

A THEORETICAL RESULTS

It is useful to understand the long-term behaviour of MAESTRO if each student and teacher agent reaches optimality. In this section, we will formally characterise this equilibrium behaviour, showing MAESTRO achieves a Bayes-Nash equilibrium in a modified game, which we call the $-i$ -knowing game, in which the other player has prior knowledge of the environment and can design their policy accordingly. We will do this by first defining the $-i$ -knowing game, showing that the policies the MAESTRO student learns in equilibria represent Bayes-Nash equilibrium strategies in this game, and then specialising this result to a corollary focused on fully observable games where MAESTRO finds a Nash Equilibrium for all environments in support of the teacher distribution.

We will define the $-i$ -knowing-game, given a UPOSG M , with a parameter distribution $\tilde{\theta}$ as POSG constructed as a modification of the original game where the distribution over parameters is determined by $\tilde{\theta}$, and the agents other than i know the true parameters of the world. This simulates the setting where the co-player is a specialist for the particular environment, who has played the game many times. More formally:

Definition 1. *The $-i$ -knowing-game of an underspecified partially observable stochastic game $M = (N, A = \times_{i \in N} A_i, O = \times_{i \in N} O_i, \Theta, R = \times_{i \in N} R_i, S, I = \times_{i \in N} I_i, T)$ with parameter distribution $\tilde{\theta}$ is defined to be the partially observable stochastic game $K = (N' = N, A'_i = A_i, O'_i = O_i + \{\Theta \text{ if } i \in -i\}, R'_i = R_i, S' = S, I'_i = I_i + \{\theta \text{ if } i \in -i\}, T' = T(\theta))$ where θ is sampled from the distribution $\tilde{\theta}$ on the first time step. That is, a POSG with the same set of players, action space, rewards, and states as the original UPOSG, but with θ sampled once at the beginning of time, fixed into the transition function and given to the agents $-i$ as part of their observation.*

We use $U^K(\pi_i; \pi_{-i}^K; \tilde{\theta})$ to refer to the utility function in the $-i$ -knowing game, which can be written in terms of the utility function of the original UPOSG as

$$U^K(\pi_i; \pi_{-i}^K; \tilde{\theta}) = \mathbb{E}_{\theta \sim \tilde{\theta}} [U(\pi_i; \pi_{-i}^K(\theta), \theta)],$$

where $\pi_{-i}^K(\theta)$ is the policy for players $-i$ in the $-i$ -knowing game conditioned on θ . Given this definition, we can prove the main theorem, that the equilibrium behaviour of MAESTRO represents a Bayes-Nash Equilibrium of this game.

Theorem 1. *In two-player zero-sum settings, the MAESTRO student at equilibrium implements a Bayes-Nash Equilibrium of the $-i$ -knowing game, over a regret-maximising distribution of levels.*

Proof. Let $\pi_i, \tilde{\theta}^M$ be a pair which is in equilibrium in the MAESTRO game. That is:

$$\pi_i \in \arg \max_{\pi_i \in \Pi_i} \left\{ \mathbb{E}_{\theta, \pi_{-i} \sim \tilde{\theta}^M} [U(\pi_i; \pi_{-i}; \theta)] \right\} \quad (3)$$

$$\tilde{\theta}^M \in \arg \max_{\tilde{\theta}^M \in \Delta(\Theta \times \Pi_{-i})} \left\{ \mathbb{E}_{\theta, \pi_{-i} \sim \tilde{\theta}^M} [U(\pi_i^*; \pi_{-i}; \theta) - U(\pi_i; \pi_{-i}; \theta)] \right\} \quad (4)$$

where π_i^* is an optimal policy for player i given π_{-i} and θ , while $\Delta(S)$ denotes the set of distributions over S . Then we can define D^{Regret} to be the marginal distribution over θ from samples $\theta, \pi_{-i} \sim \tilde{\theta}^M$. Define $\pi_{-i}^K(\theta)$ as the marginal distribution over π_{-i} sampled from $\tilde{\theta}^M$ conditioned on θ for θ in the support of $\tilde{\theta}^M$ and π_{-i} a best response to θ and π_i otherwise.

We will show that $(\pi_i; \pi_{-i}^K)$ is a Bayes-Nash equilibrium on $-i$ -knowing game of M with a regret-maximizing distribution over parameters D^{Regret} . We can show both of these by unwrapping and re-wrapping our definitions.

First to show $\pi_i \in \arg \max_{\pi_i \in \Pi_i} \left\{ \mathbb{E}_{\tilde{\theta} \sim D^{Regret}} [U^K(\pi_i; \pi_{-i}^K; \tilde{\theta})] \right\}$:

$$\pi_i \in \arg \max_{\pi_i \in \Pi_i} \left\{ \mathbb{E}_{\tilde{\theta} \sim D^{Regret}} [U^K(\pi_i; \pi_{-i}^K; \tilde{\theta})] \right\} \quad (5)$$

$$\iff \pi_i \in \arg \max_{\pi_i \in \Pi_i} \left\{ \mathbb{E}_{\theta, \pi_{-i} \sim \tilde{\theta}^M} [U(\pi_i; \pi_{-i}^K(\theta); \theta)] \right\} \quad (6)$$

$$\iff \pi_i \in \arg \max_{\pi_i \in \Pi_i} \left\{ \mathbb{E}_{\theta, \pi_{-i} \sim \tilde{\theta}^M} [U(\pi_i; \pi_{-i}; \theta)] \right\} \quad (7)$$

Which is known by the definition of Nash-Equilibrium in the MAESTRO game, in Equation 3.

Similarly, we show that we have $\pi_{-i}^K \in \arg \max_{\pi_{-i}^K \in \Pi_{-i}^K} \mathbb{E}_{\tilde{\theta} \sim D^{Regret}} [U^K(\pi_i; \pi_{-i}^K; \tilde{\theta})]$ by:

$$\pi_{-i}^K \in \arg \max_{\pi_{-i}^K \in \Pi_{-i}^K} \mathbb{E}_{\tilde{\theta} \sim D^{Regret}} [U^K(\pi_i; \pi_{-i}^K; \tilde{\theta})] \quad (8)$$

$$\iff \pi_{-i}^K \in \arg \max_{\pi_{-i}^K \in \Pi_{-i}^K} \mathbb{E}_{\theta, \pi_{-i} \sim \tilde{\theta}^M} [U(\pi_i; \pi_{-i}^K(\theta); \theta)] \quad (9)$$

$$\iff \pi_{-i}^K(\theta) \in \arg \max_{\pi_{-i}^K \in \Pi_{-i}^K} \mathbb{E}_{\theta, \pi_{-i} \sim \tilde{\theta}^M} [U(\pi_i; \pi_{-i}; \theta)]. \quad (10)$$

