Blending Controllers via Multi-Objective Bandits

Parham Gohari*
Department of Electrical and Computer Engineering
University of Texas at Austin
pgohari@utexas.edu

Franck Djeumou*
Department of Electrical and Computer Engineering
University of Texas at Austin
fdjeumou@utexas.edu

Abraham P. Vinod
Oden Institute for Computational Engineering and Sciences
University of Texas at Austin
aby.vinod@gmail.com

Ufuk Topcu
Department of Aerospace and Mechanical Engineering
University of Texas at Austin
utopcu@utexas.edu

Abstract

Safety and performance are often two competing objectives in sequential decision-making problems. Existing performant controllers, such as controllers derived from reinforcement learning algorithms, often fall short of safety guarantees. On the contrary, controllers that guarantee safety, such as those derived from classical control theory, require restrictive assumptions and are often conservative in performance. Our goal is to blend a performant and a safe controller to generate a single controller that is safer than the performant and accumulates higher rewards than the safe controller. To this end, we propose a blending algorithm using the framework of contextual multi-armed multi-objective bandits. At each stage, the algorithm observes the environment's current context alongside an immediate reward and cost, which is the underlying safety measure. The algorithm then decides which controller to employ based on its observations. We demonstrate that the algorithm achieves sublinear Pareto regret, a performance measure that models coherence with an expert that always avoids picking the controller with both inferior safety and performance. We derive an upper bound on the loss in individual objectives, which imposes no additional computational complexity. We empirically demonstrate the algorithm's success in blending a safe and a performant controller in a safety-focused testbed, the Safety Gym environment. A statistical analysis of the blended controller's total reward and cost reflects two key takeaways: The blended controller shows a strict improvement in performance compared to the safe controller, and it is safer than the performant controller.

*Equal contribution

Preprint. Under review.

1 Introduction

Designing autonomous systems that are both safe and well-performing is a significant challenge in artificial intelligence research. On the one hand, reinforcement learning algorithms offer sophisticated controllers that perform a given task efficiently, yet they often fall short of safety guarantees [34, 3]. On the other hand, controllers that offer safety guarantees, *e.g.*, controllers derived from classical control theory, are often conservative in performance [29, 18, 13, 8]. An intuitive solution might be to blend a safe and a performant controller to obtain a single controller that is both safe and performant. In this paper, we investigate the possibility of blending such controllers.

We define "blending controllers" as learning a switching strategy between a given safe and a given performant controller using the observations and feedback from the environment. We assume that the environment dynamics are unknown, and that, upon taking an action, the environment issues the agent with a feedback vector containing a one-step reward and an auxiliary cost that measures the safety of the action. For example, the environment might issue a cost whenever the agent at the proximity of an obstacle. The framework of auxiliary costs for measuring safety is conventional in reinforcement learning algorithms that respect safety [4, 12, 14], and is not limited to this work. The performant controller achieves a higher expected total reward than the safe controller, whereas the safe controller has a lower expected total cost.

Switching between the safe and the performant controller without proper measures may take the agent to a state that neither the safe controller renders as safe nor had it been experienced by the performant controller's underlying reinforcement learning algorithm. We require a blending algorithm to justify its choice of controllers according to the *Pareto dominance relationship*, *i.e.*, we require the algorithm to avoid choosing a controller with both inferior safety and performance measures.

We formally state the problem of blending controllers as follows: Fix a safe and a performant controller. Let the environment associate every state-action pair with a two-dimensional feedback vector consisting of the one-step reward and safety measure. Consider an expert who always avoids choosing the controller with a Pareto dominated feedback vector. Then, design an online learning algorithm whose cumulative deviations from the expert's choice converges to zero on average.

In this paper, we propose a solution to the problem of blending controllers using *contextual multi-objective multi-armed bandit algorithms* [23], wherein the safe and performant controllers are the arm set, performance and safety are the objectives, and the observations from the environment are the context. The multi-objective formulation enables the algorithm to consider each safety requirement as a single objective, which simplifies the modeling of safety requirements. For example, in autonomous driving, we may consider keeping the vehicle centered in the lane as one objective and avoiding obstacles as another. Moreover, the algorithm is compatible with any number of input controllers. Therefore, for the example of autonomous driving, the algorithm may employ multiple safe controllers, each of which is safe with respect to some safety requirement. We utilize the above formulation to develop an algorithm that solves the problem of blending controllers.

The main contributions of this work are as follows:

- *Propose a novel contextual multi-armed multi-objective bandit algorithm for blending controllers.* The algorithm maintains an optimistic estimate of the next-step feedback for every arm. These estimations, on which the algorithm bases its choice of arms, become more accurate as time progresses. The algorithm then picks the arm with the smallest estimated loss in individual objectives, and we show that such a decision rule leads to picking an arm whose estimated next-step feedback vector is not Pareto dominated by any other arm.
- *Demonstrate that the algorithm's cumulative deviations from the expert's choice converges to zero on average.* We use the notion of *Pareto regret* [16] to penalize the agent whenever it chooses a Pareto dominated arm. We then show that the Pareto regret of the algorithm is sublinear, which implies that the average Pareto regret converges to zero asymptotically.
- *Establish a probabilistic bound on the average maximal loss in individual objectives.* The average maximal loss in individual objectives is a more intuitive performance measure for the problem of blending controllers than Pareto regret, a conventional performance score for multi-objective bandit algorithms. The bound is directly computed from the estimates that the algorithm maintains; therefore, it imposes no additional computational complexity and can be computed on-the-fly.

