Multi-Objective Policy Gradients with Topological Constraints

Kyle Hollins Wray,*[1] Stas Tiomkin,*[2] Mykel J. Kochenderfer,[1] and Pieter Abbeel[3]

Abstract— **Multi-objective optimization models that encode ordered sequential constraints provide a solution to model various challenging problems including encoding preferences, modeling a curriculum, and enforcing measures of safety. A recently developed theory of topological Markov decision processes (TMDPs) captures this range of problems for the case of discrete states and actions. In this work, we extend TMDPs towards continuous spaces and unknown transition dynamics by formulating, proving, and implementing the policy gradient theorem for TMDPs. This theoretical result enables the creation of TMDP learning algorithms that use function approximators, and can generalize existing deep reinforcement learning (DRL) approaches. Specifically, we present a new algorithm for a policy gradient in TMDPs by a simple extension of the proximal policy optimization (PPO) algorithm. We demonstrate this on a real-world multiple-objective navigation problem with an arbitrary ordering of objectives both in simulation and on a real robot.**

I. Introduction

In recent years, the theoretical foundations of Markov decision processes (MDPs) [1] and reinforcement learning (RL) [2] algorithms have grown to practical robotic applications in domains ranging from autonomous helicopter flight [3] to autonomous vehicles [4]. However, larger real-world domains often must consider multiple objectives such as in energy, comfort, and noise management in buildings [5] and hybrid electric vehicles [6]. In reinforcement learning, approaches like curriculum learning [7] and incorporating safety considerations [8] often sequentially learn differing objectives to incrementally develop skills. The topological MDP (TMDP) [9], [10] is a general model that captures this space of problems. There are various methods for multi-objective learning strategies such as constrained policy optimization [11] for non-incremental constrained MDPs (CMDPs) for related problems like robot walking [12]. However, there exists a gap in the theoretical foundation for solving these general multi-objective models with policy gradient-based algorithms, consequently limiting the principled development of deep RL solutions.

In a TMDP, multiple objectives are ordered in a directed acyclic graph (DAG) [9]. The ordering can capture preference, constraint, curriculum, or safety. A slack term, or equivalently budget, is assigned to each edge in the DAG. It represents the allowable deviation in value of an optimal policy in a parent objective to improve the value of a child objective. This flexible structure generalizes the constrained MDP (CMDP) [13], which is a fan-structured DAG, and a lexicographic MDP (LMDP) [10], which is a chain-structured DAG (Figure 1). The agent can learn all the objectives simultaneously [13], or it can incrementally step down the DAG in the order implied by its edges and learn each parent objective before moving on to its children [10].

Consider a robot navigation domain where the robot can be tasked with navigating to a goal location, monitoring a particular room, and avoiding another room, all based on a customer's preferences. Each customer may have a preference, such as for example prioritizing monitoring over navigation or navigation over avoiding a region. In each case, they may also have a tolerance allowing one objective to be reduced in favor of improving another. These preference structures and tolerances can be captured within a TMDP and provided to the agent as it learns to complete its objectives.

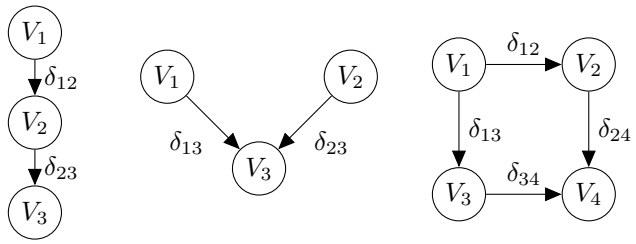
In practice, solving these kinds of multi-objective problems optimally is not feasible because all of the objectives depend on the current policy [14], [10], [11]. Chain-structured approaches that modify the value have been considered in the tabular case [14].

Local action restriction (LAR) [10], [15] is a scalable approximate method to solve TMDPs. It moves the global constraint into a set of local constraints over the states. This reformulation removes the dependence of the policy on the constraints' values. Fan-structured approaches (Figure 1 (b)) have considered policy gradients, such as in CPO [11]. However, the method only works for CMDPs and cannot model a DAG's topologically ordered constraints as in TMDPs. Also, it requires computing expensive second-order terms as a Hessian to approximate constraint satisfaction (see Section 6.1 [11]). In general, these approaches are computationally expensive and/or do not admit a direct policy gradient.

The goal of this work is to provide a multiple objective generalization of the policy gradient theorem for any given DAG of constraints and slack values. Our theorem derives a policy gradient that performs an accurate optimization of the objectives, each being subject to the constraints of their ancestors. We apply this general result in the specific case of extending proximal policy optimization (PPO) [16] for use in TMDPs. The resulting new algorithm is called topological policy optimization (TPO), integrating the new policy gradient into the PPO objective.