The final line of which follows from the fact that π_{-i} is a best-response π_i for each θ . More concretely, this can be seen in Equation 4 by noting that θ and π_{-i} conditioned on a specific θ can be independently optimised and holding θ fixed. \square

Using this theorem, we can also prove a natural and intuitive corollary for the case where the environment is fully observable:

Corollary 1. *In fully-observable two-player zero-sum settings, the MAESTRO student at equilibrium implements a Nash Equilibrium in each environment in the support of the environment distribution.*

Proof. From Theorem 1 we have:

$$\pi_i \in \arg \max_{\pi_i \in \Pi_i} \mathbb{E}_{\tilde{\theta} \sim D^{Regret}} [U^K(\pi_i; \pi_{-i}^K; \tilde{\theta})]$$

However, since the world is fully observable, the policy π_{-i}^K can be made independent of the additional observation θ in the $-i$ -knowing game since that can be inferred from the information already in the agent’s observations. As such, π_{-i}^K can be interpreted as a policy in the original game, giving:

$$\pi_i \in \arg \max_{\pi_i \in \Pi_i} \mathbb{E}_{\theta \sim D^{Regret}} [U(\pi_i; \pi_{-i}^K(\theta); \theta)]$$

Moreover, since the environment is fully observable, π_i can condition on θ , so for it to be optimal for the distribution, it must be optimal for each level in the support of the distribution. Giving, for each θ be in the support of $\tilde{\theta}^M$:

$$\pi_i \in \arg \max_{\pi_i \in \Pi_i} \{U(\pi_i; \pi_{-i}^K(\theta); \theta)\}$$

Showing that π is a best-response to $\pi_{-i}^K(\theta)$ on θ . The same arguments can be followed to show that $\pi_{-i}^K(\theta)$ is a best response to π on θ . Since each policy is a best response to the other, they are in a Nash Equilibrium as desired. \square

Thus, if MAESTRO reaches an equilibrium in a fully observable two-player zero-sum setting it behaves as expected by achieving a Nash Equilibrium in every environment in support of the curriculum distribution $\tilde{\theta}^M$.

B ABLATION STUDY

We perform an ablation study to evaluate the effectiveness of co-player selection in MAESTRO from a population based on their per-environment regret scores, as described in Equation (1). We consider different methods for selecting a co-player in MAESTRO (line 6 in Algorithm 1). MAESTRO-R samples the co-player uniformly at random, whilst MAESTRO-P uses PFSP’s win rate heuristic for opponent prioritisation. Figure 8 illustrates that MAESTRO outperforms both variants in terms of sample-efficiency and robustness.

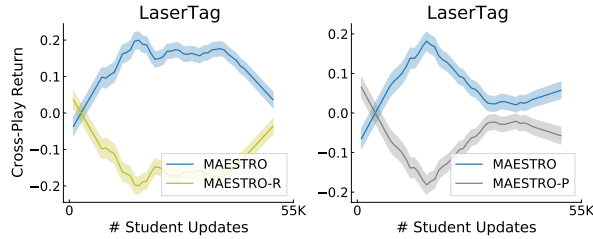


Figure 8: **Ablation Study Results.** Comparing MAESTRO against two variants: MAESTRO-R and MAESTRO-P. Plots show the mean and standard error across 5 training seeds.

C ENVIRONMENT DETAILS

This section describes the environment-specific details used in our experiments. For both LaserTag and MultiCarRacing, we outline the process of environment generation, present held-out evaluation environments, as well as other relevant information.

C.1 LASERTAG

LaserTag is a two-player zero-sum grid-based game, where two agents aim to tag each other with a light beam under partial observability. It is inspired by prior singleton variation used in (Lanctot et al., 2017; Leibo et al., 2017) and developed using the Griddly sandbox framework (Bamford et al., 2020; 2022). LaserTag challenges the agent to master various sophisticated behaviour, such as chasing opponents, hiding behind walls, keeping clear of long corridors, maze solving, etc. Each agent can only observe a 5×5 area of the grid in front of it. The action space includes the following 5 actions: turn right, turn left, move forward, shoot, and no-op. Upon tagging an opponent, an agent receives a reward of 1, while the opponent receives a -1 penalty, after which the episode is restarted. If the episode terminates while the two agents are alive, neither of the agents receives any reward.

All environment variations (or levels) that are used to train agents are procedurally generated by an environment generator. Firstly, the generator samples the size of the grid (from 5×5 to 15×15) and the percentage of the walls it is in (from 0% to 50%) uniformly at random. Then the generator samples random locations for the walls, followed by the locations and directions of the two agents. Figure 2 illustrates some levels sampled from the level generator. Note that the generator can generate levels where agents are unreachable.

Upon training the agents on randomly generated levels, we assess their robustness on previously unseen human-designed levels shown in Figure 9 against previously unseen agents. For this environment, we recognise the agent with a higher episodic return as the winner of that episode.

C.2 MULTICARRACING

MultiCarRacing is a continuous control problem with dense rewards and pixel-based observations (Schwartz et al., 2021). Each track consists of n tiles, with cars receiving a reward of $1000/n$ or $500/n$, depending on if they reach the tile first or second respectively. An additional penalty of -0.1 is applied at every timestep. Episodes finish when all tiles have been driven over by at least one car. If a car drives out of bounds of the map (rectangle area encompassing the track), the car "dies" and the episode is terminated. Each agent receives a $96 \times 96 \times 3$ image as observation at each timestep. The action space consists of 3 simultaneous moves that change the gas, brakes, and steering direction of the car.

All tracks used to train student agents are procedurally generated by an environment generator, which was built on top of the original MultiCarRacing environment (Schwartz et al., 2021). Each track consists of a closed loop around which the agents must drive a full lap. In order to increase the expressiveness of the original MultiCarRacing, we reparameterized the tracks using Bézier curves. In our experiments, each track consists of a Bézier curve based on 12 randomly sampled control points within a fixed radius of $B/2$ of the centre O of the playfield with $B \times B$ size.

