic objectives," *Neural Inf Process Syst*, vol. 36, pp. 39 147–39 175, 2023.
- [37] T. Brázdil, K. Chatterjee, M. Chmelík, V. Forejt, J. Křetínský, M. Kwiatkowska, D. Parker, and M. Ujma, "Verification of Markov decision processes using learning algorithms," *arXiv [cs.LO]*, 10 Feb. 2014.
- [38] N. Hamilton, P. K. Robinette, and T. T. Johnson, "Training agents to satisfy timed and untimed signal temporal logic specifications with reinforcement learning," in *Software Engineering and Formal Methods*, ser. Lecture notes in computer science. Cham: Springer International Publishing, 2022, pp. 190–206.
- [39] D. Sadigh, E. S. Kim, S. Coogan, S. S. Sastry, and S. A. Seshia, "A learning based approach to control synthesis of Markov decision processes for linear temporal logic specifications," in *53rd IEEE Conference on Decision and Control*. IEEE, Dec. 2014, pp. 1091–1096.
- [40] A. K. Bozkurt, Y. Wang, M. M. Zavlanos, and M. Pajic, "Control synthesis from linear temporal logic specifications using model-free reinforcement learning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2020, p. 10349–10355.
- [41] F. Bacchus, C. Boutilier, and A. J. Grove, "Rewarding behaviors," in *Proceedings of the National Conference on Artificial Intelligence*. cs.toronto.edu, 4 Aug. 1996, pp. 1160–1167.
- [42] S. Thiebaux, C. Gretton, J. Slaney, D. Price, and F. Kabanza, "Decision-theoretic planning with non-Markovian rewards," *J. Artif. Intell. Res.*, vol. 25, pp. 17–74, 29 Jan. 2006.
- [43] A. Camacho, O. Chen, S. Sanner, and S. McIlraith, "Non-Markovian rewards expressed in LTL: Guiding search via reward shaping," *Proceedings of the International Symposium on Combinatorial Search*, vol. 8, no. 1, pp. 159–160, 1 Sep. 2021.
- [44] R. T. Icarte, T. Q. Klassen, R. Valenzano, and S. A. McIlraith, "Using reward machines for high-level task specification and decomposition in reinforcement learning," *ICML*, vol. 80, pp. 2112–2121, 3 Jul. 2018.
- [45] D. Aksaray, A. Jones, Z. Kong, M. Schwager, and C. Belta, "Q-learning for robust satisfaction of signal temporal logic specifications," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, Dec. 2016, pp. 6565–6570.
- [46] X. Li, C.-I. Vasile, and C. Belta, "Reinforcement learning with temporal logic rewards," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Sep. 2017, pp. 3834–3839.
- [47] M. Cai, M. Hasanbeig, S. Xiao, A. Abate, and Z. Kan, "Modular deep reinforcement learning for continuous motion planning with temporal logic," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, p. 7973–7980, Oct. 2021. [Online]. Available: <http://dx.doi.org/10.1109/LRA.2021.3101544>
- [48] R. Wang, P. Zhong, S. S. Du, R. R. Salakhutdinov, and L. Yang, "Planning with general objective functions: Going beyond total rewards," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 14 486–14 497.
- [49] W. Cui and W. Yu, "Reinforcement learning with non-cumulative objective," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 1, pp. 124–137, 2023.
- [50] Y. Tang, Y. Zhang, J. Ackermann, Y.-J. Zhang, S. Nishimori, and M. Sugiyama, "Recursive reward aggregation," in *Reinforcement Learning Conference*, 2025.
- [51] H. van Seijen, M. Fatemi, J. Romoff, R. Larochelle, T. Barnes, and J. Tsang, "Hybrid reward architecture for reinforcement learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 5398–5408.
- [52] S. Pitis, "Consistent aggregation of objectives with diverse time preferences requires non-markovian rewards," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

- [53] Z. Lin, D. Yang, L. Zhao, T. Qin, G. Yang, and T.-Y. Liu, "Rd²: Reward decomposition with representation decomposition," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 11 298–11 308.
- [54] M. Chen, Q. Tam, S. C. Livingston, and M. Pavone, "Signal temporal logic meets reachability: Connections and applications," in *International Workshop on the Algorithmic Foundations of Robotics*. Springer, 2018, pp. 581–601.
- [55] O. So, C. Ge, and C. Fan, "Solving minimum-cost reach avoid using reinforcement learning," in *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. [Online]. Available: <https://openreview.net/forum?id=jzngdJQ2lY>
- [56] K.-C. Hsu, V. Rubies-Royo, C. J. Tomlin, and J. F. Fisac, "Safety and liveness guarantees through reach-avoid reinforcement learning," in *Proceedings of Robotics: Science and Systems*, Held Virtually, July 2021.
- [57] J. F. Fisac, N. F. Lugovoy, V. Rubies-Royo, S. Ghosh, and C. J. Tomlin, "Bridging hamilton-jacobi safety analysis and reinforcement learning," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8550–8556.
- [58] M. Ganai, C. Hirayama, Y.-C. Chang, and S. Gao, "Learning stabilization control from observations by learning lyapunov-like proxy models," *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023.