Our main contributions include: (1) a formal definition and derivation a multi-objective policy gradient within a TMDP (Section III); (2) the deep reinforcement learning algorithm called TPO using this new policy gradient (Section IV); (3) experiments demonstrating the success of TPO in both simulation and on a real robot (Section V).

* Equal contribution.
[1] Stanford University, Stanford, CA, USA.
[2] San Jose State University, CA, USA.
[3] University of California, Berkeley, CA, USA.
kylewray@stanford.edu, stas.tiomkin@sjsu.edu, mykel@stanford.edu, and pabbeel@cs.berkeley.edu



(a) Chain (b) Fan (c) DAG

Fig. 1. Example TMDP objective V_i nodes and edges with slacks δ_{ij}.

II. Background

A. Markov Decision Process (MDP)

A **Markov decision process (MDP)** is defined by the tuple $\langle S, A, T, R\rangle$. S is a set of states. A is a set of actions. $T(s' \mid s, a)$ is a Markovian state transition that captures the probability of transitioning to a successor state s' given that action a was taken in state s. $R(s,a)$ is a reward function that describes the effect of performing an action a in a state s. For discounted MDPs, $\gamma \in [0,1)$ is a discount factor on the reward over time. It is also common to have an initial s^0.

Reinforcement learning can be described as the collection of algorithms that do not assume T is provided [2]. Moreover, many of these algorithms do not assume S, A, or R are fully provided a priori either. Instead, they are only observed upon visiting a state and performing an action.

A **policy** $\pi(a \mid s)$ maps each state s to a probability of performing each action a. Let $\tau = \langle s^0, a^0, r^0, s^1, a^1, r^1, \ldots\rangle$ denote a **trajectory** with each state $a^t \sim \pi(\cdot \mid s^t)$, $r^t = R(s^t, a^t)$, and $s^{t+1} \sim T(\cdot \mid s^t, a^t)$. The goal in reinforcement learning is to explore and use experienced trajectories τ to find a policy π^* that maximizes expected reward.

Thus, the agent seeks an optimal policy π^* such that:

$$\pi^* = \operatorname*{argmax}_{\pi} \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t R(s^t, a^t) \mid \pi, s^0\Big].$$

Let the **value** $V^\pi : S \to \mathbb{R}$ of a policy π be its expected reward. Given the model, we can solve the MDP by iteratively applying the Bellman optimality equation at states s:

$$V^\pi(s) = \max_a Q^\pi(s,a)$$

$$Q^\pi(s,a) = R(s,a) + \gamma \sum_{s'} T(s' \mid s, a) V^\pi(s')$$

with V^* and Q^* denoting optimal values. The **advantage** is defined as $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$ [16] and is useful in describing the advantage in value of one action over another.

B. Policy Gradient

Policy gradient methods improve the policy along a gradient and forms the foundation for function approximation [17]. It assumes the policy π is parameterized by parameters θ. Let $\rho(\pi) = V^\pi(s^0)$ be the value of the initial state s^0 following a policy π. The policy gradient is:

$$\frac{\partial \rho^\pi}{\partial \theta} = \sum_s d^\pi(s) \sum_a \frac{\partial \pi(a \mid s)}{\partial \theta} Q^\pi(s,a)$$

with d^π denoting the stationary state distribution of π.

C. Topological Markov Decision Process

A **topological Markov decision process (TMDP)** [9] generalizes the MDP to multiple objectives and is defined by $\langle S, A, T, \mathbf{R}, E, \delta\rangle$. $\mathbf{R} : S \times A \to \mathbb{R}^k$ emits k rewards; each $i \in K = \{1, \ldots, k\}$ can be written as $R_i(s,a)$. $E \subseteq K \times K$ forms a directed acyclic graph (DAG) over the rewards, with one leaf node, assumed to be k without loss of generality. $\delta = \{\delta_{wv} \mid (w,v) \in E\}$ denotes the set of slack variables—allowable deviation from optimal value—for each parent-child objective pair. The objective in a TMDP is: for each objective $i \in K$, following the order of the DAG E, we solve:

$$\begin{aligned} \underset{\pi}{\text{maximize}} \quad & V_i^\pi(s^0) \\ \text{subject to} \quad & V_w^*(s^0) - V_w^\pi(s^0) \le \delta_{wv}, \quad \forall v \in \mathcal{A}_i \cup \{i\}, \forall w \in \mathcal{P}_v \end{aligned} \tag{1}$$

with $\mathcal{P}_v, \mathcal{A}_i \subset K$ denoting the parents and ancestors of v and i in E respectively; and $V_j^*(s^0)$ denoting the optimal value of j following this same constrained objective. In other words, the constraints state that every vertex v must satisfy the slacks η_{wv} from its parents w following an edge $e = (w,v)$. Also, let $E_i = \{e = \langle w, v\rangle \in E \mid v \in \mathcal{A}_i \cup \{i\} \text{ and } w \in \mathcal{P}_v\}$ denote all i's ancestral edges in E. An optimal policy π^* is the policy π_k^* computed by the leaf node k in E. The leaf optimizes $V_k^*(s^0)$ ensuring that the union of all ancestral constraints are satisfied.