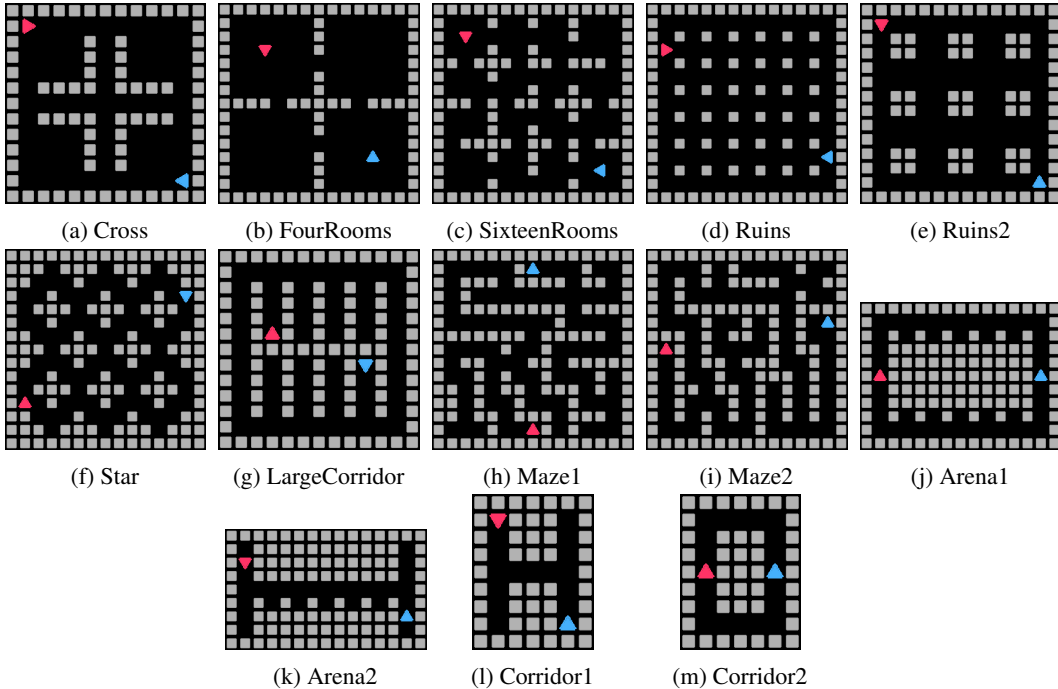


Figure 9: Evaluation environments for LaserTag.

For training, additional reward shaping was introduced similar to (Ma, 2019): an additional reward penalty of -0.1 for driving on the grass, a penalty of -0.5 for driving backwards, as well as an early termination if cars spent too much time on grass. These are all used to help terminate less informative episodes. We utilise a memory-less agent with a frame stacking = 4 and with sticky actions = 8. After training the agents on randomly generated tracks, we assess their robustness on previously unseen 20 real-world Formula One (F1) tracks designed to challenge professional racecar drivers proposed by (Jiang et al., 2021a) and shown in Figure 10.

Given the poor performance of random opponents on the MultiCarRacing domain, agents are added to co-player populations in MAESTRO as well as FSP- and PFSP-based baselines only after 400 PPO updates. All baselines are trained using SP until that point.

D IMPLEMENTATION DETAILS

In this section, we detail the agent architectures, hyperparameter choices, and evaluation procedures used in our experiments discussed in Section 4. We use PPO to train the student agent in all experiments. Table 2 summarises our final hyperparameter choices for all methods.

All experiments are performed on an internal cluster. Each job (representing a seed) is performed with a single Tesla V100 GPU and 10 CPUs. For each method, we train 10 LaserTag agents for approximately 7 days and 5 MultiCarRacing agents for approximately 15 days.

D.1 LASERTAG

Agent Architecture: The student policy architecture is adapted from (Dennis et al., 2020; Jiang et al., 2021a). Our model encodes the partial grid observation using a convolution layer (3×3 kernel, stride length 1, 16 filters) followed by a ReLU activation layer over the flattened convolution outputs. This is then passed through an LSTM with hidden dimension 256, followed by two fully-connected layers, each with a hidden dimension of size 32 with ReLU activations, to produce the action logits over the 5 possible actions. The model does not receive the agent’s direction as input.

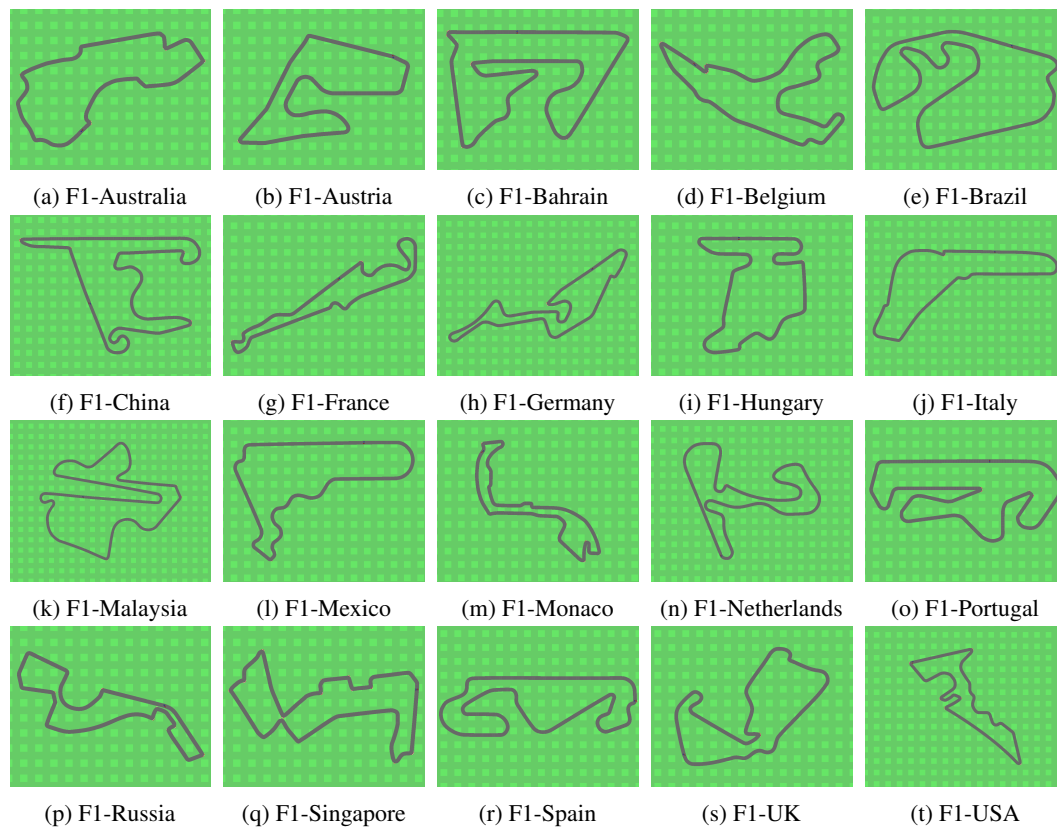


Figure 10: Evaluation Formula 1 tracks for MultiCarRacing.