We use *Safety Gym* [28], a testbed for reinforcement learning algorithms that respect safety, to demonstrate the algorithm's effectiveness in blending controllers. In our experiments, we cover three levels of task and safety complexities in Safety Gym environments. We construct the performant and the safe controllers using deep reinforcement learning methods [27]. We generate the context for the proposed bandit algorithm using the action values estimated by the underlying neural networks. In each environment, an analysis of the statistics of the reward and cost of the blended controller confirms that the blended controller shows a significant improvement in its safety when compared to the performant controller and in its performance when compared to the safe controller.

The rest of this paper proceeds as follows: In Section 2, we fix the notation and definitions used throughout the paper followed by the problem statement. In Section 3, we introduce the algorithm that we propose for blending controllers as well as the theoretical developments. We discuss the numerical results in Section 4, and review the related works in Section 5. We provide conclusions and directions for future research in Section 6. Finally, we take a step back from the technicalities and discuss the potential societal impacts of this work in Section 7.

2 Preliminaries and problem statement

2.1 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 $n \in \mathbb{N}$, $[n] := \{1, \dots, n\}$. Let $v \in \mathbb{R}^n$ and $i \in [n]$, then $v^\top$ 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.,* $\|v\|_W^2 := v^\top W v$, 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 $v_j < u_j$.

2.2 Contextual multi-objective bandits

In this section, we establish the definitions corresponding to c*ontextual multi-armed multi-objective bandits*. We denote the arm set of the bandit by $\mathcal{X}$, the environment state space by $\mathcal{Z}$, and the learning horizon by T. At any stage $t \in [T]$, a context vector $\Psi_t \in \mathbb{R}^d$ characterizes the agent's observation of the environment. The context vector is defined using a known feature mapping $\psi : \mathcal{Z} \times \mathcal{X} \mapsto \mathbb{R}^d$. Specifically, let $z_t \in \mathcal{Z}$ be the current state of the environment and $x_t \in \mathcal{X}$ 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 feedback vector $y_t \in \mathbb{R}^m$. The feedback consists of m objectives that the bandit seeks to simultaneously optimize. Without loss of generality, we assume that the objective measurements are normalized such that each unit measurement has equal importance amongst all objectives.

We use the notion of *Pareto regret* [16] as the underlying performance measure in multi-objective bandits to penalize the agent whenever it picks an arm whose corresponding feedback vector is Pareto dominated by another arm. We also introduce an additional performance measure, the *cumulative maximal loss*, which provides a finer criterion to distinguish between the set of non-dominated arms. In the following definition, we define *Pareto suboptimality gap* [26] and the maximal loss in individual objectives, which we later use to compute the Pareto regret and cumulative maximal loss of the algorithm.

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

$$\Delta_t(x) := \inf \{\epsilon \in \mathbb{R}_+ \mid \mu_{t,x} + \epsilon \not\succ \mu_{t,x'}, \forall x' \in \mathcal{X}\}, \tag{1}$$

and the maximal loss due to arm x is

$$\epsilon_t(x) := \inf \{\epsilon \in \mathbb{R}_+ \mid \mu_{t,x} + \epsilon \succ \mu_{t,x'}, \forall x' \in \mathcal{X}\}. \tag{2}$$

We illustrate the difference between the two measures in Definition 1 using a simple example. Let $\mathcal{X} = \{A, B\}$ and at some stage t, $\mu_{t,A} = [0 \quad 1]^\top$ and $\mu_{t,B} = [2 \quad 0]^\top$. 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.

Next, we define Pareto regret and cumulative maximal loss based on a sequence of 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*

$$\mathrm{PR}(h_T) := \sum_{t\in[T]} \Delta_t(x_t), \qquad \text{and} \qquad \mathrm{CML}(h_T) := \sum_{t\in[T]} \epsilon_t(x_t). \tag{3}$$

The expert, who always picks a non-dominated arm, achieves a zero Pareto regret. In bandit algorithms, however, 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 that one can expect from a bandit algorithm [23], *i.e.,*

$$\lim_{T\to\infty} \frac{1}{T}\mathrm{PR}(h_T) = 0. \tag{4}$$

2.3 Problem statement

We consider a learning agent who is provided with a set of pre-defined controllers and seeks to pick the controller whose m-dimensional feedback from the environment is not Pareto dominated by another controller's feedback. We assume that the agent has access to a feature mapping, ψ, which generates the context vector at each state of the environment. With the given controllers as the arms of a contextual multi-armed multi-objective bandit, we formulate two problems.

Problem 1. *Design an online algorithm for blending controllers that achieves a sublinear Pareto regret.*

We address Problem 1 under the following assumption on the structure of the feedback.

Assumption 1 (LINEAR FEEDBACK WITH SUBGAUSSIAN NOISE)**.** *At all stages $t \in [T]$, the context vector, Ψ_t, and feedback vector, y_t, satisfy*

$$(y_t)_i = \theta_{*,i} \cdot \Psi_t + \eta_t, \quad \forall i \in [m], \tag{5}$$

where $\theta_{*,i} \in \mathbb{R}^d$ is an unknown coefficient vector and for a fixed $\sigma \in \mathbb{R}_+$, η_t is conditionally σ-subgaussian, i.e.,

$$\mathbf{E}\left[\exp\left(\alpha\eta_t\right) \mid \Psi_1, \ldots, \Psi_t, \eta_1, \ldots, \eta_{t-1}\right] \leq \exp\left(\sigma^2\alpha^2/2\right), \quad \forall \alpha \in \mathbb{R}. \tag{6}$$

