Abstract—Performance and safety are often two competing objectives in decision-making problems. We study the problem of integrating a collection of controllers with different safety and performance levels into one that takes a middle-ground position amongst them. In the first contribution, we formulate the problem of blending controllers using the framework of constrained Markov decision processes and contextual multi-objective bandits. We use the reward function and the auxiliary costs of the Markov decision process to measure the performance and the safety of a controller, respectively. We subsequently use these measures to form the feedback of a bandit whose arms are the input controllers. The blending algorithm must interact with the bandit and minimize a regret term that measures the suboptimality of the pulled arms with respect to an expert whose choice of arms is Pareto optimal. In the second contribution, we design a blending algorithm and show that its average regret converges to zero. We also derive an upper bound on the algorithm’s suboptimality in performance and safety and we show that its computation imposes no additional computational complexity. We empirically demonstrate the algorithm’s success in blending a safe and a performant controller in a variety of Safety Gym environments. The results reflect the following key takeaway: the blended controller shows a strict improvement in performance compared to the safe controller and is safer than the performant controller.

I. INTRODUCTION

Designing autonomous systems that are both safe and well-performing is challenging. For example, reinforcement-learning algorithms perform their given tasks efficiently; however, their choice of actions often raises safety concerns [1], [2]. On the other hand, techniques from control theory or formal methods, which address the safety concerns, require restrictive assumptions on the environment dynamics and are often conservative in performance [3], [4], [5].

We study a pragmatic approach of blending controllers with different safety and performance profiles into one that takes a middle-ground position. Consider two given controllers: a performant controller that accumulates higher rewards than the other but may frequently take unsafe actions, and a safe controller that takes unsafe actions less frequently than the former but may accumulate lower rewards as well. The goal of “blending controllers” in this example is to find a switching strategy between the two controllers such that the resulting controller is safer than the performant controller and accumulates a higher total reward than the safe controller. Finding a proper switching strategy is challenging because, for example, it may take the agent to a state that neither the safe controller renders as safe nor had it been experienced by the reinforcement-learning algorithm that may drive the performant controller.

We consider constrained Markov decision processes, wherein the reward function is augmented with auxiliary costs to measure the safety of an agent [6]. For example, the environment may issue a cost whenever the agent collides with an obstacle. Such a framework provides a measure to compare the safety of a collection of controllers [7], [8], which makes it amenable to the problem of blending controllers. Under this framework, a controller is safer than another if it achieves a lower expected total cost.

A. Contributions

In the first contribution, we formulate the problem of blending controllers using the framework of contextual multi-objective bandits. We require a blending algorithm to justify its choice of controllers according to a Pareto dominance relationship, i.e., we impose a regret on the agent whenever it chooses a controller with both inferior safety and performance measures. We formulate the regret based on the notion of Pareto regret [9], which is an aggregate suboptimality of the algorithm with respect to an expert who always picks the controller with its next-step reward and cost not Pareto dominated by any other controller. Thus, we desire an algorithm whose average Pareto regret converges to zero, i.e., one that achieves a sublinear Pareto regret.

We allow an arbitrary number of controllers and objectives to be blended together. Such an approach may simplify the overall safety modeling because it facilitates blending a performant controller with multiple safe controllers, where each controller specializes in a specific safety specification. For example, in autonomous driving vehicles, a myriad of safety specifications are taken into account such as centering the vehicle in the lane and maintaining a safe distance from other vehicles, and it would be simpler to consider each objective separately.

In the next contribution, we propose a multi-armed multi-objective contextual bandit algorithm for blending controllers. The set of the input controllers form the arm set of the bandit, the total reward and the negation of the costs collectively form the bandit feedback, and the context is generated by a fixed feature mapping that maps from the possibly high-dimensional state space to a lower-dimension feature vector. The proposed bandit algorithm is inspired
by [10]. However, we differentiate between the arms whose feedback is not Pareto dominated by any other arm. In particular, we pick the one that sacrifices the least amount in individual objectives.

We show that, with high probability, the proposed algorithm for blending controllers achieves a sublinear Pareto regret under a linear-feedback assumption, which we formally state and justify. We then establish a probabilistic bound on the maximal individual objective that the algorithm sacrifices during execution. The bound only requires the estimates that the algorithm already maintains from the next-step feedback vector; therefore, its computation imposes no additional computational complexity.

We use the Safety Gym benchmark suite [11] as the test bed for the numerical experiments. The Safety Gym benchmarks focus on the proximal policy-optimization (PPO) [12] and trust-region policy-optimization (TRPO) [13] algorithms alongside some safety-aware versions of them that take the auxiliary costs into account. The benchmarks suggest that these algorithms achieve a significantly lower cost than the regular PPO and TRPO algorithms. In return, their total reward is significantly lower as well.

To find a middle ground between the algorithms above, we test the proposed algorithm considering various scenarios that arise from different choices of the algorithms above. In all of the tested scenarios, a statistical analysis of the total reward and cost of the algorithm confirms that the blended controller shows a significant improvement in its total cost when compared to the performant controller and in its total reward when compared to the safe controller.

B. Related Work

We first review multi-objective reinforcement-learning algorithms. A popular approach is to use scalarization functions to order the preference between different objective outcomes [14], [15]. In this approach, the goal is to find the optimal policies that lie on the Pareto-front line. Linear scalarization functions have been shown to be ineffective when the Pareto-front line is not convex, for which the work in [16] proposes the Chebyshev scalarization as an alternative solution. However, there still remain concerns that such a method may not converge as it does not satisfy the Bellman equation of optimality [17]. We avoid scalarization methods to prevent the issues mentioned above. Moreover, the results in [9] suggest a superior performance of multi-objective bandit algorithms that use the Pareto regret compared to ones that use a regret based on the above scalarization methods.