Local action restriction (LAR) [10], [9] is an approximation that restriction actions locally by a local slack η, rather than a global slack δ. It has been shown that if $\eta_i = (1-\gamma)\delta_i$, then global slack can be preserved, if desired. LAR enables scalable algorithms to be devised. The resulting TMDP objective for each objective i becomes:

$$\begin{aligned} \underset{\pi}{\text{maximize}} \quad & V_i^\pi(s^0) \\ \text{subject to} \quad & V_w^*(s) - Q_w^*(s, \pi(s)) \le \eta_{wv}, \\ & \qquad \forall v \in \mathcal{A}_i \cup \{i\}, \forall w \in \mathcal{P}_v, s \in S \end{aligned} \tag{2}$$

TMDPs generalize both LMDPs and feasible CMDPs. See Figure 1 for examples.

III. Policy Gradient Theorem for TMDP

Solving the LAR optimization in Equation 2 can be done using a modified Bellman optimality equation for tabular policy [10], [15], [9]. This formulation does not admit function approximation, or specifically, the use of a policy gradient or deep reinforcement learning methods. Our main result in this section is a formulation and proof of a policy gradient theorem to solve Equation 2.

The goal is to prove a TMDP policy gradient theorem. To accomplish this, we need to formulate a constrained Bellman optimality equation corresponding to Equation 2. LAR allows us to move the constraints into the Bellman optimality equation [9]. However, this becomes a constrained optimization problem. We seek to reduce this to an unconstrained optimization problem in order to leverage the abundant prior work on (approximate) policy gradients [18], [16]. We convert the constrained Bellman equation into an unconstrained Bellman equation using Lagrange multipliers.

However, the standard approach results in the reward itself being penalized by an arbitrary amount based on the ancestors' advantages. Consequently, this would change the optimization's value function.

We provide a solution that will not affect the reward in this manner. This requires both: (1) the Lagrange multiplier to be assigned properly via a bound, and (2) the constraint term to be transformed such that it is zero when the constraint is not violated. We derive a bound on the Lagrange multiplier to ensure it is large enough such that any action that violates the constraint will not be chosen. We prove that the transformed constraint term preserves the original constraint satisfaction. When these facts are combined, the result is that the original value is preserved and the constraints are satisfied.

With this new Bellman optimality equation, Lagrange multiplier bounds, and transformed constraint, we prove the constrained policy gradient theorem for the TMDP.

A. Lagrangian Bellman Optimality Equation

We need to derive the corresponding Bellman optimality equation. However directly writing a Bellman equation for the unconstrained optimization formulation in Equation 1 can be onerous due to the extra terms. Instead, as shown in prior work [10], [9], we can leverage LAR to move the constraints in Equation 2 into the Bellman optimization problem over actions. Ensuring the original constraints are satisfied at each state, implies they are satisfied for the original optimization. This Bellman optimality equation with LAR at a state s is the constrained optimization [9]:

$$\begin{aligned} \underset{a}{\text{maximize}} \quad & Q_i^\pi(s,a) = R_i(s,a) + \gamma \sum_{s'} T(s' \mid s,a) V_i^\pi(s') \\ \text{subject to} \quad & -A_w^*(s,a) \leq \eta_{wv}, \forall w, v \end{aligned} \tag{3}$$

with $V_i^\pi(s) = Q_i^\pi(s,a^*)$ for constraint-optimal action a^*.

The naive use of Lagrange multipliers would result in the undesirable Equation 4 below that arbitrarily modifies the rewards, affecting the values as a consequence:

$$\begin{aligned} \mathbf{V}_i^\pi(s) = \max_a \Big(& R_i(s,a) + \gamma \sum_{s'} T(s' \mid s,a) \mathbf{V}_i^\pi(s') \\ & - \sum_{v,w} \beta_{wvs} (-A_w^*(s,a) - \eta_{wv}) \Big) \end{aligned} \tag{4}$$

with Lagrange multipliers $\beta_{wvs} \geq 0$ and Lagrangian value function $\mathbf{V}_i^\pi$. To illustrate the issue, consider two cases. If objective i chooses an action beyond the budgeted slack η_{wv}, then there should be a penalty. Otherwise, there should not be an increase in the reward for an action less than the budgeted slack. Instead, the agent should simply maximize its original reward. Thus, crucially, the extra constraint terms should only be a penalty if the constraint is violated. Otherwise, the reward should remain unaffected. To accomplish this, we need to transform the constraint as follows.