Problem 2. *Characterize an upper bound on the cumulative maximal loss of the proposed bandit algorithm for blending controllers.*

3 Theoretical contributions

We propose Algorithm 1 for blending controllers. The algorithm is a bandit algorithm that picks an arm based on an upper confidence bound (UCB) index that it computes for each arm. Based on the UCB indices, the algorithm estimates the maximal loss in the individual objectives corresponding to each arm. It then picks the arm with the least estimated loss. We compute the UCB indices using regularized least squares to exploit the results in [1], in which it is shown that the accuracy of such estimates increases as time progresses. In this section, we first show that the algorithm achieves a sublinear Pareto regret as the proposed solution to Problem 1. We then state our solution to Problem 2, wherein we establish an upper bound on the cumulative maximal loss of the algorithm.

Algorithm 1: Blending controllers algorithm

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

Initialize $V_0 = \lambda I_{d\times d}, \forall i \in [m] : \hat{\theta}_{1,i} = 0_{d\times 1}, W_{0,i} = 0_{d\times 1}, \mathcal{O}_0 = \mathcal{X}$.
for $t = 1$ **to** T **do**:
  Pull an arm $x_t \in \mathcal{O}_t$ uniformly at random.
  Observe the state of the environment z_t and feedback y_t, and let $\Psi_t := \psi(z_t, x_t)$.
  Update $V_t = V_{t-1} + \Psi_t {\Psi_t}^\top$.
  for $i = 1$ **to** m **do**:
    Update $W_{t,i} = W_{t-1,i} + (y_t)_i \Psi_t$.
    Compute $\hat{\theta}_{t,i} = V_t^{-1} W_{t,i}$.
    Compute C_t^i by (8).
  For each arm $x \in \mathcal{X}$, compute the UCB index, $\hat{\mu}_{t,x}$, by (10).
  For each arm $x \in \mathcal{X}$, compute the estimated maximal loss, $\hat{\epsilon}_t$, by (11).
  Update $\mathcal{O}_{t+1} = \operatorname{argmin}_{x\in\mathcal{X}} \hat{\epsilon}_t(x)$.

For all objectives $i \in [m]$, and let $\hat{\theta}_{t,i}$ be the ℓ^2-regularized least-squares estimate of the unknown coefficient vector $\theta_{*,i}$ with a user-defined regularizing parameter $\lambda > 0$, *i.e.,*

$$\hat{\theta}_{t,i} := \left(\Psi_{1:t}^\top \Psi_{1:t} + \lambda I\right)^{-1} \Psi_{1:t}^\top Y_{1:t}^i, \tag{7}$$

where $\Psi_{1:t}$ is the matrix with rows equal to $\Psi_1^\top, \Psi_2^\top, \ldots, \Psi_t^\top$ and $Y_{1:t}^i = [(y_1)_i, \ldots, (y_t)_i]^\top$. In Algorithm 1, we implement a memory-efficient incremental implementation of (7), see Appendix A. The following lemma shows that, for all objectives $i \in [m]$, the estimated coefficient vector, $\hat{\theta}_{t,i}$, is statistically close to its true value, $\theta_{*,i}$.

Lemma 1 (Theorem 2 in [1])**.** *Let Assumption 1 hold and $\{\Psi_t\}_{t=1}^\infty$ be an $\mathbb{R}^d$-valued stochastic process, where $\Psi_t := \psi(z_t, x_t)$. For any $t \geq 1$, define*

$$V_t := V_0 + \sum_{s=1}^{t} \Psi_s {\Psi_s}^\top,$$

where $V_0 = \lambda I_{d\times d}$, and $\lambda > 0$. Furthermore, let $S, L \in \mathbb{R}_+$ and assume that, for all objectives $i \in [m]$ and stages $t \in \mathbb{N}$, $\|\theta_{*,i}\|_2 \leq S$ and $\|\Psi_t\|_2 \leq L$. Then, for any $\delta > 0$, with probability at least $1 - \delta$, the true coefficient vector, $\theta_{*,i}$, lies within ellipsoid C_t^i defined as

$$C_t^i := \left\{ \theta \in \mathbb{R}^d : \left\| \theta - \hat{\theta}_{t,i} \right\|_{V_t} \leq \beta_t := \sigma \sqrt{d \log\left(\frac{1 + tL^2/\lambda}{\delta} \right)} + \lambda^{1/2} S \right\}. \tag{8}$$

At each stage $t \in [T]$, we use confidence ellipsoids C_t^i to compute the UCB indices corresponding to every arm $x \in \mathcal{X}$ and objective $i \in [m]$. In particular, the UCB index of arm x for objective i is

$$\hat{\mu}_{t,x}^i := \max_{\theta \in C_t^i} \theta \cdot \psi(z_t, x). \tag{9}$$

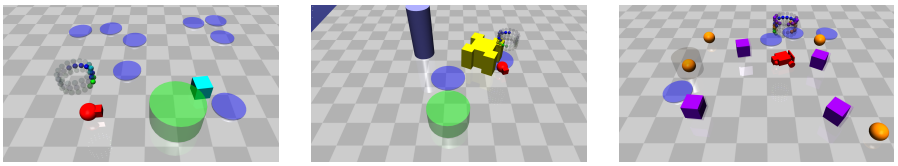
Equation (9) is the evaluation of the support function of ellipsoid C_t^i at vector $\psi(z_t, x)$. Using the closed-form solution for the support function of an ellipsoid [9], we can write

$$\hat{\mu}_{t,x}^i = \left(\hat{\theta}_{t,i} + \frac{\beta_t V_t^{-1} \psi(z_t, x)}{\|\psi(z_t, x)\|_{V_t^{-1}}} \right) \cdot \psi(z_t, x) = \hat{\theta}_{t,i} \cdot \psi(z_t, x) + \beta_t \|\psi(z_t, x)\|_{V_t^{-1}}. \tag{10}$$