Next, we review the related safety-aware reinforcement learning algorithms that use the framework of constrained Markov decision processes. In particular, we consider constrained policy optimization (CPO) [18], TRPO-Lagrangian and PPO-Lagrangian [11], which are present in the Safety Gym benchmarks. In these algorithms, although multiple safety objectives may be considered during learning, even a slight change to the safety specifications will require learning a new policy from scratch. In contrast, the modular approach of blending controllers allows the decision-maker to simply blend the outdated policy with a new policy that satisfies the new safety specification.

The algorithm suggested in [19] finds a switching strategy between policies that are easy to implement such as PID controllers, and therefore, is closely related to the problem of blending controllers. However, unlike the proposed algorithm in this work, it does not consider multiple objectives. In another related line of work, a decision logic between a safe and a performant controller is developed to control a plant based on either a Lyapunov function or a reachability analysis [20], [21], [22], [23], [24]. In contrast to this work, these approaches assume known environment dynamics.

Finally, we review shared control protocols, where a robot’s commands are blended with a human user. We share the same goal of balancing the safety and performance of the given controllers with these approaches. However, the algorithms in [25], [26], [27] use a weighted sum of the given controllers’ outputs, hence, they are restricted to continuous-domain controllers. In [28], the dynamics are unknown but assumed to be linear. Finally, the authors in [29] use semidefinite programming for blending controllers, which requires prior knowledge regarding the environment dynamics and a confidence level for the controllers. In contrast to the above two works, we take a model-free approach.

II. PRELIMINARIES AND PROBLEM STATEMENT

In this section, we first establish the notations used throughout the paper. We then review contextual multi-objective bandit algorithms and introduce the required preliminary definitions. Finally, we formally state the problem of blending controllers using the notion of Pareto regret.

A. Notation

We denote the set of real numbers by $\mathbb{R}$, non-negative reals by $\mathbb{R}_+$, and natural numbers by $\mathbb{N}$. For any $n \in \mathbb{N}$, $\{n\} := \{1, \ldots, n\}$. Let $v \in \mathbb{R}^n$ and $i \in \{n\}$, then $v^i$ is the transpose and $(v)_i$ is the $i$th component of $v$. For any $u, v \in \mathbb{R}^n$, the inner product of $u$ and $v$ is denoted by $u \cdot v$. Let $v \in \mathbb{R}^n$ and $W \in \mathbb{R}^{n \times n}$, then, $\|v\|_W$ is the matrix norm of $v$ with respect to $W$, i.e., $\|v\|_W := v^T W v$, and $\|v\|_2$ is the second norm of $v$.

Let $u, v \in \mathbb{R}^n$, then $u$ is said to Pareto dominate $v$, denoted $u \succ v$ if and only if, for all $i \in \{n\}$, we have that $u_i \geq v_i$ and there exists $j \in \{n\}$ such that $u_j$ is strictly greater than $v_j$. We use notation $u \not\succ v$ if $u$ is not Pareto dominated by $v$, i.e., $v \succ u$ or there exists $i, j \in \{n\}$ such that $i \neq j$, $v_i > u_i$, and $v_j < u_j$.

B. Contextual Multi-Objective Bandits

We adopt the same framework for bandit algorithms as in [30]. $X$ denotes the arm set of the bandit. We use notation $Z$ for the environment state space, and $T$ denotes the learning horizon. For every $t \in \{T\}$, a context $\Psi_t \in \mathbb{R}^d$ is provided to the agent. The context is generated using a fixed-in-time feature mapping $\psi: Z \times X \rightarrow \mathbb{R}^d$. Specifically, let $z_t \in Z$ be the current state of the environment and $x_t \in X$ be the arm picked by the algorithm. Then, $\Psi_t = \psi(z_t, x_t)$.
Upon pulling an arm, the environment issues the agent with a bandit feedback \( y_t \in \mathbb{R}^m \). The feedback consists of \( m \) objectives that the algorithm seeks to maximize. Without loss of generality, we assume that the objective values are normalized such that each unit measurement has equal importance amongst all objectives.

Next, we define Pareto suboptimality gap [10] and the maximal loss in individual objectives, which we later use to formally define the Pareto regret and cumulative maximal loss in individual objectives.

**Definition 1.** Let \( z_t \in \mathcal{Z} \) denote the state of the environment at stage \( t \) and \( x \in \mathcal{X} \). Define \( \mu_{t,x} := \mathbb{E}[y_t | x, z_t] \), the expected value of the feedback vector corresponding to arm \( x \) at state \( z_t \). Then, the Pareto suboptimality gap of \( x \) is

\[
\Delta_t(x) := \inf \{ \epsilon \in \mathbb{R}_+ | \mu_{t,x} + \epsilon \nmid \mu_{t,x'}, \forall x' \in \mathcal{X} \},
\]

and the maximal loss due to arm \( x \) is

\[
\epsilon_t(x) := \inf \{ \epsilon \in \mathbb{R}_+ | \mu_{t,x} + \epsilon > \mu_{t,x'}, \forall x' \in \mathcal{X} \},
\]

where \( \mu_{t,x} + \epsilon \) is the vector resulted by adding the scalar \( \epsilon \) to every component of the vector \( \mu_{t,x} \).

