We thank all four reviewers for the constructive feedback. Our main objective was the development of theoretical and analytical results. Given the reviewers' interest in the empirical performance aspect of our results, we further discuss the experimental settings below, and provide new and larger-scale experimental results in view of the comments.

**Theoretical significance.** Our main contribution is more on the theoretical/analytic aspect rather than being empirical/algorithmic, serving as a theoretical justification of the integration of MCTS with continuous-armed bandits (CABs) for continuous MDPs. This intuitive idea has been comprehensively evaluated, with demonstration of excellent empirical performance. However, rigorous justification for such an approach is missing. Our primary intention was not to provide more empirical results to support this line of success. Accordingly, we hope that our paper could be (re)-evaluated with that in mind, as it constitutes an original attempt towards theoretically understanding MCTS for continuous MDPs. Our theoretical findings have advocated the use of polynomial bonuses, which was largely ignored in existing empirical solutions; we expect our proof techniques to inspire and guide the design of better empirical methods as well. Additionally, our Thm 2 constitutes an initial investigation of non-stationary bandits in continuous space—an important research area, where our techniques might be useful for future developments.

**Detailed Responses**

1. *Novelty of the general concept* (Rev. 4). Our work is far more than an algorithmic "slight modification of the standard HOOT"; the main contribution/novelty is actually on analysis. First, HOOT is a purely empirical work and establishing convergence guarantee of HOOT is challenging. We are to offer a first theoretical milestone, and hopefully, the abstractions/developments here might pave the way for useful finite-time results under stronger structural assumptions. Additionally, our analytic framework has novel contributions itself. We introduce a new framework to handle non-stationarity in CABs (by translating $L^\infty$ concentration of non-stationarity into a regret convergence rate), which is the first in the literature.

2. *Choice of baselines* (Revs. 1 & 3). The 3 baselines we choose, together with (Kim et al., 2020), are the only solutions we are aware of to MC planning in general continuous spaces. Other methods in *Related Work* either only work in the discrete setting or require specific/different structures of the problem. We did not include (Kim et al., 2020) in experiments because it was posted online only very recently, and there was not enough time for us to implement and evaluate their method. It is indeed a very relevant baseline, and we will include a comparison in the revised version.

3. *Performance improvement seems marginal* (Rev. 1). Our performance gain (over UCT and PUCT) is not marginal on CartPole-IG and Pendulum. For instance, in CartPole-IG, the pole falls roughly after 120 steps for UCT or PUCT, while for `POLY-HOOT` it is steady for at least 2000 steps. The reason that the performance gain "seems" marginal is that the horizon is set to be 150 (see Appendix F), which is large enough to depict the numerical difference between the algorithms, but puts `POLY-HOOT` in an unfavorable situation with "seemingly" insignificant numerical improvement.

4. *Time complexity / Time per decision* (Revs. 1 & 4). It suffices to compare the time complexity (TC) of the bandit algorithms, as the structures of the search tree are essentially the same. TC for discretized-UCT is $O(KT)$, where $T$ is the length of the planning horizon and $K$ is the number of discretized actions. For HOOT, it is $O(T \log T)$ as pointed out by Bubeck et al. (2011). For `POLY-HOOT`, TC reduces to $\min\{O(\bar{H}T), O(T \log T)\}$ due to the existence of the depth limitation $\bar{H}$. For PUCT, a loose upper bound is $O(T^{1+\alpha_d})$, where $\alpha_d$ is the progressive widening coefficient, but in practice the complexity is much lower. We also provide the following "time per decision" results (averaged over 10 runs on a laptop with an Intel Core i5-9300H CPU) on the task of CartPole-IG, with default parameter values.

| Algorithm / $\bar{H}$ value for `POLY-HOOT` | discretized-UCT | PUCT | HOOT | 2 | 4 | 6 | 8 | 10 |
|---|---|---|---|---|---|---|---|---|
| Reward | 69.03 | 70.79 | 77.85 | 42.45 | 48.54 | 63.27 | 77.85 | 77.85 |
| Time per decision (s) | 0.950 | 0.305 | 1.173 | 0.054 | 0.149 | 0.610 | 1.030 | 1.057 |

5. *Maximum depth* (Revs. 2 & 4). The maximum depth $\bar{H}$ is indeed important for convergence guarantees. The effect of $\bar{H}$ should be clear when combined with Lines 19, 24, and 25 in Algorithm 2, where we associate an action with each node on depth $\bar{H}$. Otherwise, HOO will be allowed to explore a new action at each step, which preempts any concentration result (as it is not a martingale). In practice, $\bar{H}$ can be viewed as a hyper-parameter to trade off optimality v.s. computation time. In the above table, we show how $\bar{H}$ influences this trade-off in `POLY-HOOT` on CartPole-IG. The computation time of $\bar{H} = 8$ and 10 are close because most of the time `POLY-HOOT` does not reach the maximum depth when $\bar{H}$ is large enough. The choice of $\bar{H}$ should depend on the horizon $T$, and this will be detailed in the revision.

6. *Variance of numerical evaluations* (Rev. 2). We repeated our evaluations over 40 runs (previously it was 10), and the results turned out to be very similar. We do not include the larger-scale results and variances due to space limitations.

7. *Function approximators* (Rev. 3). Function approximators are indeed very important for MCTS to achieve good empirical performance, especially in continuous spaces. Our goal here is to develop a general analysis for MCTS itself (not restricted to specific designs). As a theory-oriented paper, it was not our primary intention to optimize the empirical performance, although we believe that a combination with function approximators will achieve promising performance.

8. *Discussion on LunarLander* (Rev. 4). `POLY-HOOT` outperforms HOOT more significantly on LunarLander mostly because of the reward structure of the task itself. Sparse and large rewards cause more severe "non-stationarities", and HOOT might get trapped to an area of suboptimal actions in the earlier stages. We will provide the variation trajectories of values inside critical nodes to demonstrate this phenomenon in the final version.