Proposition 1: For an objective i and state s, the optimization in Equation 3 is equivalently solved by:

$$\begin{aligned} \underset{a}{\text{maximize}} \quad & Q_i^\pi(s,a) = R_i(s,a) + \gamma \sum_{s'} T(s' \mid s,a) V_i^\pi(s') \\ \text{subject to} \quad & C_{wv}(s,a) \leq 0, \forall w, v \end{aligned} \tag{5}$$

with $C_{wv}(s,a) = \max\{0, -A_w^*(s,a) - \eta_{wv}\}$.

Proof: The TMDP Bellman optimality equation with LAR at a state s follows Equation 3. By definition, for the optimal policy's values, $-A_w^*(s,a) = V_w^*(s) - Q_w^*(s,a) \geq 0$ and there always exists an action a^* such that $-A_w^*(s,a) = 0$. Thus this optimization always feasible and we have:

$$\begin{aligned} -A_w^*(s,a) \leq \eta_{wv} &\Rightarrow -A_w^*(s,a) - \eta_{wv} \leq 0 \\ &\Rightarrow \max\{0, -A_w^*(s,a) - \eta_{wv}\} \leq 0 \\ &\Rightarrow C_{wv}(s,a) \leq 0 \end{aligned}$$

This lets us rewrite the optimization problem as in Equation 5, yielding our result. ∎

Now we can compute the Lagrangian of the constrained optimization problem in Equation 5. The **Lagrangian Bellman optimality equation** is:

$$\begin{aligned} \mathbf{V}_i^\pi(s) = \max_a \Big(& R_i(s,a) + \gamma \sum_{s'} T(s' \mid s,a) \mathbf{V}_i^\pi(s') \\ & - \sum_{v,w} \beta_{wvs} C_{wv}(s,a) \Big) \end{aligned} \tag{6}$$

For notational convenience, for any objective $j \in K$, the **Lagrangian Q-value** is $\mathbf{Q}_j(s,a)$ and the **Lagrangian advantage** is: $\mathbf{A}_j(s,\pi(s)) = \mathbf{Q}_j(s,\pi(s)) - \mathbf{V}_j(s)$. In practice, $\mathbf{A}_j$ can be computed using a new form of generalized advantage, discussed in the next section.

Equation 6 has two main properties. First, it converted the original constrained optimization problem into an unconstrained Bellman optimality equation. Second, if its constraints are satisfied, then it preserves the original reward; otherwise it penalizes the reward.

The scale of the constraint terms $C_{wv}(s,a)$ can be arbitrary based on the scale of the values. If we do not select a sufficiently large multiplier, then it is possible that the penalty will not be enough to ensure constraint-violating actions are never chosen.

Thus, we derive a lower bound on β_{wvs} such that the application of the Lagrangian Bellman optimality equation (Equation 6) will never select constraint-violating actions. This is derived in Proposition 2.

Finally, we must also prove that it preserves optimality of the original problem. In summary, the use of Equation 6 with β_{wvs} as described below will converge to the correct values and simultaneously enforce the LAR constraints. This is proven in Theorem 1.

Proposition 2: Given objective i, state s, ancestor edges $\langle w, v \rangle \in E_i$, values Q_i, at least one constraint-violating action (infeasibility) exists, and optimal constraint-satisfying action $\hat{a}^*$, if all β_{wvs} satisfy:

$$\beta_{wvs} \geq \max_a \begin{cases} \frac{Q_i(s,a) - Q_i(s,\hat{a}^*)}{\sum_{v,w} C_{wv}(s,a)} & \text{if } \sum_{v,w} C_{wv}(s,a) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

then no constraint-violating actions $\hat{a}$ will be chosen, i.e., $\max_a \mathbf{Q}_i^\pi(s,a) = \mathbf{Q}_i^\pi(s,\hat{a}^*)$.

Proof: At any iteration of the Bellman equation, for any states s', assume no other states have chosen suboptimal action. By Proposition 1 and Equation 6, the Lagrangian values are identical to the constrained optimization problems' values $\mathbf{V}_i^\pi(s') = V_i^\pi(s')$.

Below, let $Q_i(s,a)$ be the result of applying the standard Bellman equation. Let $\hat{a}^*$ be the optimal constraint-satisfying action solving Equation 3. Note that both $Q_i(s,a)$ and $\hat{a}^*$ can be computed separately without depending on the equation being considered (Equation 6).