Subsequently, Algorithm 1 estimates the maximal losses using the computed UCB indices as

$$\hat{\epsilon}_t(x) := \inf \left\{ \epsilon \in \mathbb{R}_+ \mid \hat{\mu}_{t,x} + \epsilon \succ \hat{\mu}_{t,x'}, \forall x' \in \mathcal{X} \right\}, \quad \forall x \in \mathcal{X}, \tag{11}$$

and picks the arm with the least estimated maximal loss. In Lemma 2, we show that such a decision rule between the arms picks an arm from the set of non-dominated arms.



a: *Point-Goal* b: *Point-Push* c: *Car-Button*

Figure 1: The selected Safety Gym environments for the experiments in Section 4.

Lemma 2. *In Algorithm 1, let $x_t \in \mathcal{X}$ denote the arm picked by the algorithm at stage $t \in [T]$. Then, the UCB index corresponding to arm x_t is not Pareto dominated by any other arm's UCB index.*

Proof. See Appendix B □

We now state the main theorem of this paper, 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 $t \in [T]$, $\|\theta_{*,i}\|_2 \leq S$ and $\|\Psi_t\|_2 \leq L$. Then, with high probability $1-\delta$, Algorithm 1 satisfies*

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

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

Proof. See Appendix C. □

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 and performance in individual objectives. In the following theorem, we state our solution to Problem 2.

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

$$0 \leq \frac{1}{T}\mathrm{CML}(h_T) \leq \frac{1}{T}\sum_{t=1}^{T}\hat{\epsilon}_t(x_t) + \mathcal{O}(1/\sqrt{T}), \tag{12}$$

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

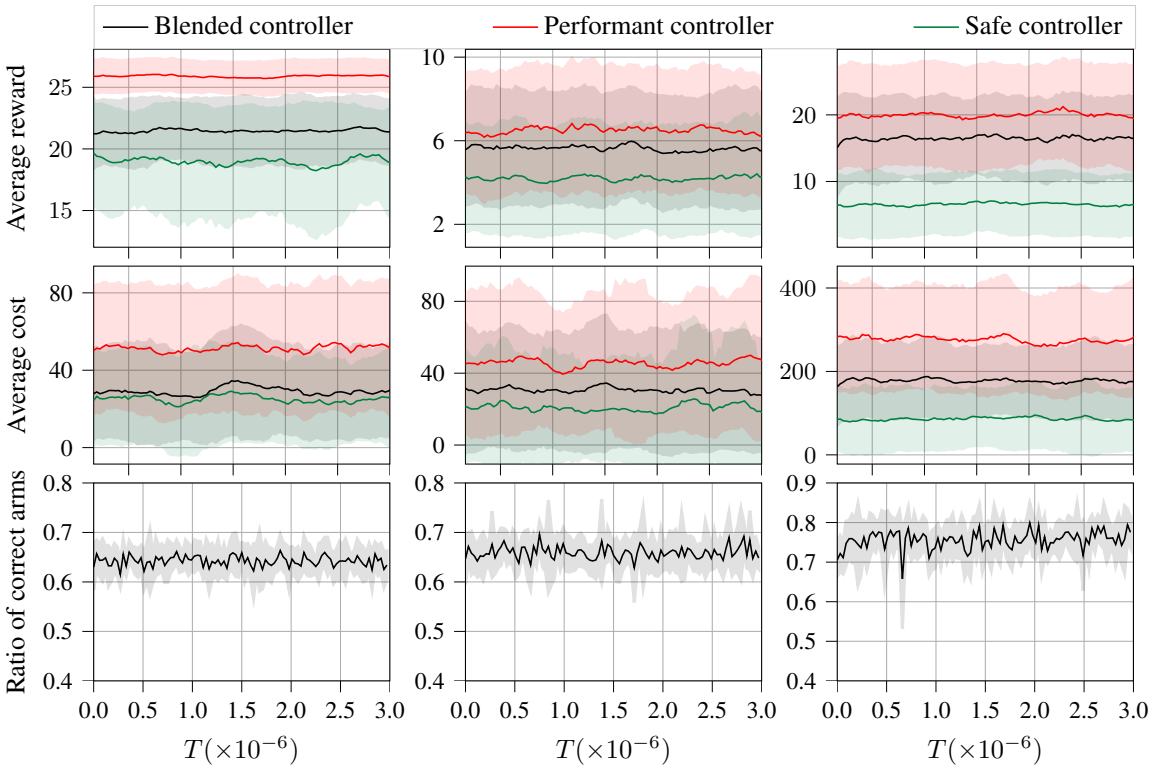
Proof. See Appendix D. □

Observe that the upper bound in (12) depends on the computed estimated maximal loss values and a diminishing term of the order $\mathcal{O}(1/\sqrt{T})$. Thus, the proposed upper bound does not impose any additional computational complexity in computation.