We illustrate the difference between the two measures in Definition 1 using an example. Let \( \mathcal{X} = \{A, B\} \) and at some stage \( t \), \( \mu_{t,A} = [0, 1]^\top \) and \( \mu_{t,B} = [2, 0]^\top \). According to Definition 1, \( \Delta_t(A) = 0 \) and \( \Delta_t(B) = 0 \), whereas \( \epsilon_t(A) = 2 \) and \( \epsilon_t(B) = 1 \). In this example, neither arm’s expected value of the feedback vector Pareto dominates the other, which the value of Pareto suboptimality gaps confirms. However, pulling arm \( A \) incurs a loss of \( 2 \) in the first objective, \( \epsilon_t(A) = 2 \), whereas pulling arm \( B \) incurs a loss of \( 1 \) in the second objective, \( \epsilon_t(B) = 1 \). As a result, the algorithm must pick arm \( B \) over \( A \), or in general, it must pick the arm with the least value of maximal loss in individual objectives. We are now ready to define the Pareto regret and cumulative maximal loss in individual objectives based on Pareto suboptimality gaps and maximal losses, respectively.

**Definition 2.** Let \( h_T := (z_1, x_1, y_1, \ldots, z_T, x_T, y_T) \) be a history of states, actions, and feedback vectors over the learning horizon \( T \). Then, the Pareto regret (PR) and cumulative maximal loss (CML) corresponding to history \( h_T \) are defined as

\[
\text{PR}(h_T) := \sum_{t \in [T]} \Delta_t(x_t), \quad \text{and} \quad \text{CML}(h_T) := \sum_{t \in [T]} \epsilon_t(x_t).
\]

Notice that the expert who always picks the arm with its feedback not dominated by any other arm achieves a zero Pareto regret. In bandit algorithms, it is assumed that the agent does not know the distribution of the feedback vector, which is consistent with the assumption of unknown environment dynamics in blending controllers. Under this assumption, a sublinear Pareto regret is often the best outcome one can expect from a bandit algorithm [30]. By a sublinear Pareto regret, we refer to one that satisfies

\[
\lim_{T \to \infty} \frac{1}{T} \text{PR}(h_T) = 0.
\]

**C. Problem Statement**

We state the problem of blending controllers using the notion of Pareto regret. Recall that we intend to formulate the problem in a way that it is compatible with an arbitrary number of controllers and objectives to be blended together. The total reward is the underlying performance and the total costs are the safety measures; therefore, the blending algorithm must maximize both the total reward and the negation of the total costs. Next, we formulate two problems:

**Problem 1.** Fix a control set \( \mathcal{X} \), time horizon \( T \), and a feature-mapping \( \psi : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}^d \). At each time-step \( t \in [T] \), an agent is asked to pick a controller \( x_t \in \mathcal{X} \) based on the feedback \( y \in \mathbb{R}^m \) and the context \( \Psi_t = \psi(z_t, x_t) \), where \( z_t \in \mathcal{Z} \) is the state of the agent at \( t \). Find an algorithm with a sublinear Pareto regret that, at all time-steps \( t \in [T] \), determines the agent’s choice of controller.

**Problem 2.** For the desired algorithm in Problem 1, characterize an upper bound on the algorithm’s cumulative maximal loss.

**III. Theory**

In this section, we introduce the multi-armed multi-objective contextual bandit algorithm that we propose for blending controllers. In the algorithm, as described in Algorithm 1, the arms are the input controllers, the reward and the negation of the auxiliary costs collectively form the bandit feedback, and the output of the fixed feature mapping forms the context. The algorithm is an upper-bound confidence (UCB) bandit algorithm, where the UCB indices are used to estimate the maximal loss in the individual objectives for every arm. The algorithm then picks the arm with the least estimated value thereof. We compute the UCB indices using a regularized least-squares estimation method, for which we show its accuracy increases as the time progresses. We assume the following for the structure of the feedback:

**Assumption 1 (LINEAR FEEDBACK WITH SUBGAUSSIAN NOISE).** At all time-steps \( t \in [T] \), the context \( \Psi_t \), and feedback \( y_t \), satisfy

\[
(y_t)_i = \theta_{*,i} : \Psi_t + \eta_t, \quad \forall i \in [m],
\]

where \( \theta_{*,i} \in \mathbb{R}^d \) is an unknown coefficient vector and for a fixed \( \sigma \in \mathbb{R}_+ \), \( \eta_t \) is conditionally \( \sigma \)-subgaussian, i.e., for all \( \alpha \in \mathbb{R} \),

\[
\mathbb{E} \left[ \exp(\alpha \eta_t) | \Psi_t, \eta_t, \ldots, \eta_{t-1} \right] \leq \exp(\sigma^2 \alpha^2 / 2).
\]

The above assumption is standard in linear contextual bandits [31], [32], [33]. Although it may be restrictive for some applications, later in our experiments, we test the algorithm in multiple environment settings of Safety Gym and we demonstrate that the assumption holds empirically.

The algorithm maintains an estimation of the unknown coefficient vector \( \theta_{*,i} \) for every objective \( i \in [m] \). Let \( \theta_{t,i} \) be the \( \ell^2 \)-regularized least-squares estimate of \( \theta_{*,i} \) corresponding to objective \( i \), and \( \lambda > 0 \) be a user-defined regularizing coefficient vector.
Algorithm 1: Blending controllers algorithm

Input: A set of controllers $\mathcal{X}$, feature mapping $\psi: \mathbb{Z} \times \mathcal{X} \mapsto \mathbb{R}^d$, regularization parameter $\lambda \geq \max\{1, L^2\}$, time horizon $T$, maximum failure probability tolerated $\delta$.