Evaluation Procedure: For each pair of methods in LaserTag, we evaluate cross-play performance across all 5 training seeds of each method (25 possible pair combinations) over 4 episodes on 11 test environments, resulting in a total of 1100 evaluation episodes per method pair.

Choice of Hyperparameters: Many of our hyperparameters are inherited from previous works such as (Dennis et al., 2020; Jiang et al., 2021b;a; Parker-Holder et al., 2022) with some small changes. We selected the best performing settings based on the average return on the unseen validation levels against previously unseen opponents on at least 5 seeds.

We conducted a coarse grid search over student learning rate in $\{5 * 10^{-4}, 10^{-4}, 5 * 10^{-5}, 10^{-5}\}$, number of minibatches per epoch in $\{1, 2, 4\}$, entropy coefficients in $\{0, 10^{-3}, 5 * 10^{-3}\}$, and number of epochs in $\{2, 5, 10\}$. For agent storage, we tested adding a copy of the student agent in the storage after every $\{2000, 4000, 6000, 8000\}$ student update. For PFSP, we compute the win rate between agents in the last 128 episodes. We further conducted a grid search over the entropy parameter of f_{hard} in $\{1.5, 2, 3\}$ and a smoothing constant which adds a small value to each probability in PFSP with $\{0.1, 0.2\}$ values so that, at all previous checkpointed agents have a nonzero probability to be replayed again.³ For the parameters of PLR, we conducted a grid search over level replay rate p in $\{0.5, 0.9\}$, buffer size in $\{4000, 8000, 12000\}$, staleness coefficient ρ in $\{0.3, 0.7\}$, as well as the level replay score functions in $\{\text{MaxMC}, \text{PVL}\}$ (see Appendix D.4 for information on score functions in PLR). For MAESTRO, we evaluated the co-player exploration coefficients in $\{0.05, 0.1\}$, and per-agent environment buffer sizes in $\{500, 1000\}$.

D.2 MULTICARRACING

Agent Architecture: The student policy architecture is based on the PPO implementation in (Ma, 2019). The model utilises an image embedding module consisting of a stack of 2D convolutions with square kernels of sizes 2, 2, 2, 2, 3, 3, channel outputs of 8, 16, 32, 64, 128, 256, and stride lengths of 2, 2, 2, 2, 1, 1 respectively, resulting in an embedding of size 256. This is then passed through a fully-connected layer with a hidden size of 100, followed by a ReLU nonlinearity. Then, the output is fed through two separate fully-connected layers, each with a hidden size of 100 and an output dimension equal to the action dimension, followed by softplus activations. We then add 1 to each component of these two output vectors, which serve as the α and β parameters respectively for the Beta distributions used to sample each action dimension. We normalize rewards by dividing rewards by the running standard deviation of returns so far encountered during the training.

Evaluation Procedure: For each pair of methods in MultiCarRacing, we evaluate cross-play performance between 5 training seeds of each method (random assignments between 5 pairs) over 2 episodes on 5 test environments, resulting in a total of 50 evaluation episodes per method pair.

Choice of Hyperparameters: Many of our hyperparameters are inherited from (Jiang et al., 2021a) with some small changes. We conducted a limited grid search over student learning rate in $\{10^{-4}, 3 * 10^{-4}\}$, number of actors in $\{16, 32\}$, PPO rollout length in $\{125, 256\}$. For agent storage, we tested adding a copy of the student agent in the storage after every $\{200, 400\}$ student update. For PFSP, we compute the win rate between agents in the last 128 episodes, while recognising the agent with a higher episodic return as the winner. For the parameters of PLR, we conducted a grid search over level buffer size in $\{4000, 6000, 8000\}$, staleness coefficient ρ in $\{0.3, 0.7\}$, as well as the level replay prioritisation in $\{\text{rank}, \text{proportional}\}$ (Jiang et al., 2021a). For MAESTRO, we evaluated the co-player exploration coefficients in $\{0.05, 0.1\}$, and per-agent environment buffer sizes in $\{500, 1000\}$.

D.3 ALL HYPERPARAMETERS

Table 2 summarises our final hyperparameter choices for all methods.

D.4 PLR STRATEGIES

Jiang et al. (2021a) proposes the following two score functions to approximate regret in PLR.

³Otherwise, if the student agent wins all the episodes in their first encounter against opponent B, B will have 0 probability of being selected again.

Table 2: Hyperparameters used for training each method in the LaserTag and MultiCarRacing environments.

Parameter	LaserTag	MultiCarRacing
<i>PPO</i>		
γ	0.995	0.99
λ_{GAE}	0.95	0.9
PPO rollout length	256	125
PPO epochs	5	8
PPO mini-batches per epoch	4	4
PPO clip range	0.2	0.2
PPO number of workers	32	32
Adam learning rate	1e-4	1e-4
Adam ϵ	1e-5	1e-5
PPO max gradient norm	0.5	0.5
PPO value clipping	yes	no
Return normalization	no	yes
Value loss coefficient	0.5	0.5
Student entropy coefficient	0.0	0.0
<i>PLR</i>		
Replay rate, p	0.5	0.5
Buffer size, K	4000	8000
Scoring function	MaxMC	PVL
Prioritization	rank	rank
Temperature, β	0.3	1.0
Staleness coefficient, ρ	0.3	0.7
<i>FSP</i>		
Agent checkpoint interval	8000	400
<i>PFSP</i>		
f_{hard} entropy coef	2	2
Win rate episodic memory	128	128
<i>MAESTRO</i>		
λ coef	0.1	0.1
Buffer size for \mathfrak{B} members	1000	1000

Positive Value Loss (PVL) estimates the regret by computing the difference between maximum achieved return and predicted return on an episodic basis. When GAE (Schulman et al., 2016) is used to estimate bootstrapped value targets, this loss takes the form of $\frac{1}{T} \sum_{t=0}^T \max \left(\sum_{k=t}^T (\gamma \lambda)^{k-t} \delta_k, 0 \right)$, where λ and γ are the GAE and MDP discount factors respectively, and δ_t , the TD-error at timestep t .

Maximum Monte Carlo (MaxMC) mitigates some of the bias of the PVL by replacing the value target with the highest empirical return observed on the given environment variation throughout training. MaxMC ensures that the regret estimate does not depend on the agent’s current policy. It takes the form of $(1/T) \sum_{t=0}^T R_{\text{max}} - V(s_t)$.