4 Experiments

In this section, we use the Safety Gym suite [28] to demonstrate the proposed algorithm's effectiveness in blending controllers in practice. We find Safety Gym a suitable testbed because (i) state-of-the-art reinforcement learning algorithms that respect safety have been benchmarked in all of its varied environments, and (ii) similar to this paper, it uses the framework of auxiliary cost functions to enforce safety requirements. In our experiments, we chose the Point-Goal, Point-Push, and Car-Button Safety Gym environments, with each environment described as follows:

- *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.



a: *Point-Goal* b: *Point-Push* c: *Car-Button*

Figure 2: Simulation results of testing Algorithm 1 in the selected Safety Gym environments in Section 4. The safe controllers used in a, b, and c are derived from PPO-Lagrangian, CPO, and TRPO-Lagrangian algorithms, respectively. The performant controllers are learned by PPO in a and TRPO in both b and c. The confidence clouds correspond to the standard deviation of each data point.

- *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.

We now describe and justify our choices of $\mathcal{X}$, the set of input controllers, ψ, the feature mapping, and the hyper parameters, λ, L, S and σ that are required by Algorithm 1.

Input controllers $\mathcal{X}$. For each environment, we use the Safety Gym benchmark suite [28] to identify the suitable choice of the safe and the performant controllers. We use reinforcement learning algorithms such as trust region policy optimization (TRPO) [30] and proximal policy optimization (PPO) [31] to generate the performant controller. According to the above benchmarking, these controllers show a superior performance in terms of their total reward; however, they perform poorly in terms of safety. We generate the safe controller using constrained reinforcement learning algorithms such as constrained policy optimization (CPO) [2] or a combination of Lagrangian methods with PPO and TRPO. In contrast to TRPO and PPO, these algorithms have been shown to meet their safety requirements in the sense that their average cost is below a fixed threshold; however, they are conservative in performance.

Feature mapping ψ. The above choices of the safe and the performant controllers belong to the family of deep reinforcement learning algorithms [27], which use a neural network to estimate the action values at each state of the environment. We incorporate the underlying neural networks as the feature mapping to generate the context at each state. Precisely, each controller uses a separate feedforward multilayer perceptron network of size $(256, 256)$ with $\tanh$ activation functions. In

practice, the trained neural networks provide accurate approximations of the next-step reward and cost, and therefore, we consider that Assumption 1 holds.

Hyperparameters. Based on the observed values of the output layer of the neural networks, we set L, the upper bound on the 2-norm of the feature vector, to 1, and we subsequently set $\lambda = 1$. Since the neural networks directly estimate the reward and cost, we expect the true coefficient vector to be at the vicinity of either $[1 \quad 0]^\top$ or $[0 \quad 1]^\top$; therefore, we set S, the upper bound on the 2-norm of the true coefficient vector θ_*, to 1.5. We choose a small value for σ because we expect the variance of the mismatch noise to be small. In particular, we set $\sigma = 0.1$.

We visualize the results of each experiment with plots as depicted in Figure 2. The first row compares the average reward corresponding to the performant, the safe, and the blended controller. In the second row, we compare the blended controller's safety with its safe and performant counterparts by comparing their average cost. 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 Pareto dominated reward and cost feedback. In order to find 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 2 represents an average of the metrics of 30 episodes, with each episode length fixed at 1,000 iterations.

As desired, the numerical results suggest that the proposed algorithm for blending controllers is an effective method of finding a controller that improves the safety of a given performant controller and the performance of a given safe controller. An analysis of the standard deviation across batches of 30,000 iterations supports the claim that the average cost and reward of the blended controller lie in-between the given performant and safe controllers. Additionally, the ratio at which the correct controller is picked is always above 0.5, which indicates that compared to naively switching between the safe and the performant controller at random, Algorithm 1 performs significantly better.

We used a desktop computer with Intel Core $i9$-9900, $3.10\,\mathrm{GHz} \times 16$ processors and $32\,\mathrm{Gb}$ of RAM for the experiments. Each iteration of the *Point-Goal*, *Point-Push*, and *Car-Button* environments took $2\,\mathrm{ms}$, $3\,\mathrm{ms}$, and $5\,\mathrm{ms}$, respectively. See [19] for all codes and datasets used in this section.

5 Related Work

We review existing work on constrained reinforcement learning algorithms that can enforce safety. We share the use of auxiliary costs to model safety with constrained reinforcement learning algorithms, such as CPO and PPO-Lagrangian. Although these algorithms show promise in the *Safety Gym* benchmark suite, even slight changes to safety specifications require learning a new policy from scratch. In contrast, the proposed modular approach of blending controllers allows the decision-maker to simply blend the outdated policy with a new policy that satisfies the new constraint.

In [33], the authors propose a *programmatic* reinforcement learning algorithm that shares the same modular methodology as employed in this paper. Although the proposed algorithm improves upon the performance of the input controllers, it does not consider safety as an objective, which is the main concern of this paper. In another approach, the *simplex architecture* [32, 5, 24, 6], a decision logic for controlling a plant is developed using a safe and a performant controller. In these works, the decision logic is computed by either a Lyapunov function or a reachability analysis. These approaches require prior knowledge of the environment dynamics, whereas we do not assume any such prior knowledge.

The work in [11] introduces a regularization scheme for a policy gradient reinforcement learning algorithm, such that the policy updates occur at the vicinity of a given reference controller. The introduced methodology splits the environment dynamics into a sum of a known and an unknown environment dynamics. Then, the authors show that the learned policy inherits the Lyapunov stability guarantees that the reference controller provides for the known environment dynamics. However, these guarantees are not applicable for an environment with fully unknown dynamics. In contrast, the safety modeling employed in this paper is specialized for such environments.