1. Initialize $V_0 = \lambda I_{d \times d}, \forall i \in [m]: \theta_{i,1} = 0_{d \times 1}, W_{0,i} = 0_{d \times 1}, O_1 = \mathcal{X}$.

2. for $t = 1$ to $T$ do
   3. Pull an arm $x_t \in O_t$ uniformly at random.
   4. Observe the state of the environment $z_t$ and feedback $y_t$, and context $\Psi_t := \psi(z_t, x_t)$.
   5. Update $V_t = V_{t-1} + \Psi_t \Psi_t^T$.
   6. for $i = 1$ to $m$ do
      7. Update $W_{t,i} = W_{t-1,i} + (y_t)_i \Psi_t$.
      8. Compute $\hat{\theta}_{t,i} = V_{t-1}^{-1} W_{t,i}$.
      9. Compute $\hat{C}^i_t$ by (6).
   10. for each arm $x \in \mathcal{X}$, compute the UCB index, $\tilde{\mu}_{t,x}$, by (8).
   11. for each arm $x \in \mathcal{X}$, compute the estimated maximal loss, $\hat{\epsilon}_t$, by (9).
   12. Update $O_{t+1} = \arg\min_{x \in \mathcal{X}} \hat{\epsilon}_t(x)$.

UBC index of arm $x$ for the $i$th objective is

$$\tilde{\mu}_{t,x} := \max_{\theta \in \hat{C}^i_t} \theta \cdot \psi(z_t, x).$$

Equation (7) is the evaluation of the support function of the ellipsoid $C^i_t$ at $\psi(z_t, x)$. Using the closed-form solution for the support function of an ellipsoid [35], we can write

$$\tilde{\mu}_{t,x} = \left( \hat{\theta}_{t,i} + \beta_i V^{-1}_t \psi(z_t, x) \right) \cdot \psi(z_t, x) = \hat{\theta}_{t,i} \cdot \psi(z_t, x) + \beta_i \|\psi(z_t, x)\|_{V^{-1}_t}.$$  

Subsequently, Algorithm 1 estimates the maximal losses in individual objectives due to every arm $x \in \mathcal{X}$ using the computed UCB indices as

$$\hat{\epsilon}_t(x) := \inf \{ \epsilon \in \mathbb{R}^+_+ | \tilde{\mu}_{t,x} + \epsilon \succ \tilde{\mu}_{t,x'}, \forall x' \in \mathcal{X} \},$$

and picks the arm with the least estimated value above. We are now ready to state the first main theorem, which alongside the algorithm itself, solves Problem 1.

**Theorem 1 (Sublinear Pareto Regret).** Let Assumption 1 hold and $S, L \in \mathbb{R}_+$. Assume that, for all objectives $i \in [m]$ and all stages $t \in [T]$, $\|\theta_{i,t}\|_2 \leq S$ and $\|\Psi_t\|_2 \leq L$. Then, with high probability $1 - \delta$, Algorithm 1 satisfies

$$\text{PR}(h_T) \leq O \left( \sqrt{T \log(T)} \right),$$

where $h_T$ is the history of states, actions and feedback vectors over the learning horizon $T$.

Proof idea. We first show that the adopted decision rule by Algorithm 1 picks an arm whose UCB index is not Pareto dominated by that of any other arm. To show this, assume that there exists an arm, $x' \in \mathcal{X} \setminus \arg\min_{x \in \mathcal{X}} \hat{\epsilon}(x)$, whose UCB index, $\tilde{\mu}_{t,x'}$, Pareto dominates that of the picked arm, $x_t$. Then, by the definition of $\hat{\epsilon}$ in (9), it follows that $\hat{\epsilon}(x') = 0$, and therefore, $x' \in \arg\min_{x \in \mathcal{X}} \hat{\epsilon}(x)$, which is in contradiction with the initial assumption. We then use the above argument and the results in [10] to generalize Lemma 11 from [31] to a multi-objective setting, which concludes the proof. See Appendix for the details of the proof. \hfill \Box

We now establish an upper bound on the average cumulative maximal loss of the algorithm during execution. We use the upper bound as an additional performance measure to Pareto regret in order to bridge the gap between the multi-objective performance measure and the performance in individual objectives. In the following theorem, we state our solution to Problem 2.

**Theorem 2.** With the same assumptions as in Theorem 1, for any $T \in \mathbb{N}$, the average cumulative maximal loss of Algorithm 1 satisfies

$$0 \leq \frac{1}{T} \text{CML}(h_T) \leq \frac{1}{T} \sum_{t=1}^{T} \hat{\epsilon}_t(x_t) + O(1/\sqrt{T}),$$

where $h_T$ is the history of states, actions and feedback vectors until stage $T$, and $x_t$ is the chosen arm at $t$.  