Consider the application of the Lagrangian Bellman optimality equation on a state s (Equation 6). Assume the existence of a constraint violating action $\hat{a}$. (By construction in Proposition 1, any constraint-satisfying action will result in the penalty constraint terms being 0.) For this maximization, it is sufficient to compare the actions $\hat{a}^*$ and $\hat{a}$:

$$\mathbf{Q}_i^\pi(s,\hat{a}^*) \geq \mathbf{Q}_i^\pi(s,\hat{a})$$

$$R_i(s,\hat{a}^*) + \gamma \sum_{s'} T(s' \,|\, s,\hat{a}^*) \mathbf{V}_i^\pi(s') - \sum_{v,w} \beta_{wvs} C_{wv}(s,\hat{a}^*)$$
$$\geq R_i(s,\hat{a}) + \gamma \sum_{s'} T(s' \,|\, s,\hat{a}) \mathbf{V}_i^\pi(s') - \sum_{v,w} \beta_{wvs} C_{wv}(s,\hat{a})$$

Since $\mathbf{V}_i^\pi(s') = V_i^\pi(s')$ and $C_{wv}(s,\hat{a}^*) = 0$, we have:

$$R_i(s,\hat{a}^*) + \gamma \sum_{s'} T(s' \,|\, s,\hat{a}^*) V_i^\pi(s') - \sum_{v,w} \beta_{wvs} 0$$
$$\geq R_i(s,\hat{a}) + \gamma \sum_{s'} T(s' \,|\, s,\hat{a}) V_i^\pi(s') - \sum_{v,w} \beta_{wvs} C_{wv}(s,\hat{a})$$
$$Q_i(s,\hat{a}^*) - Q_i(s,\hat{a}) + \sum_{v,w} \beta_{wvs} C_{wv}(s,\hat{a}) \geq 0$$

Let β_s be the same constant used for all β_{wvs}, yielding:

$$Q_i(s,\hat{a}^*) - Q_i(s,\hat{a}) + \beta_s \sum_{v,w} C_{wv}(s,\hat{a}) \geq 0$$
$$\beta_s \geq \frac{Q_i(s,\hat{a}) - Q_i(s,\hat{a}^*)}{\sum_{v,w} C_{wv}(s,\hat{a})}$$

This inequality must be satisfied for all $\hat{a}$, enforcable with a maximization over actions. This results in Equation 7. ∎

Theorem 1: Lagrangian Bellman optimality Equation 6 solves the optimization in Equation 2.

Proof: By Proposition 2, there exists a β_{wvs} such that the use of the Lagrangian Bellman optimality equation (Equation 6) ensures the constraints: $C_{wv}(s,a) \leq 0, \forall w, v$, are always satsified.

Since the constraints are satisfied, by construction Equation 6 has constraint terms 0, implying $\mathbf{Q}_i^*(s,\hat{a}^*) = Q_i^*(s,\hat{a}^*) - 0 = Q_i^*(s,\hat{a}^*)$. Thus it solves the optimization in Equation 5.

By Proposition 1, it also solves the optimization problem in Equation 3. By construction of Equation 3's constraints, for all states s, $-A_w^*(s,a) \leq \eta_{wv}$, and again $\mathbf{V}_i^*(s,a) = V_i^*(s,a)$. Thus, all constraints of the original optimization in Equation 2 are satified and return the same resulting policy at all states, solving this original optimization. ∎

Algorithm 1 Topological policy optimization

Require: k: Number of objectives
Require: E: DAG of objective relationships
Require: δ: Slacks for each objectives
Require: θ^0: Initial policy parameters

$\pi_\theta, \hat{\mathbf{V}} \leftarrow \pi_{\theta^0}, \emptyset$
for $i \leftarrow$ TOPOLOGICALSORT(E) **do**
  // $\hat{\mathbf{A}}_i$ is a function of trajectories computed during
  // PPO's rollouts; it uses the learned ancestral
  // critics' set $\hat{\mathbf{V}}$ as a parameter
  $\hat{\mathbf{A}}_i \leftarrow$ Eq. 10 using this iteration's $\hat{\mathbf{V}}$ in C_{wv}
  $\pi_\theta, \hat{V}_i \leftarrow \text{PPO}(\pi_\theta, \hat{\mathbf{A}}_i)$ *// Use $\hat{\mathbf{A}}_i$ for PPO's adv.*
  $\hat{\mathbf{V}} \leftarrow \hat{\mathbf{V}} \cup \{\hat{V}_i\}$ *// Extend critic set for descendants*
return π_θ, $\hat{\mathbf{V}}$

B. Multi-Objective Lagrangian Policy Gradient Theorem

We have changed the objective into a form involving additional ancestor objectives. Consequently, it is not a given that parameterized policies learning from samples—such as in the case of deep reinforcement learning—is able to converge to the optimal policy. Here we present a novel policy gradient theorem [17] for any algorithm using this Lagrangian TMDP objective. This *multi-objective Lagrangian policy gradient theorem* is presented in Proposition 3 below.