E FULL RESULTS

E.1 MAESTRO VERSUS SPECIALISTS

Figures 11 and 12 show the cross-play performances between MAESTRO and specialist agents trained directly on the target environments for LaserTag and MultiCarRacing, respectively.

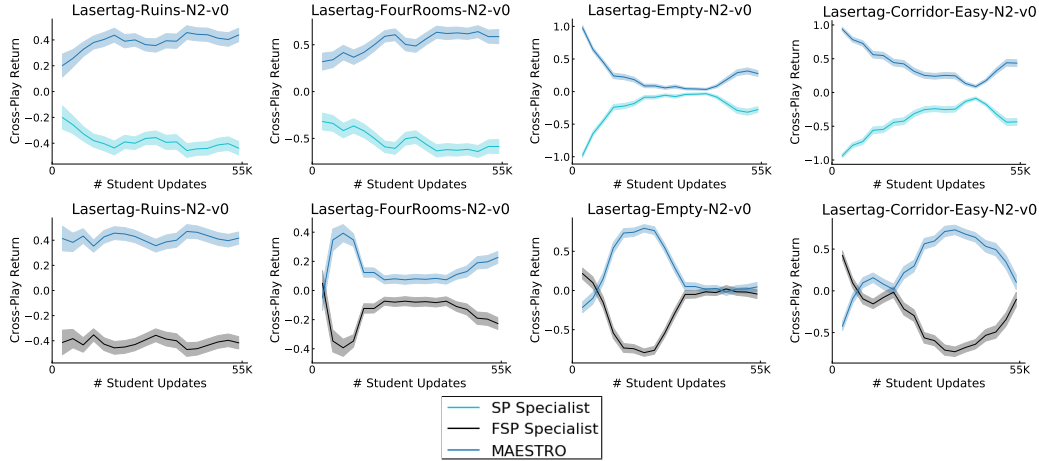


Figure 11: Cross-play between MAESTRO and specialist agents trained directly on the target environment in LaserTag.

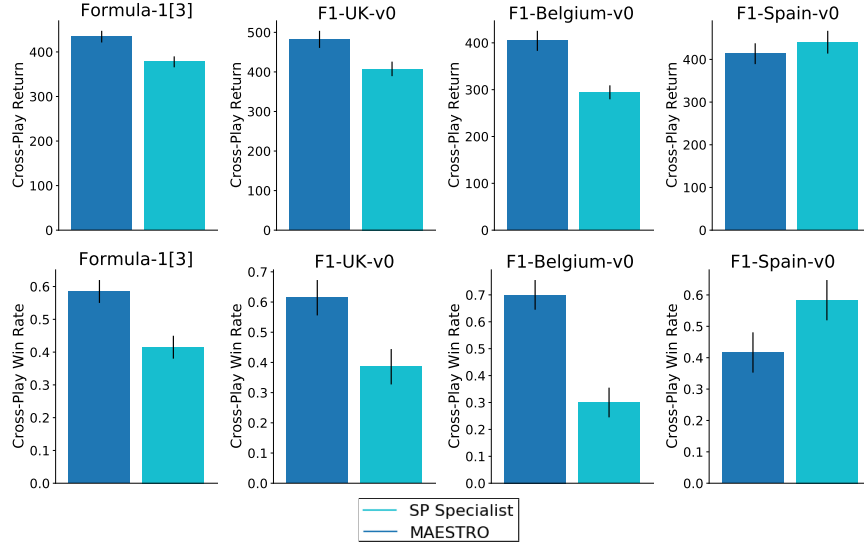


Figure 12: Cross-play between MAESTRO and specialist agents trained directly on the target environment in MultiCarRacing.

E.2 CROSS-PLAY RESULTS

E.2.1 LASERTAG CROSS-PLAY

Figure 13 shows the mean return of MAESTRO against each of the baselines throughout training. Figure 14 illustrates the minimum expected return between MAESTRO and other baselines throughout training on each held-out evaluation environment in LaserTag domain. 15 and 16 show the minimum expected return and round-robin returns between MAESTRO and other baselines after the training.

E.2.2 MULTICARRACING CROSS-PLAY

Figure 14 illustrates the min expected return between MAESTRO and other baselines on each track of the Formula 1 benchmark (Jiang et al., 2021a). Figures 18 and 19 show the win rates and returns in a round robin tournament between MAESTRO after the training on Formula 1 benchmark.

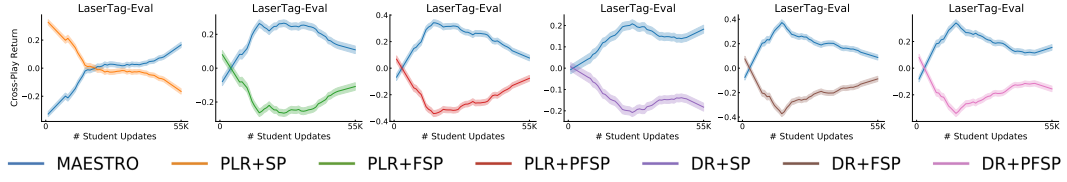


Figure 13: Mean return of MAESTRO versus each of the baselines throughout training on LaserTag evaluation environments.