91
Proof. We first reformulate \( \epsilon \) in (2). For any arm \( x \in \mathcal{X} \), we can write
\[
\epsilon_t(x) = \max \left\{ \max_{k \in [m]} \max_{x' \in \mathcal{X}} \left( \mu_{t,x'}^k - \hat{\mu}_{t,x}^k \right), 0 \right\}.
\] (11)
Analogously, for all arms \( x \in \mathcal{X} \), we have that
\[
\hat{\epsilon}_t(x) = \max \left\{ \max_{k \in [m]} \max_{x' \in \mathcal{X}} \left( \hat{\mu}_{t,x'}^k - \hat{\mu}_{t,x}^k \right), 0 \right\}.
\] (12)
Let \( x_t \in \mathcal{X} \) be the arm that the algorithm picks at stage \( t \). Then, \( \hat{\epsilon}(x_t) = \min_{x \in \mathcal{X}} \epsilon_t(x) \). Then, for all arms \( x \in \mathcal{X} \) and objectives \( j \in [m] \), there exist an arm \( x^j \in \mathcal{X} \) and objective \( k \in [m] \) such that
\[
\hat{\mu}_{t,x}^j - \hat{\mu}_{t,x^j}^j \leq \hat{\mu}_{t,x}^k - \hat{\mu}_{t,x}^k.
\] (13)
By Lemma 1, with probability \( 1 - \delta \), it holds that \((\hat{\mu}_{t,x})_j \geq (\mu_{t,x})_j \). Rearranging (13), we arrive at
\[
\mu_{t,x}^j + \hat{\mu}_{t,x^j}^j \leq \mu_{t,x}^k + \hat{\mu}_{t,x}^k.
\] (14)
For all arms \( x \in \mathcal{X} \) and objectives \( j \in [m] \), we can write
\[
\mu_{t,x}^j - \mu_{t,x}^i \leq \hat{\mu}_{t,x}^j + \hat{\mu}_{t,x^j}^j - \hat{\mu}_{t,x}^i - \hat{\mu}_{t,x}^k
\leq (\theta_{t,j} - \theta_{t,i}) \Psi_{t} + \hat{\mu}_{t,x}^k - \hat{\mu}_{t,x}^k
\leq 2\beta_T \| \Psi_t \|_{V_{t-1}^{-1}} + \hat{\mu}_{t,x}^k - \hat{\mu}_{t,x}^k
\leq 2\beta_T \| \Psi_t \|_{V_{t-1}^{-1}} + \max_{k \in [m]} \max_{x' \in \mathcal{X}} \left( \hat{\mu}_{t,x'}^k - \hat{\mu}_{t,x}^k \right)
\leq 2\beta_T \| \Psi_t \|_{V_{t-1}^{-1}} + \hat{\epsilon}_t(x_t),
\] (15)
where the first inequality is resulted from (14), and the second inequality follows from the same argument as in the proof of Theorem 1 in Appendix. Equation (15) holds for all arms \( x \) and objectives \( j \), hence
\[
0 \leq \epsilon_t(x_t) \leq 2\beta_T \| \Psi_t \|_{V_{t-1}^{-1}} + \hat{\epsilon}_t(x_t).
\] (16)
By taking the average of the both sides of (16) alongside the results in Theorem 1, we arrive at (10).

Observe that the upper bound in (10) depends on the computed estimated maximal loss values and a diminishing term of the order \( O(1/\sqrt{T}) \). Thus, the computation of the upper bound does not impose any additional computational complexity. Furthermore, the UCB indices in (9) can be multiplied by importance weights in order to prioritize one objective over another, and this will not affect the results in Theorems 1 and 2.

IV. EXPERIMENTS

In this section, we demonstrate the proposed blending algorithm’s empirical success in blending a safe and a performant controller. We use the Safety Gym benchmark suite to perform our experiments. All Safety Gym environments use the framework of constrained Markov decision processes, and the agent is desired to maximize its expected total reward while minimizing its expected total cost. The numerical results in this section use the following Safety Gym environments:

- **Point-Goal.** A robot with two actuators, one that sets the thrust and the other sets the angle, has to reach the green zone in Figure 1a, while staying clear from the dangerous areas highlighted as blue circles. The robot itself is depicted in red.
- **Point-Push.** The same robot has to navigate the yellow box in Figure 1b to the green zone. In addition to the safety specifications in the Point-Goal environment, the agent has to avoid the erected pillars in the environment.
- **Car-Button.** A car with two independently actuated front wheels and a free-rolling rear wheel has to press the orange button that is highlighted as in Figure 1c. The agent has to avoid the purple moving boxes as a more sophisticated safety requirement.

The next step after fixing an environment is to choose the appropriate controllers to be blended. The Safety Gym suite focuses on the PPO and TRPO algorithms alongside their safety-aware versions, CPO, TRPO-Lagrangian, and PPO-Lagrangian. According to the Safety Gym benchmarks, the PPO and TRPO algorithms accumulate a higher total reward than their safety-aware versions; however, their total cost is higher as well. Therefore, we train the performant controller using either PPO or TRPO, and we train the safe controller using CPO, PPO-Lagrangian, or TRPO-Lagrangian. We train each of the instances of the safe or the performant controllers for 50 iterations. After the training is complete, we fix the controllers’ hyperparameters and feed them to Algorithm 1 as the input controllers.

As the next main ingredient, Algorithm 1 requires the feature-mapping \( \psi \) to be fixed. The feature mapping maps a pair of environment state and controller to a context in \( \mathbb{R}^d \). Notice that all of the algorithms above belong to the class of deep reinforcement-learning algorithms and they comprise a critic, which estimates the value function at each state, and an actor, which outputs the policy. For each of the controllers, in addition to the critic neural network, we train a similar feed-forward multi-layer perceptron neural network of size \((256, 256)\) with tanh activation functions to estimate the next-step reward and cost. Therefore, the context for each state-controller pair is the corresponding estimated next-step reward and cost.