Proposition 3: For any TMDP, with an discounted infinite horizon objective (or an average-reward objective), for each objective i:

$$\frac{\partial \rho_i^\pi}{\partial \theta} = \sum_s d^\pi(s) \sum_a \frac{\partial \pi(a \,|\, s)}{\partial \theta} \mathbf{Q}_i^\pi(s,a). \tag{8}$$

Proof: We begin by rewriting Equation 6:

$$\mathbf{V}_i^\pi(s) = \max_a \Big(\mathbf{R}_i(s,a) + \gamma \sum_{s'} T(s' \,|\, s,a) \mathbf{V}_i^\pi(s') \Big)$$

with $\mathbf{R}_i(s,a) = R_i(s,a) - \sum_{v,w} \beta_{wvs} C_{wv}(s,a)$. Consider the partial derivative of $\mathbf{R}_i$ with respect to policy parameters θ. First, $\frac{\partial}{\partial \theta} R_i(s,a) = 0$. Second, by definition in Equation 5, each C_{wv} does not depend on the current policy parameters θ, as it is proportional to the optimal advantages and slacks of ancestors: $C_{wv}(s,a) = \max\{0, -A_w^*(s,a) - \eta_{wv}\}$. We have $\frac{\partial}{\partial \theta} C_{wv}(s,a) = 0$. Thus, $\frac{\partial}{\partial \theta} \mathbf{R}_i(s,a) = 0$.

The rest of this proof follows directly from the original policy gradient theorem's proof [17], using $\mathbf{R}_i$ as the reward and the fact that $\frac{\partial}{\partial \theta} \mathbf{R}_i(s,a) = 0$. ∎

IV. Topological Policy Optimization

Proximal policy optimization (PPO) [16] uses a (clipped) surrogate objective, computing an approximate policy gradient over batch trajectories τ of length h, and using a generalized advantage estimation. We apply our Lagrangian Bellman equation results from the previous section to PPO. The following equations refer to this novel contribution. At time step t, the objective i's **Lagrangian surrogate objective** is:

$$\mathbf{L}_i(\theta) = \hat{\mathbb{E}}^t \Big[\log \pi_\theta(a^t | s^t) \hat{\mathbf{A}}_i^t \Big] \tag{9}$$

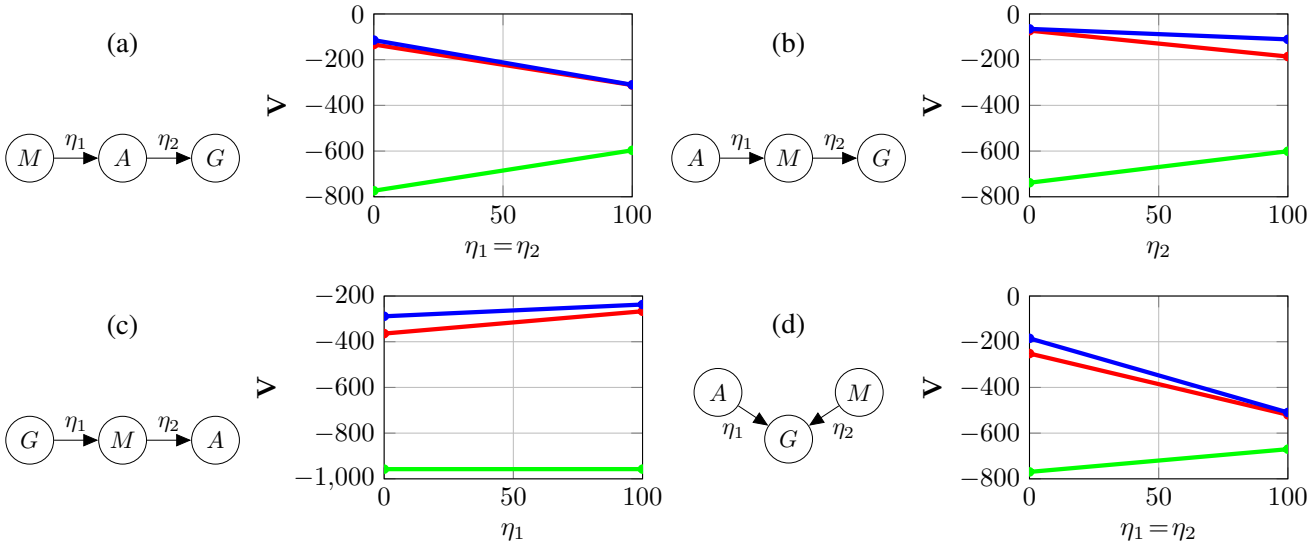


Fig. 2. Results for four different DAGs and slack assignments on the robot navigation domain. The x-axis denotes slack η. The y-axis denotes values for all three objectives: V_{avoid} (red), V_{monitor} (blue), and V_{goal} (green). Note that the units of the values for the three objectives are distinct from one another.

with $\hat{\mathbf{A}}_i^t$ denoting advantages computed by **generalized Lagrangian advantage estimation** (Equation 10 below).