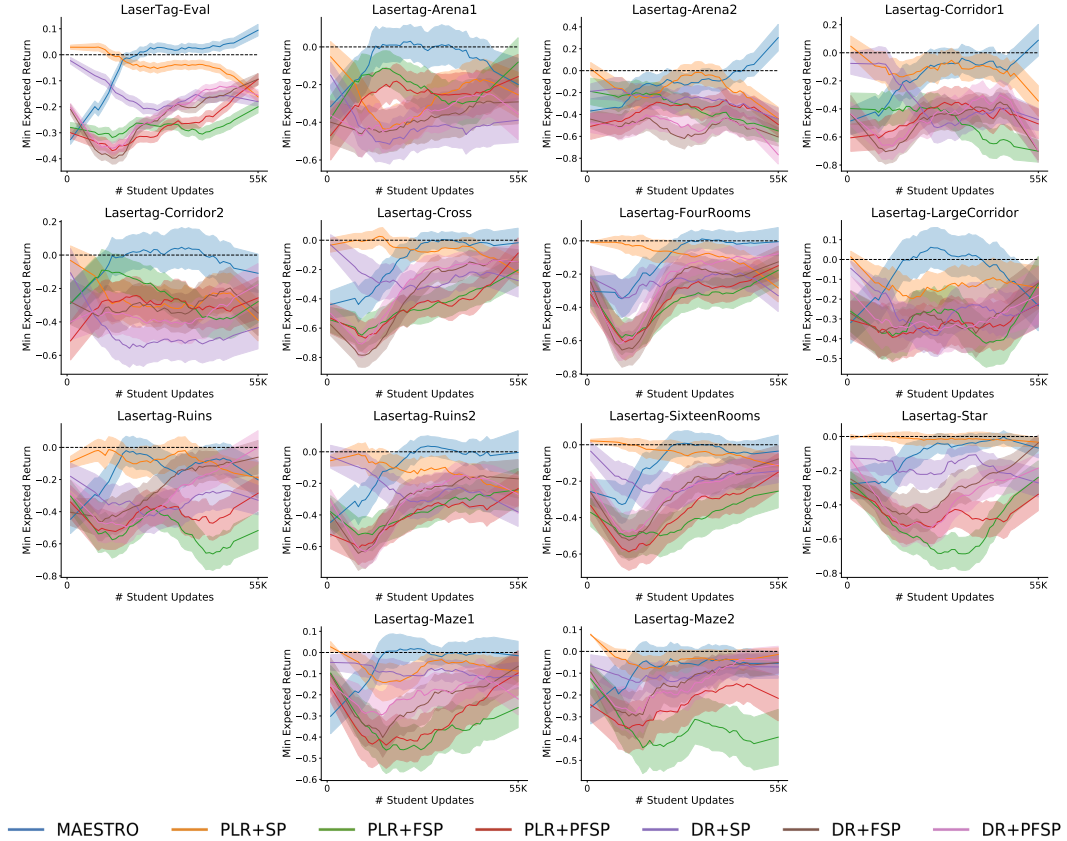


Figure 14: Minimum expected return in cross-play between MAESTRO and 6 baselines on all LaserTag evaluation environments throughout training.

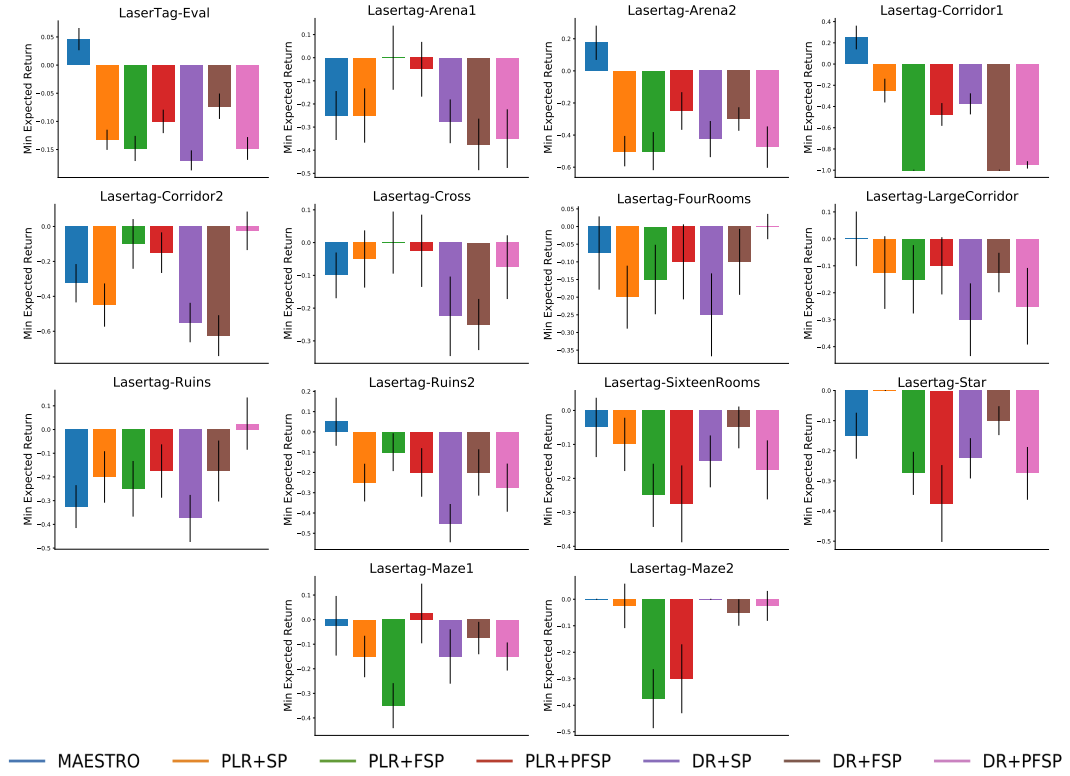


Figure 15: Minimum expected returns in cross-play between MAESTRO and 6 baselines on all LaserTag evaluation environments after the training.