By the condition of the linear-feedback assumption in (4) and the context structure above, Assumption 1 holds if (i) \( \theta_{s,\text{reward}} = [1 \ 0]^\top \) and \( \theta_{s,\text{cost}} = [0 \ 1]^\top \), and (ii) if the mismatch noise is bounded and has a zero mean, which makes it an instance of the subgaussian noise. Although we do not theoretically show that Assumption 1 holds in the experiments, we observed that, with \( \theta_{s,\text{reward}} = [1 \ 0]^\top \) and \( \theta_{s,\text{cost}} = [0 \ 1]^\top \), the mismatch noise has an empirical mean close to zero and a small empirical standard deviation.
For example, for the Point-Goal environment, we plot the next-step reward and cost versus their estimates in Figure 6. For the mismatch noise in reward, we observed an empirical mean of $4.1 \times 10^{-4}$ and standard deviation of $1.7 \times 10^{-3}$, and for the mismatch noise in cost, the corresponding values were $5.2 \times 10^{-4}$ and $5 \times 10^{-3}$, respectively. As a result, it is reasonable to assume that Assumption 1 holds in the experiments.

We now fix the rest of the hyperparameters required to run Algorithm 1. Recall that $L$ upper bounds the $\ell^2$-norm of the context at all time steps. Since the context approximates the next-step reward and cost, and in Safety Gym environments the maximum reward and cost is 1, the upper bound is $\sqrt{2}$; however, we account for the mismatch noise and set $L = 2$. Accordingly, we set $\lambda = 2$. We set $S = 1$ because the $\ell^2$-norm of $\theta^*_{\text{reward}}$ and $\theta^*_{\text{cost}}$ is one. For assigning a value to $\sigma$, based on the observations of the empirical standard deviation of the mismatch noise, we set $\sigma = 0.1$. Finally, we set $\delta = 0.1$ as we tolerate a failure probability of 0.1.

We run the blending algorithm with its inputs set as above and visualize a selection of the results in Figure 5. The first row corresponds to the average reward accumulated by the performant, the safe, and the blended controllers, and in the second row, we compare the controllers’ average costs. Finally, in the third row we evaluate the rate at which the algorithm employs the correct controller, i.e., it successfully avoids the controller with its feedback Pareto dominated by the other controller. In order to compute such a metric, at each decision step, we fix the environment behavior and separately employ each controller to reveal their true reward and cost. Then, we are able to establish the Pareto dominance relationship amongst them. Each data point in Figure 5 represents an average of the metrics of 30 episodes, with
Average Reward
Blended controller Baseline
−5
0
1 2 3 4 5 6 7 8 ... from the fact that $\beta_t$ is an increasing function of $t$. Therefore,
\[
\theta^*_t, j \cdot \psi(z_t, x') - \theta^*_t, j \cdot \psi(z_t, x) \leq 2\beta_T \|\Psi_t\|_{V^{-1}}.
\] (19)

As desired, the numerical results suggest that Algorithm 1 performs significantly better. At which the algorithm picks the correct controller suggest the reward is higher than the safe controller. The observed ratio at which the algorithm chooses among the set of arms whose UCB indices are noted the arm that is chosen by the algorithm and $z_t \in Z$ denote the state of the environment at stage $t$. For all objectives $i \in [m]$, let
\[
\hat{\mu}_t^i, x = \hat{\theta}_t, i \cdot \psi(z_t, x), \quad \text{with} \quad \hat{\theta}_t, i := \arg\max_{\theta \in C_t^i} \theta \cdot \psi(z_t, x).
\]
The algorithm chooses amongst the set of arms whose UCB indices are not Pareto dominated by that of any other arm; therefore, for each arm $x \in X$, there exists an objective $j \in [m]$ such that $\hat{\mu}_t^j, x \geq \hat{\mu}_t^j, x$. By Lemma 1, we have that with probability at least $1 - \delta$,
\[
\hat{\mu}_t^j, x = \max_{\theta \in C_t^j} \theta \cdot \psi(z_t, x) \geq \theta^*_j \cdot \psi(z_t, x).
\]
Hence, with probability at least $1 - \delta$, for all arms $x \in X$,
\[
\hat{\theta}_t, j \cdot \psi(z_t, x_t) \geq \theta^*_j \cdot \psi(z_t, x_t).
\] (17)

We now consider the case in which the algorithm has picked the wrong arm, i.e., there exists an arm $x' \in X$ such that $\theta^*_j \cdot \psi(z_t, x_t) < \theta^*_j \cdot \psi(z_t, x')$. Then, for any $t \geq 1$, we can write
\[
\theta^*_j \cdot \psi(z_t, x') - \theta^*_j \cdot \psi(z_t, x_t)
\leq \hat{\theta}_t, j \cdot \psi(z_t, x_t) - \theta^*_j \cdot \psi(z_t, x_t)
= (\hat{\theta}_t, j - \theta^*_j) \cdot \psi(z_t, x_t) + (\theta^*_j \cdot \psi(z_t, x_t)
\leq \|\hat{\theta}_t, j - \theta^*_j\|_{V_t} + \|\hat{\theta}_t, j - \theta^*_j\|_{V_t} \|\psi(z_t, x_t)\|_{V_t^{-1}}
\leq 2\beta_t \|\psi(z_t, x_t)\|_{V_t^{-1}} \leq 2\beta_T \|\psi(z_t, x_t)\|_{V_t^{-1}}.
\] (18)

The first inequality is a result of (17). Inequality (18) holds because of Hölder’s inequality. Finally the last inequality follows from the fact that $\beta_t$ is an increasing function of $t$. Therefore,
\[
\theta^*_j \cdot \psi(z_t, x') - \theta^*_j \cdot \psi(z_t, x_t) \leq 2\beta_T \|\Psi_t\|_{V_t^{-1}}.
\] (19)

V. CONCLUSIONS AND FUTURE WORK

We developed a contextual multi-armed multi-objective bandit algorithm to solve the problem of blending controllers. The algorithm achieves a sublinear Pareto regret, which characterizes its performance measure. We also derived an upper bound on the algorithm’s cumulative maximal loss, which shows how much cost or reward the algorithm sacrifices while execution. We empirically demonstrated the algorithm’s performance in the Safety Gym environments. The results show that the algorithm succeeds in learning an appropriate switching strategy to find a middle ground between them.