Finally, we review *shared control protocols* [10, 21, 15, 7, 17, 20], wherein a robot's commands are blended with its human user. Shared control protocols and blending controllers have the same goal of balancing the safety and performance of their given controllers. The algorithms in [7, 17, 20] use a weighted sum of the given controllers' outputs, whereas in this work, we switch between the given controllers. As a result, we are not restricted to continuous control signals. In [10], the environment

dynamics are unknown but assumed to be linear. Finally, the authors in [21] use semi-definite programming for blending controllers, which requires prior knowledge of the environment dynamics and the confidence level of each controller. In contrast to [10] and [21], the approach proposed in this paper is model-free.

6 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 suite. The simulation results show that the algorithm succeeds in learning the appropriate switching strategy in the sense that the algorithm's total reward is higher than its given safe controller, and it accumulates less cost than the given performant controller.

For the future work, we are interested in considering adversarial environments to investigate when the Pareto regret achieved by Algorithm 1 is optimal. We will also consider a generalized UCB-based decision rule beyond the comparison of the estimated maximal losses. There, we are interested in the sufficient conditions under which the algorithm maintains the sublinearity of its Pareto regret.

7 Broader impacts

In this section, we retreat from the technicalities into the broader societal impacts of this work. We formulated the problem of blending controllers to address the safety concerns that arise with AI systems, for example safety concerns in autonomous driving [22]. Blending controllers improves the safety of its input controllers while it is blind to the algorithm that drives them. Such blindness helps resolving the privacy concerns that may keep industrial companies from sharing their breakthroughs in safe control algorithms. Blending state-of-the-art safe controllers while protecting the privacy of their underlying algorithms may prompt industrial companies to at least share the output of their controllers in the name of protecting human lives.

On the other hand, even the most innocent intentions may lead to negative consequences. Multiple studies have found human users trusting automated systems in inappropriate circumstances, see [25] and references therein. Improving the safety of artificial intelligent systems may exacerbate such over-reliance. We emphasize that although blending controllers improves the safety of the overall system, its level of safety depends on its input safe controllers. Therefore, it is crucial that potential users be warned to perceive any improvements in automation safety as a lower chance of safety hazards and not an elimination of its vulnerabilities.

References

[1] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, pages 2312–2320, 2011.

[2] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 22–31. JMLR. org, 2017.

[3] Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[4] Eitan Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

[5] Stanley Bak, Taylor T Johnson, Marco Caccamo, and Lui Sha. Real-time reachability for verified simplex design. In *2014 IEEE Real-Time Systems Symposium*, pages 138–148. IEEE, 2014.

[6] Stanley Bak, Karthik Manamcheri, Sayan Mitra, and Marco Caccamo. Sandboxing controllers for cyber-physical systems. In *2011 IEEE/ACM Second International Conference on Cyber-Physical Systems*, pages 3–12. IEEE, 2011.

[7] Amir Benloucif, Anh-Tu Nguyen, Chouki Sentouh, and Jean-Christophe Popieul. Cooperative trajectory planning for haptic shared control between driver and automation in highway driving. *IEEE Transactions on Industrial Electronics*, 66(12):9846–9857, 2019.

[8] Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. In *Advances in neural information processing systems*, pages 908–918, 2017.

[9] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[10] Alexander Broad, Todd Murphey, and Brenna Argall. Learning models for shared control of human-machine systems with unknown dynamics. *arXiv preprint arXiv:1808.08268*, 2018.

[11] Richard Cheng, Abhinav Verma, Gabor Orosz, Swarat Chaudhuri, Yisong Yue, and Joel W Burdick. Control regularization for reduced variance reinforcement learning. *arXiv preprint arXiv:1905.05380*, 2019.

[12] Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research*, 18(1):6070–6120, 2017.

[13] Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. In *Advances in neural information processing systems*, pages 8092–8101, 2018.

[14] Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018.

[15] Anca D Dragan and Siddhartha S Srinivasa. A policy-blending formalism for shared control. *The International Journal of Robotics Research*, 32(7):790–805, 2013.

[16] Madalina M Drugan and Ann Nowe. Designing multi-objective multi-armed bandits algorithms: A study. In *The 2013 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2013.

[17] Chinemelu Ezeh, Pete Trautman, Catherine Holloway, and Tom Carlson. Comparing shared control approaches for alternative interfaces: a wheelchair simulator experiment. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 93–98. IEEE, 2017.

[18] Nathan Fulton and André Platzer. Safe reinforcement learning via formal methods: Toward safe control through proof and learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[19] Parham Gohari. Blending controllers. `https://github.com/parhamgohari/blending-controllers`, 2013.

[20] Siddarth Jain and Brenna Argall. Recursive bayesian human intent recognition in shared-control robotics. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3905–3912. IEEE, 2018.

[21] Nils Jansen, Murat Cubuktepe, and Ufuk Topcu. Synthesis of shared control protocols with provable safety and performance guarantees. In *2017 American Control Conference (ACC)*, pages 1866–1873. IEEE, 2017.

[22] Philip Koopman and Michael Wagner. Autonomous vehicle safety: An interdisciplinary challenge. *IEEE Intelligent Transportation Systems Magazine*, 9(1):90–96, 2017.

[23] Tor Lattimore and Csaba Szepesvári. Bandit algorithms. *preprint*, page 28, 2018.

[24] Kihwal Lee and Lui 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)*, pages 160–165. IEEE, 2005.