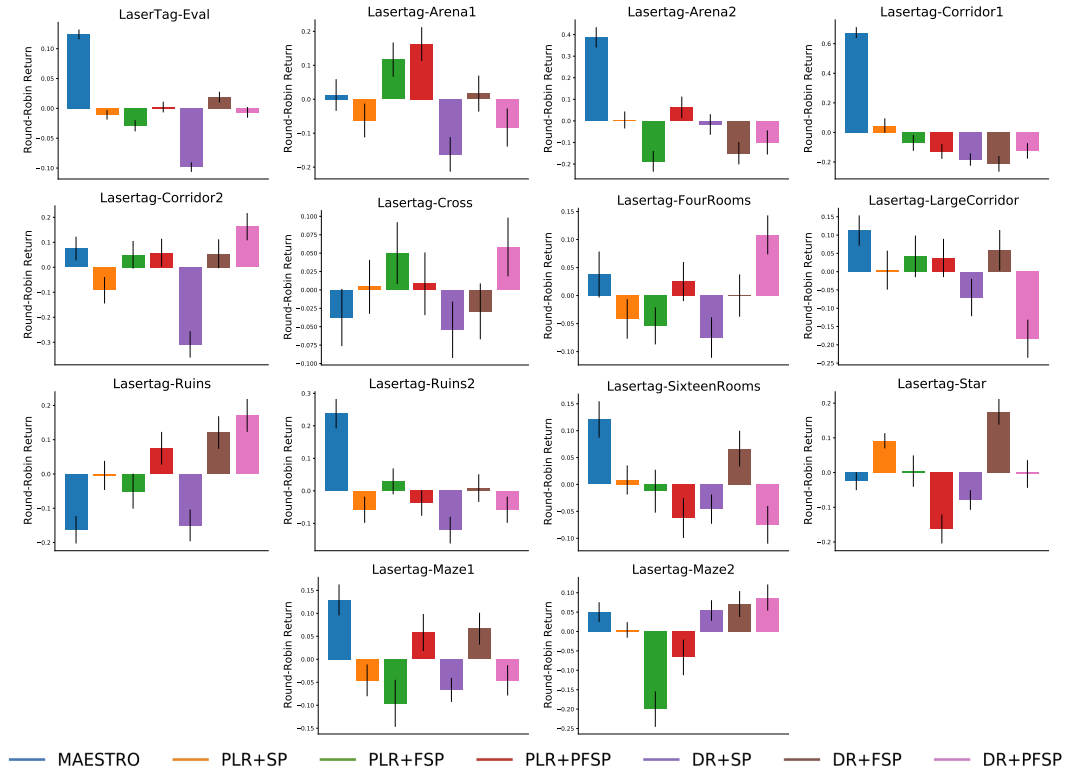


Figure 16: Returns in a round-robin tournament between MAESTRO and 6 baselines on all LaserTag evaluation environments.

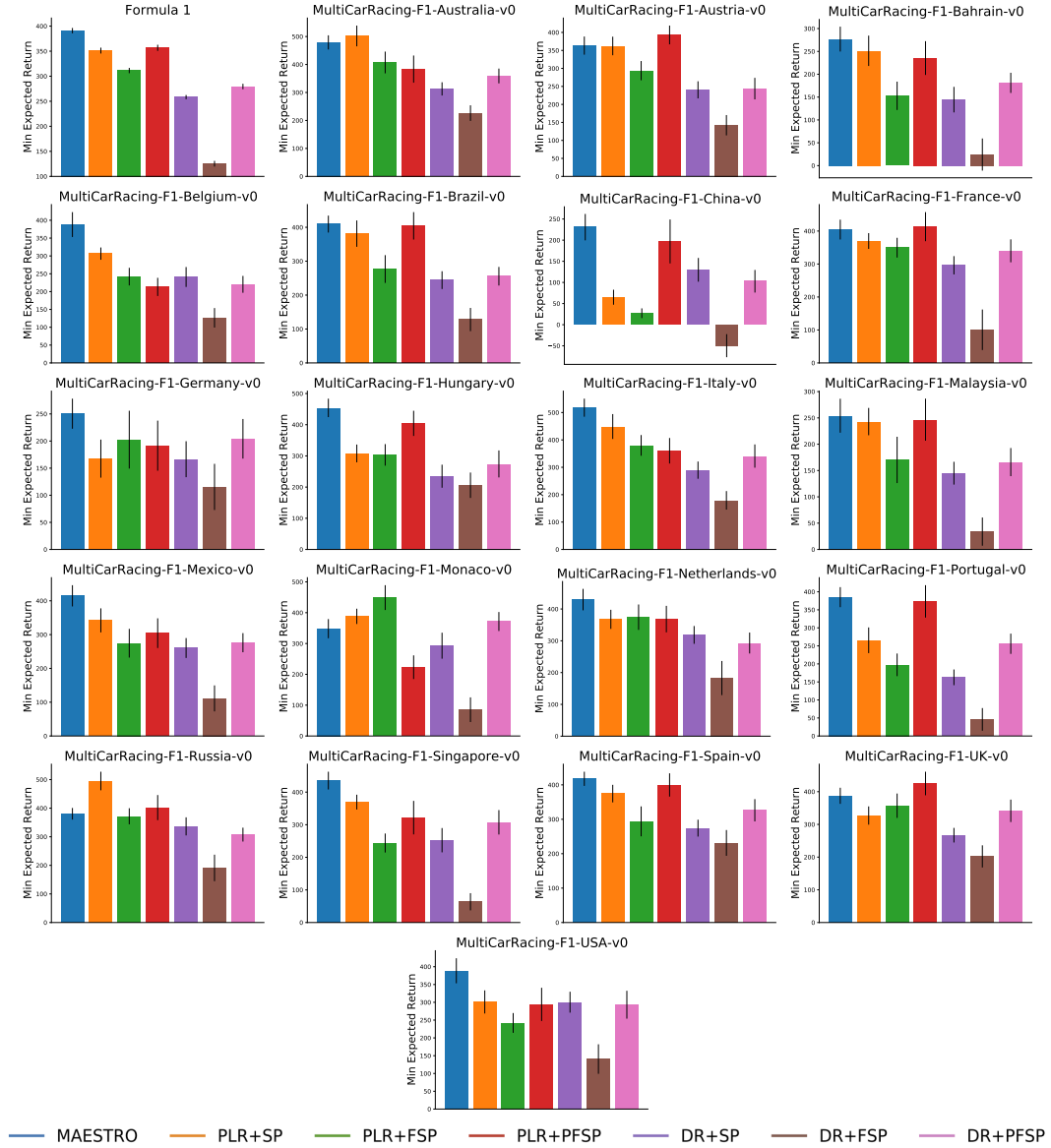


Figure 17: Minimum expected return between MAESTRO and 6 baselines on all Formula 1 tracks.