- [59] D. Yu, H. Ma, S. Li, and J. Chen, "Reachability constrained reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2022, pp. 25 636–25 655.
- [60] K. Zhu, F. Lan, W. Zhao, and T. Zhang, "Safe multi-agent reinforcement learning via approximate hamilton-jacobi reachability," *J. Intell. Robot. Syst.*, vol. 111, no. 1, 30 Dec. 2024.
- [61] O. Maler and D. Nickovic, "Monitoring temporal properties of continuous signals," in *International symposium on formal techniques in real-time and fault-tolerant systems*. Springer, 2004, pp. 152–166.
- [62] A. Donzé and O. Maler, "Robust satisfaction of temporal logic over real-valued signals," in *International conference on formal modeling and analysis of timed systems*. Springer, 2010, pp. 92–106.
- [63] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, "Hamilton-jacobi reachability: A brief overview and recent advances," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 2017, pp. 2242–2253.
- [64] C. Baier and J.-P. Katoen, *Principles of model checking*. MIT press, 2008.
- [65] O. Grumberg, E. Clarke, and D. Peled, "Model checking," in *International Conference on Foundations of Software Technology and Theoretical Computer Science; Springer: Berlin/Heidelberg, Germany*, 1999.
- [66] G. Bombara, C.-I. Vasile, F. Penedo, H. Yasuoka, and C. Belta, "A decision tree approach to data classification using signal temporal logic," in *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control*, 2016, pp. 1–10.
- [67] Y. Meng, F. Chen, and C. Fan, "Tgpo: Temporal grounded policy optimization for signal temporal logic tasks," *arXiv preprint arXiv:2510.00225*, 2025.
- [68] O. So, W. Sharpless, S. Herbert, and C. Fan, "Value functions for temporal logic: Optimal policies and safety filters," *arXiv preprint arXiv:2605.01051*, 2026.
- [69] S. Park, K. Frans, B. Eysenbach, and S. Levine, "Ogbench: Benchmarking offline goal-conditioned rl," in *International Conference on Learning Representations (ICLR)*, 2025.
- [70] M. Hasanbeig, D. Kroening, and A. Abate, "Lcrl: Certified policy synthesis via logically-constrained reinforcement learning," in *International Conference on Quantitative Evaluation of SysTems*. Springer, 2022, pp. 217–231.
- [71] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 1433–1440.
- [72] P. Halder, H. Homburger, L. Kiltz, J. Reuter, and M. Althoff, "Trajectory planning with signal temporal logic costs using deterministic path integral optimization," *arXiv preprint arXiv:2503.01476*, 2025.
- [73] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [74] J. Cichon, A. Czubak, and A. Jasinski, "Minimal büchi..." in *DepCoS*, 2009. [Online]. Available: <https://dl.acm.org/doi/abs/10.1109/DepCoS-RELCOMEX.2009.31>
- [75] M. Ganai, Z. Gong, C. Yu, S. Herbert, and S. Gao, "Iterative reachability estimation for safe reinforcement learning," *NeurIPS*, 2023. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2023/file/dca63f2650fe9e88956c1b68440b8ee9-Paper-Conference.pdf
- [76] O. So and C. Fan, "Solving stabilize-avoid optimal control via epigraph form and deep reinforcement learning," in *Proceedings of Robotics: Science and Systems*, 2023. [Online]. Available: <https://www.roboticsproceedings.org/rss19/p085.pdf>
- [77] J. J. Choi, D. Lee, K. Sreenath, C. J. Tomlin, and S. L. Herbert, "Robust Control Barrier-Value Functions for Safety-Critical Control," in *Conference on Decision and Control (CDC)*. IEEE, 2021.
- [78] L. Yifru and A. Baheri, "Concurrent learning of control policy and unknown safety specifications in reinforcement learning," *IEEE Open Journal of Control Systems*, vol. 3, pp. 266–281, 2024.
- [79] D. Kasenberg and M. Scheutz, "Interpretable apprenticeship learning with temporal logic specifications," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 4914–4921.
- [80] M. Gaon and R. Brafman, "Reinforcement learning

- with non-markovian rewards,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, pp. 3980–3987, Apr. 2020.
- [81] K. Jothimurugan, S. Bansal, O. Bastani, and R. Alur, “Compositional reinforcement learning from logical specifications,” in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 10 026–10 039.
- [82] A. Duret-Lutz, E. Renault, M. Colange, F. Renkin, A. Gbaguidi Aisse, P. Schlehuber-Caissier, T. Medioni, A. Martin, J. Dubois, C. Gillard *et al.*, “From spot 2.0 to spot 2.10: What’s new?” in *International Conference on Computer Aided Verification*. Springer, 2022, pp. 174–187.
- [83] R. Diestel, *Graph theory*. Springer Nature, 2025.
- [84] W. Rudin, “Principles of mathematical analysis,” *3rd ed.*, 1976.
- [85] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, “High-dimensional continuous control using generalized advantage estimation,” *arXiv preprint arXiv:1506.02438*, 2015.