The generalized Lagrangian advantage estimation defines $\hat{\mathbf{A}}_i^t = \delta_i^t + (\gamma\lambda)\delta_i^{t+1} + \cdots + (\gamma\lambda)^{h-t+1}\delta_i^{h-1}$ with single advantage estimates $\delta_i^k = \mathbf{r}_i^k + \gamma\mathbf{V}_i^\pi(s^{k+1}) - \mathbf{V}_i^\pi(s^k)$. The sample-based estimate for this expression is computed by:

$$
\begin{aligned}
\hat{\mathbf{A}}_i^t &= \sum_{k=t}^{h-1}(\gamma\lambda)^k\delta_i^k = \sum_{k=t}^{h-1}(\gamma\lambda)^k\Big(\mathbf{r}_i^k + \gamma\mathbf{V}_i^\pi(s^{k+1}) - \mathbf{V}_i^\pi(s^k)\Big) \\
&= \sum_{k=t}^{h-1}(\gamma\lambda)^k\Big(r_i^k - \sum_{v,w}\beta_{wvs^k}C_{wv}(s^k,a^k) \\
&\qquad + \gamma(V_i^\pi(s^{k+1}) - \sum_{v,w}\beta_{wvs^{k+1}}C_{wv}(s^{k+1},a^{k+1})) \\
&\qquad - (V_i^\pi(s^k) - \sum_{v,w}\beta_{wvs^k}C_{wv}(s^k,a^k))\Big) \\
&= \sum_{k=t}^{h-1}(\gamma\lambda)^k\Big(r_i^k + \gamma V_i^\pi(s^{k+1}) - V_i^\pi(s^k) \\
&\qquad - \gamma\sum_{v,w}\beta_{wvs^{k+1}}C_{wv}(s^{k+1},a^{k+1})\Big) \qquad (10)
\end{aligned}
$$

with C_{wv} incorporating the ancestral advantages A_{wv} as defined by Equation 5.

Additional components of PPO and similar algorithms may be used, which are applied inside of the PPO routine in line 6, Algorithm 1. Specifically, we employ PPO's policy gradient entropy term and its clipped ratio for the policy.

V. Experiments

In this section, we consider experiments in the *multi-objective robot navigation* [19] domain, which is fully implemented on a real robot acting in an actual household environment. Importantly, the main contribution of this paper remains the multi-objective policy gradient's theoretical results and formulation, rather than the implementation of the approach in TPO and a real world navigation domain. These results are included to provide evidence of the approach's usefulness and build intuitions in modeling objective structures and behavioral customizability using slack.

A. Multi-Objective Robot Navigation Domain

We consider navigation domains in which a robot is provided with the tuple $\langle \mathcal{M}, \mathcal{L}^0, \mathcal{L}^g, \mathcal{R}^a, \mathcal{R}^m \rangle$. $\mathcal{M} \in \{0,1\}^{m\times n}$ is a map, described here as a occupancy grid (or binary matrix), with a starting location $\mathcal{L}^0 \in \mathcal{M}$ and a goal location $\mathcal{L}^g \in \mathcal{M}$. All this information and the state transitions, which describe the robot's movement dynamics and its interaction with walls or obstacles in the map, are unknown a priori.

The *goal* objective is to navigate from $\mathcal{L}^0$ to $\mathcal{L}^g$ as fast as possible. The robot receives a -1 for any non-goal state, including when interacting with walls or obstacles. Once the robot reaches the goal $\mathcal{L}^g$, its navigation ends.

Two additional objectives are also provided. The *avoid* objective is to penalize entering a rectangular region $\mathcal{R}^a$. The robot receives a -1 for each time step in this region. The *monitor* objective is to encourage entering into a rectangular region $\mathcal{R}^m$. The robot receives a $+1$ for each time step in this region. For both objectives, any interaction with walls or obstacles is still met with a penalty of -1.

This domain builds on prior work on multi-objective (PO)MDPs for home healthcare robots [19]. In particular, the navigation of a home healthcare robot can be tailored to different homes via the map $\mathcal{M}$; delivering medicine through the assignment of $\mathcal{L}^0$ and $\mathcal{L}^g$; the preference of the human(s) or patients to avoid traversing rooms via $\mathcal{R}^a$; the preference to monitor rooms while navigating via $\mathcal{R}^m$; and so on.