[25] Michael Lewis, Katia Sycara, and Phillip Walker. The role of trust in human-robot interaction. In *Foundations of trusted autonomy*, pages 135–159. Springer, Cham, 2018.

[26] Shiyin Lu, Guanghui Wang, Yao Hu, and Lijun Zhang. Multi-Objective Generalized Linear Bandits. *arXiv e-prints*, page arXiv:1905.12879, May 2019.

[27] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[28] Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning.

[29] Sadra Sadraddini and Calin Belta. 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)*, pages 147–156, 2018.

[30] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897, 2015.

[31] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[32] Lui Sha. Using simplicity to control complexity. *IEEE Software*, (4):20–28, 2001.

[33] Abhinav Verma, Hoang Le, Yisong Yue, and Swarat Chaudhuri. Imitation-projected programmatic reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 15726–15737, 2019.

[34] He Zhu, Zikang Xiong, Stephen Magill, and Suresh Jagannathan. An inductive synthesis framework for verifiable reinforcement learning. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 686–701, 2019.

Appendices

A Incremental ℓ^2-regularized least squares

Recall that at stage t, the ℓ^2-regularized least-squares estimate of $\theta_{*,i}$, the coefficient vector corresponding to objective i, is

$$\hat{\theta}_{t,i} := \left(\Psi_{1:t}^\top \Psi_{1:t} + \lambda I\right)^{-1} \Psi_{1:t}^\top Y_{1:t}^i. \tag{13}$$

Let $V_0 := \lambda I$, $\forall i \in [m] : W_{0,i} := 0_{d\times 1}$, and for all $t \geq 1$,

$$V_t := \Psi_{1:t}^\top \Psi_{1:t} + \lambda I \quad \text{and} \quad W_{t,i} := \Psi_{1:t}^\top Y_{1:t}^i. \tag{14}$$

Then, at stage $t+1$, we can write

$$V_{t+1} = \begin{bmatrix} \Psi_1 & \dots & \Psi_t & \Psi_{t+1} \end{bmatrix} \begin{bmatrix} \Psi_1^\top \\ \vdots \\ \Psi_t^\top \\ \Psi_{t+1}^\top \end{bmatrix} + \lambda I = V_t + \Psi_{t+1}\Psi_{t+1}^\top, \tag{15}$$

and

$$W_{t+1,i} = \begin{bmatrix} \Psi_1 & \dots & \Psi_t & \Psi_{t+1} \end{bmatrix} \begin{bmatrix} (y_1)_i \\ \vdots \\ (y_t)_i \\ (y_{t+1})_i \end{bmatrix} = W_{t,i} + (y_{t+1})_i \Psi_{t+1}. \tag{16}$$

By (15) and (16), we arrive at

$$\hat{\theta}_{t,i} = V_t^{-1} W_{t,i}, \quad \forall i \in [m], \tag{17}$$

which no longer requires storing all the values of Ψ and y.

B Proof of Lemma 2

Assume that there exists an arm, $x' \in \mathcal{X} \backslash \operatorname{argmin}_{x \in \mathcal{X}} \hat{\epsilon}(x)$, whose UCB index, $\hat{\mu}_{t,x'}$, Pareto dominates that of the picked arm, x_t. Then, by the definition of $\hat{\epsilon}$ in (11), it follows that $\hat{\epsilon}(x') = 0$, and therefore $x' \in \operatorname{argmin}_{x \in \mathcal{X}} \hat{\epsilon}(x)$, which is in contradiction with the initial assumption.

C Proof of Theorem 1

Let $x_t \in \mathcal{X}$ denote the arm that is chosen by the algorithm and $z_t \in \mathcal{Z}$ denote the state of the environment at stage t. For all objectives $i \in [m]$, let

$$\hat{\mu}^i_{t,x_t} = \tilde{\theta}_{t,i} \cdot \psi(z_t, x_t), \quad \text{with} \quad \tilde{\theta}_{t,i} := \operatorname*{argmax}_{\theta \in \mathcal{C}^i_t} \theta \cdot \psi(z_t, x_t). \tag{18}$$

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 $x \in \mathcal{X}$, there exists an objective $j \in [m]$ such that $\hat{\mu}^j_{t,x_t} \geq \hat{\mu}^j_{t,x}$. By Lemma 1, we have that with probability at least $1 - \delta$,

$$\hat{\mu}^j_{t,x} = \max_{\theta \in \mathcal{C}^j_t} \theta \cdot \psi(z_t, x) \geq \theta_{*,j} \cdot \psi(z_t, x).$$

Hence, with probability at least $1 - \delta$, for all arms $x \in \mathcal{X}$,

$$\tilde{\theta}_{t,j} \cdot \psi(z_t, x_t) \geq \theta_{*,j} \cdot \psi_t(z_t, x). \tag{19}$$

We now consider the case in which the algorithm has picked the wrong arm, *i.e.,* there exists an arm $x' \in \mathcal{X}$ 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

$$\begin{aligned}
\theta_{*,j} \cdot \psi(z_t, x') - \theta_{*,j} \cdot \psi(z_t, x_t) &\leq \tilde{\theta}_{t,j} \cdot \psi(z_t, x_t) - \theta_{*,j} \cdot \psi(z_t, x_t) \\
&= (\tilde{\theta}_{t,j} - \hat{\theta}_{t,j}) \cdot \psi(z_t, x_t) + (\hat{\theta}_{t,j} - \theta_{*,j}) \cdot \psi(z_t, x_t) \\
&\leq \left(\|\tilde{\theta}_{t,j} - \hat{\theta}_{t,j}\|_{V_t} + \|\hat{\theta}_{t,j} - \theta_{*,j}\|_{V_t} \right) \|\psi(z_t, x_t)\|_{V_t^{-1}} \qquad (20) \\
&\leq 2\beta_t \|\psi(z_t, x_t)\|_{V_t^{-1}} \leq 2\beta_T \|\psi(z_t, x_t)\|_{V_t^{-1}}.
\end{aligned}$$