A question that we are interested to address in future works is to find the sufficient conditions under which the sublinearity of the Pareto regret implies an absolute improvement in the performance measure of the given safe controllers and in the safety measure of the given performant controller. In the experiments, we observed such a phenomenon in the tested Safety Gym environments; however, whether this holds in all of the applications of blending controllers is subject to ongoing research.

APPENDIX: PROOF OF THEOREM 1

Let $x_t \in X$ denote the arm that is chosen by the algorithm and $z_t \in Z$ denote the state of the environment at stage $t$. For all objectives $i \in [m]$, let
\[
\hat{\mu}_t^i, x = \hat{\theta}_t, i \cdot \psi(z_t, x_t), \quad \text{with} \quad \hat{\theta}_t, i := \arg\max_{\theta \in C_t^i} \theta \cdot \psi(z_t, x_t).
\]
The algorithm chooses amongst the set of arms whose UCB indices are not Pareto dominated by that of any other arm; therefore, for each arm $x \in X$, there exists an objective $j \in [m]$ such that $\hat{\mu}_t^j, x \geq \hat{\mu}_t^j, x$. By Lemma 1, we have that with probability at least $1 - \delta$,
\[
\hat{\mu}_t^j, x = \max_{\theta \in C_t^j} \theta \cdot \psi(z_t, x) \geq \theta^*_j \cdot \psi(z_t, x).
\]
Hence, with probability at least $1 - \delta$, for all arms $x \in X$,
\[
\hat{\theta}_t, j \cdot \psi(z_t, x_t) \geq \theta^*_j \cdot \psi(z_t, x_t).
\] (17)

We now consider the case in which the algorithm has picked the wrong arm, i.e., there exists an arm $x' \in X$ such that $\theta^*_j \cdot \psi(z_t, x_t) < \theta^*_j \cdot \psi(z_t, x')$. Then, for any $t \geq 1$, we can write
\[
\theta^*_j \cdot \psi(z_t, x') - \theta^*_j \cdot \psi(z_t, x_t)
\leq \hat{\theta}_t, j \cdot \psi(z_t, x_t) - \theta^*_j \cdot \psi(z_t, x_t)
= (\hat{\theta}_t, j - \theta^*_j) \cdot \psi(z_t, x_t) + (\theta^*_j \cdot \psi(z_t, x_t)
\leq \|\hat{\theta}_t, j - \theta^*_j\|_{V_t} + \|\hat{\theta}_t, j - \theta^*_j\|_{V_t} \|\psi(z_t, x_t)\|_{V_t^{-1}}
\leq 2\beta_t \|\psi(z_t, x_t)\|_{V_t^{-1}} \leq 2\beta_T \|\psi(z_t, x_t)\|_{V_t^{-1}}.
\] (18)

The first inequality is a result of (17). Inequality (18) holds because of Hölder’s inequality. Finally the last inequality follows from the fact that $\beta_t$ is an increasing function of $t$. Therefore,
\[
\theta^*_j \cdot \psi(z_t, x') - \theta^*_j \cdot \psi(z_t, x_t) \leq 2\beta_T \|\Psi_t\|_{V_t^{-1}}.
\] (19)
Notice that the upper bound in (19) is independent of index \( j \) and arm \( x' \). Therefore, the Pareto suboptimality gap of \( x_t, \Delta_t(x_t) \), is also upper bounded by \( 2 \beta_T \| \Psi_t \|_{V_T^{-1}} \). By the Cauchy-Schwarz inequality, we have that for all \( i \in [m] \) and all \( x \in \mathcal{X}, \theta_{z,i} \cdot \psi(x_i) \leq \| \theta_{z,i} \|_2 \| \Psi(x_i) \|_2 \leq SL \). Let \( (a \setminus b) := \max(a, b) \). Then, we can write

\[
\Delta_t(x_t) \leq \left( 2SL \land 2 \beta_T \| \Psi_t \|_{V_T^{-1}} \right) = 2 \beta_T \left( SL \land \| \Psi_t \|_{V_T^{-1}} \right) \leq 2 \beta_T \left( 1 \land \| \Psi_t \|_{V_T^{-1}} \right).
\]

Taking the sum of both sides of (20) and using the Cauchy-Schwarz inequality, we can write

\[
\Pr(h_T) \leq \sqrt{8 \beta_T^2 T \sum_{t=1}^{T} \left( 1 \land \| \Psi_t \|_{V_T^{-1}} \right)}.
\]

Finally, by Lemma 11 in [31] we have that with probability at least \( 1 - \delta \),

\[
\Pr(h_T) \leq 8 \sqrt{2Td \log(\lambda + TL/d)} \left( \sigma \sqrt{d \log \left( \frac{1 + TL^2/\lambda}{\delta} \right)} + \lambda^{1/2} S \right)^2,
\]

which concludes the proof.

REFERENCES

[1] H. Zhu, Z. Xiong, S. Magill, and S. Jagannathan, “An inductive synthesis framework for verifiable reinforcement learning,” in Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation, 2019, pp. 686–701.

[2] M. Alshiekh, R. Bloem, R. Ehlers, B. Königshofer, S. Niekum, and U. Topcu, “Safe reinforcement learning via shielding,” in Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[3] S. Sadraei and C. Beita, “Formal guarantees in data-driven model identification and control synthesis,” in Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control (part of CPS Week), 2018, pp. 147–156.