B. Experimental Setting

The thesis of this paper is to propose a novel multi-objective policy gradient formulation for the new TMDP model, for which no algorithm currently exists for continuous state spaces. Therefore, the theoretical result and formulation

Fig. 3. Experiments that implement *home navigation* on a real robot in an actual household environment. This uses the $M \rightarrow A \rightarrow G$ DAG with two paths: (1) constrained-to-monitor (blue path) with $\eta_1 = \eta_2 = 0$, and (2) constrained-to-avoid (red path) with $\eta_1 = 100$ and $\eta_2 = 0$. The start point (S), *goal* point (G), *avoid* region, and *monitor* region are shown.

must ideally be evaluated by comparing its efficacy at modeling different permutations of multiple objective DAGs and slack assignments. Thus, the primary metrics must be the values of all three objectives: V_{avoid}, V_{monitor}, and V_{goal}. The results indicate how effective the approach is at capturing the slack and the preferences encoded by each of the DAGs.

The implementation of this theoretical approach is the approximate TPO algorithm. For each of the configurations of DAG and slacks, the algorithm is trained for 300000 iterations. The experiments compute the value for each objective via Monte Carlo simulations using the agent's final policy network. Analysing convergence is left to future work. The approach is implemented in Julia 1.6.1 on Ubuntu 18.04.

C. Results and Discussion

Figure 2 provides results of four distinct constraint DAGs, with each varying the slack values. This figure illustrates the effect of the topological structure of the constraint DAG and the slack assignments on the three objectives' values.

In Figures 2 (a) and (b), we see the desired effect: increasing the slack of the constraining objectives *avoid* (A) and *monitor* (M) reduces their value and enables the *goal* (G) objective to improve its value. Figure 2 (d) shows a similar effect, except with a different fan DAG structure. In this case, the result DAG structure enables a more drastic change. In Figure 2 (c), the *goal* is the first objective. We observe now that by increasing the slack of the G, the other two objectives (A and M) are able to increase their values.

Figure 3 demonstrates the implementation of the approach on an actual robot acting in a real household environment. With zero slack, the primary *monitor* objective is favored, forcing the *avoid* objective to experience a penalty. However when provided slack, the *avoid* objective is able to direct the path away from the region while navigating to the goal.

VI. Conclusion

This paper presents a policy gradient approach for capturing rich multi-objective preference structures in reinforcement learning. While primarily a theoretical paper, the method is demonstrated in a real robot navigation domain. With this paper's established theoretical foundation, future work will introduce applications of this approach for a range of reinforcement learning problems including formulations of curriculum learning and imitation learning.

References

[1] R. E. Bellman, *Dynamic Programming*. Princeton University Press, 1957.

[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[3] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *International Conference on Machine Learning (ICML)*, 2004.

[4] K. H. Wray, S. J. Witwicki, and S. Zilberstein, "Online decision-making for scalable autonomous systems," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2017, pp. 4768–4774.

[5] L. Klein, J. young Kwak, G. Kavulya, F. Jazizadeh, B. Becerik-Gerber, P. Varakantham, and M. Tambe, "Coordinating occupant behavior for building energy and comfort management using multi-agent systems," *Automation in Construction*, vol. 22, pp. 525–536, 2012.

[6] K. H. Wray, R. Lui, and L. Pedersen, "Engine activation planning for series hybrid vehicles," in *IEEE Intelligent Vehicles Symposium*, 2021, pp. 238–244.

[7] C. Florensa, D. Held, M. Wulfmeier, M. Zhang, and P. Abbeel, "Reverse curriculum generation for reinforcement learning," 2017, pp. 482–495.

[8] A. Wachi and Y. Sui, "Safe reinforcement learning in constrained Markov decision processes," in *International Conference on Machine Learning (ICML)*, 2020, pp. 9797–9806.

[9] K. H. Wray, "Abstractions in reasoning for long-term autonomy," Ph.D. dissertation, University of Massachusetts, Amherst, MA, 2019.

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

[11] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," 2017.

[12] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine, "Learning to walk via deep reinforcement learning," *arXiv preprint arXiv:1812.11103*, 2018.

[13] E. Altman, *Constrained Markov decision processes*. Chapman & Hall/CRC Press, 1999.

[14] Z. Gábor and Z. K. C. Szepesvári, "Multi-criteria reinforcement learning," in *International Conference on Machine Learning (ICML)*, 1998, pp. 197–205.

[15] K. H. Wray and S. Zilberstein, "Multi-objective POMDPs with lexicographic reward preferences," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2015, pp. 1719–1725.

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[17] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in Neural Information Processing Systems (NIPS)*, 2000, pp. 1057–1063.

[18] J. Schulman, S. Levine, P. Moritz, M. Jordan, and P. Abbeel, "Trust region policy optimization," in *International Conference on Machine Learning (ICML)*, 2015, pp. 1889–1897.

[19] K. H. Wray and K. Czuprynski, "Scalable gradient ascent for controllers in constrained POMDPs," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.