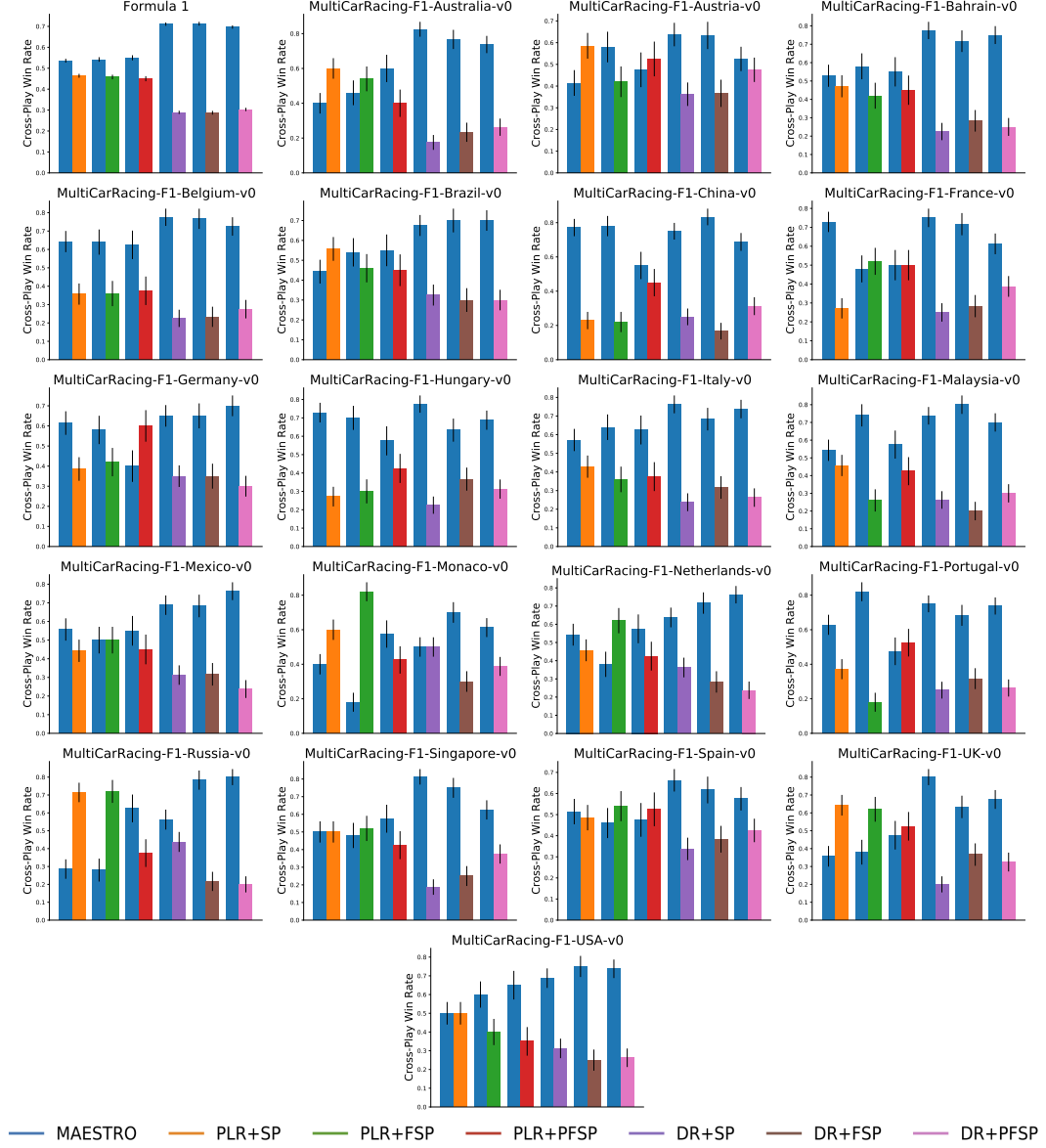


Figure 18: Win rates in cross-play between MAESTRO vs each of the 6 baselines on all Formula 1 tracks.

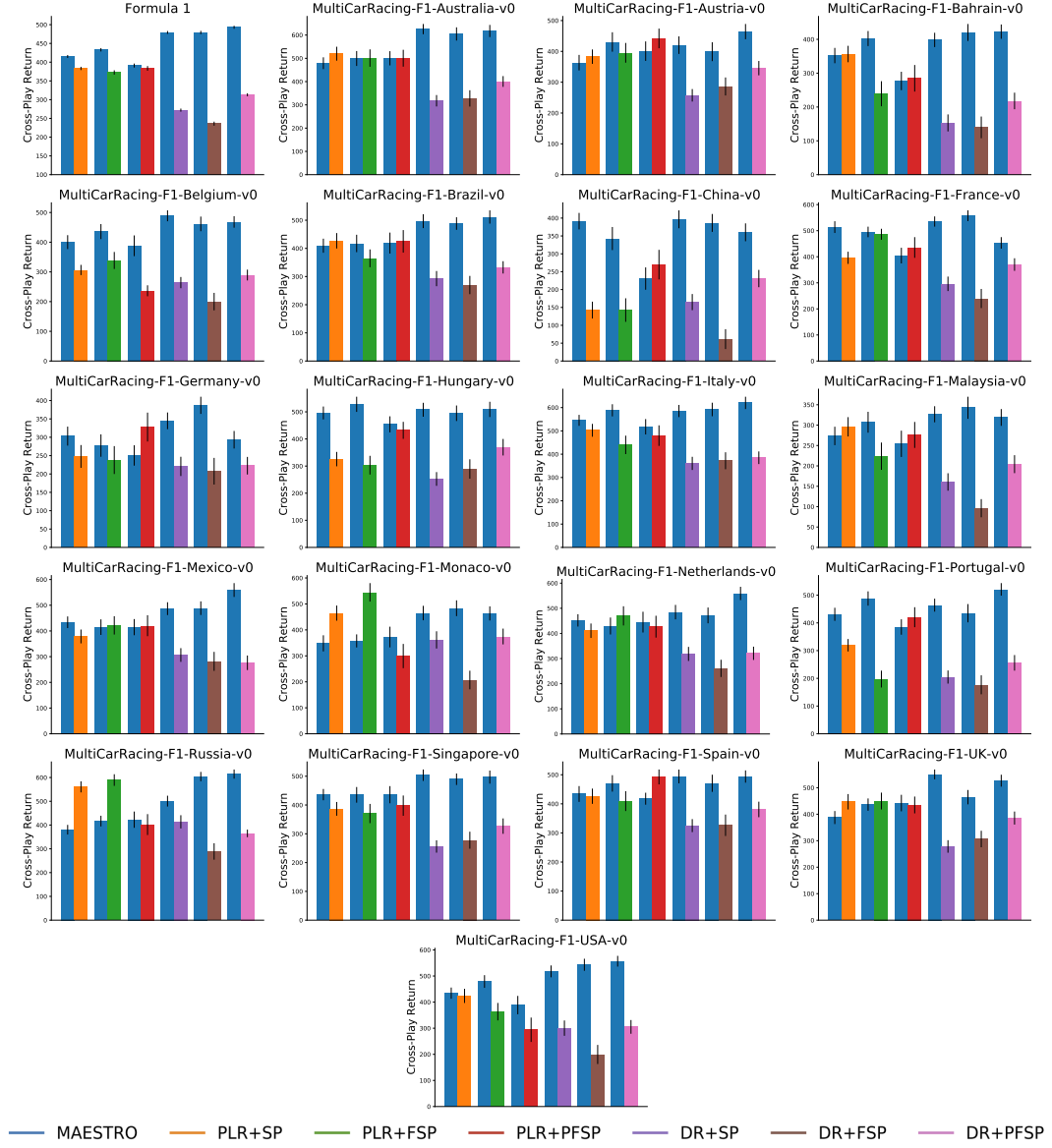


Figure 19: Returns in cross-play between MAESTRO vs each of the 6 baselines on all Formula 1 tracks.