[4] N. Fulton and A. Platzer, “Safe reinforcement learning via formal methods: Toward safe control through proof and learning,” in Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[5] Y. Chow, O. Nachum, E. Duenez-Guzman, and M. Ghavamzadeh, “A lyapunov-based approach to safe reinforcement learning,” in Advances in neural information processing systems, 2018, pp. 8092–8101.

[6] E. Altman, Constrained Markov decision processes. CRC Press, 1999, vol. 7.

[7] Y. Chow, M. Ghavamzadeh, L. Janson, and M. Pavone, “Risk-constrained reinforcement learning with percentile risk criteria,” The Journal of Machine Learning Research, vol. 18, no. 1, pp. 6070–6120, 2017.

[8] G. Dalal, K. Dvijotham, M. Vecerik, T. Hester, C. Paduraru, and Y. Tassa, “Safe exploration in continuous action spaces,” arXiv preprint arXiv:1801.08757, 2018.

[9] M. M. Drugan and A. Nowe, “Designing multi-objective multi-armed bands algorithms: A study,” in The 2013 International Joint Conference on Neural Networks (IJCNN). IEEE, 2013, pp. 1–8.

[10] S. Lu, G. Wang, Y. Hu, and L. Zhang, “Multi-Objective Generalized Linear Bandits,” arXiv e-prints, p. arXiv:1905.12879, May 2019.

[11] A. Ray, J. Achiam, and D. Amodei, “Benchmarking safe exploration in deep reinforcement learning,” 2019.

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

[13] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, “Trust region policy optimization,” in International conference on machine learning, 2015, pp. 1889–1897.

[14] I. Y. Kim and O. De Weck, “Adaptive weighted sum method for multiobjective optimization: a new method for pareto front generation,” Structural and multidisciplinary optimization, vol. 31, no. 2, pp. 105–116, 2006.

[15] I. G. Liu, “On min-norm and min-max methods of multi-objective optimization,” Mathematical programming, vol. 103, no. 1, pp. 1–33, 2005.

[16] K. Van Moofaert, M. M. Dragan, and A. Nowé, “Scalarized multi-objective reinforcement learning: Novel design techniques,” in 2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL). IEEE, 2013, pp. 191–199.

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

[18] J. Achiam, D. Held, A. Tamar, and P. Abbeel, “Constrained policy optimization,” in Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, 2017, pp. 22–31.

[19] A. Verma, H. Le, Y. Yue, and S. Chaudhuri, “Imitation-projected programmatic reinforcement learning,” in Advances in Neural Information Processing Systems, 2019, pp. 15726–15737.

[20] L. Sha, “Using simplicity to control complexity,” IEEE Software, no. 4, pp. 20–28, 2001.

[21] S. Bak, K. Manamcheri, S. Mitra, and M. Caccamo, “Sandboxing controllers for cyber-physical systems,” in 2011 IEEE/ACM Second International Conference on Cyber-Physical Systems. IEEE, 2011, pp. 3–12.

[22] K. Lee and L. Sha, “A dependable online testing and upgrade architecture for real-time embedded systems,” in 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA ’05). IEEE, 2005, pp. 160–165.

[23] S. Bak, T. T. Johnson, M. Caccamo, and L. Sha, “Real-time reachability for verified simplex design,” in 2014 IEEE Real-Time Systems Symposium. IEEE, 2014, pp. 138–148.

[24] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynema, J. Gillula, and C. J. Tomlin, “A general safety framework for learning-based control in uncertain robotic systems,” IEEE Transactions on Automatic Control, vol. 64, no. 7, pp. 2737–2752, 2018.

[25] A. Benloucif, A.-T. Nguyen, C. Sentouh, and J.-C. Popiel, “Coop-erative trajectory planning for haptic shared control between driver and automation in highway driving,” IEEE Transactions on Industrial Electronics, vol. 66, no. 12, pp. 9846–9857, 2019.

[26] C. Ezeh, P. Trautman, C. Holloway, and T. Carlson, “Comparing shared control approaches for alternative interfaces: a wheelchair simulator experiment,” in 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC). IEEE, 2017, pp. 93–98.

[27] S. Jain and B. Argall, “Recursive bayesian human intent recognition in shared-control robotics,” in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018, pp. 3905–3912.

[28] A. Broad, T. Murphey, and B. Argall, “Learning models for shared control of human-machine systems with unknown dynamics,” arXiv preprint arXiv:1808.08268, 2018.

[29] N. Jansen, M. Cubuktepe, and U. Topcu, “Synthesis of shared control protocols with provable safety and performance guarantees,” in 2017 American Control Conference (ACC). IEEE, 2017, pp. 1866–1873.

[30] T. Lattimore and C. Szepesvári, Bandit algorithms. Cambridge University Press, 2020.

[31] Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári, “Improved algorithms for linear stochastic bandits,” in Advances in Neural Information Processing Systems, 2011, pp. 2312–2320.

[32] P. Ramaswami, E. R. and J. M. Tsitsiklis, “Linearly parameterized bandits,” Mathematics of Operations Research, vol. 35, no. 2, pp. 395–411, 2010.

[33] V. Dani, T. P. Hayes, and S. M. Kakade, “Stochastic linear optimization under bandit feedback,” 2008.

[34] P. Gohari, F. Djeumou, A. P. Vinod, and U. Topcu, “Blending controllers via multi-objective bandits,” arXiv preprint arXiv:2007.15755, 2020.

[35] S. Boyd, S. P. Boyd, and L. Vandenberghe, Convex optimization. Cambridge university press, 2004.