The first inequality is a result of (19). Inequality (20) holds because of the Hölder's inequality. Finally the last inequality follows from the fact that β_t is an increasing function of t. Therefore,

$$\theta_{*,j} \cdot \psi(z_t, x') - \theta_{*,j} \cdot \psi(z_t, x_t) \leq 2\beta_T \|\Psi_t\|_{V_t^{-1}}. \tag{21}$$

Notice that the upper bound in (21) 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 $x \in \mathcal{X}$, $\theta_{*,i} \cdot \psi(z_t, x) \leq \|\theta_{*,i}\|_2 \|\Psi(z_t, x)\|_2 \leq SL$. Let $(a \wedge b) := \max(a, b)$. Then, we can write

$$\Delta_t x_t \leq \left(2SL \wedge 2\beta_T \|\Psi_t\|_{V_t^{-1}} \right) = 2\beta_T \left(\frac{SL}{\beta_T} \wedge \|\Psi_t\|_{V_t^{-1}} \right) \leq 2\beta_T \left(1 \wedge \|\Psi_t\|_{V_t^{-1}} \right). \tag{22}$$

Taking the sum of both sides of (22) and using the Cauchy-Schwarz inequality, we can write

$$\mathrm{PR}(h_T) \leq \sqrt{8\beta_T^2 T \sum_{t=1}^{T} \left(1 \wedge \|\Psi_t\|^2_{V_t^{-1}} \right)}.$$

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

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

which concludes the proof.

D Proof of Theorem 2

We start off the proof with reformulating ϵ in (2). For any arm $x \in \mathcal{X}$, we can write

$$\epsilon_t(x) = \max\left\{\max_{k\in[m]}\max_{x'\in\mathcal{X}}\left(\mu^k_{t,x'} - \mu^k_{t,x}\right), 0\right\}. \tag{23}$$

Analogously, for all arms $x \in \mathcal{X}$, we have that

$$\hat{\epsilon}_t(x) = \max\left\{\max_{k\in[m]}\max_{x'\in\mathcal{X}}\left(\hat{\mu}^k_{t,x'} - \hat{\mu}^k_{t,x}\right), 0\right\}. \tag{24}$$

Let $x_t \in \mathcal{X}$ be the arm that the algorithm picks at stage t. Then, $\hat{\epsilon}(x_t) = \min_{x\in\mathcal{X}} \hat{\epsilon}(x)$. Then, for all arms $x \in \mathcal{X}$ and objectives $j \in [m]$, there exist an arm $x' \in \mathcal{X}$ and objective $k \in [m]$ such that

$$\hat{\mu}^j_{t,x} - \hat{\mu}^j_{t,x_t} \leq \hat{\mu}^k_{t,x'} - \hat{\mu}^k_{t,x_t}. \tag{25}$$

By Lemma 1, with high probability $1-\delta$, it holds that $(\hat{\mu}_{t,x})_j \geq (\mu_{t,x})_j$. Rearranging (25), we arrive at

$$\mu^j_{t,x} + \hat{\mu}^k_{t,x_t} \leq \hat{\mu}^j_{t,x_t} + \hat{\mu}^k_{t,x'}. \tag{26}$$

For all arms $x \in \mathcal{X}$ and objectives $j \in [m]$, we can write

$$\begin{aligned}
\mu^j_{t,x} - \mu^j_{t,x_t} &\leq \hat{\mu}^j_{t,x_t} + \hat{\mu}^k_{t,x'} - \hat{\mu}^k_{t,x_t} - \mu^j_{t,x_t} \\
&= (\tilde{\theta}_{t,j} - \theta_{*,j}) \cdot \Psi_t + \hat{\mu}^k_{t,x'} - \hat{\mu}^k_{t,x_t} \\
&\leq 2\beta_T \|\Psi_t\|_{V_t^{-1}} + \hat{\mu}^k_{t,x'} - \hat{\mu}^k_{t,x_t} \\
&\leq 2\beta_T \|\Psi_t\|_{V_t^{-1}} + \max_{k\in[m]}\max_{x'\in\mathcal{X}}\left(\hat{\mu}^k_{t,x'} - \hat{\mu}^k_{t,x_t}\right) \\
&\leq 2\beta_T \|\Psi_t\|_{V_t^{-1}} + \hat{\epsilon}(x_t),
\end{aligned} \tag{27}$$

where the first inequality is resulted from (26), and the second inequality follows from the same argument as in the proof of Theorem 1 in Appendix C. Equation (27) holds for all arms x and objectives j, hence

$$0 \leq \epsilon_t(x_t) \leq 2\beta_T \|\Psi_t\|_{V_t^{-1}} + \hat{\epsilon}_t(x_t). \tag{28}$$

Taking the average of both sides of (28), followed by the results in Theorem 1, we arrive at

$$0 \leq \frac{1}{T}\sum_{t=1}^{T} \epsilon_t(x_t) \leq \frac{1}{T}\sum_{t=1}^{T} \hat{\epsilon}_t(x_t) + \mathcal{O}(1/\sqrt{T}). \tag{29}$$

This completes the proof.