A. Offline population synthesis

Here we provide extra details for Section 2.

Horizontal steps Horizontal steps occur as follows. Two random particles are sampled uniformly at random from adjacent temeprature levels. This forms a proposal for the swap, which is then accepted via standard MH acceptance conditions. Because the rest of the particles remain as-is, the acceptance condition reduces to a particualrly simple form (*cf.* Algorithm 3).

Algorithm 3 HORIZONTAL SWAP

Sample $i \sim \text{Uniform}(1, 2, \dots, L-1)$.
Sample $j, k \overset{\text{i.i.d.}}{\sim} \text{Uniform}(1, 2, \dots, D)$.
Sample $p \sim \text{Uniform}([0, 1])$
Let $a = \min\left(1, e^{f(x^{i,j}, \theta^{i,j}) - f(x^{i+1,k}, \theta^{i+1,k})}\right)$
if $p < a^{\beta_i - \beta_{i+1}}$
  swap configurations $(x^{i,j}, \theta^{i,j})$ and $(x^{i+1,k}, \theta^{i+1,k})$

We ran our experiments on a server with 88 Intel Xeon cores @ 2.20 GHz. Each run of 100 iterations for a given hyperparameter setting α took 20 hours.

B. Online robust planning

Here we provide extra details for Section 3.

B.1. Solving problem (5)

We can rewrite the constraint $D_\phi(q\|\mathbf{1}/N_w) \le \rho$ as $\|q - \mathbf{1}/N_w\|^2 \le \rho/N_w$. Then, the partial Lagrangian can be written as

$$\mathcal{L}(q, \lambda) = \sum_i q_i c_i(t) - \frac{\lambda}{2}\left(\|q - \mathbf{1}/N_w\|^2 - \rho/N_w\right).$$

By inspection of the right-hand side, we see that, for a given λ, finding $v(\lambda) = \sup_{q \in \Delta} \mathcal{L}(q, \lambda)$ is equivalent to a Euclidean-norm projection of the vector $\mathbf{1}/N_w + c(t)/\lambda$ onto the probability simplex Δ. This latter problem is directly amenable to the methods of Duchi et al. (2008).

B.2. Proof of Proposition 1

We redefine notation to suppress dependence of the cost C on other variables and just make explicit the dependence on the random index J. Namely, we let $C : \mathcal{J} \to [-1, 1]$ be a function of the random index J. We consider the convergence of

$$\sup_{Q \in \mathcal{P}_{N_w}} \mathbb{E}_Q[C(J)] \text{ to } \sup_{Q \in \mathcal{P}} \mathbb{E}_Q[C(J)].$$

To ease notation, we hide dependence on J and for a sample $J_1, \dots, J_{N_w}$ of random vectors J_k, we denote $C_k := C(J_k)$ for shorthand, so that the C_k are bounded independent random variables. Our proof technique is similar in style to that of Sinha & Duchi (2016). We provide proofs for technical lemmas that follow in support of Proposition 1 that are shorter and more suitable for our setting (in particular Lemmas 1 and 3).

Treating $C = (C_1, \dots, C_{N_w})$ as a vector, the mapping $C \mapsto \sup_{Q \in \mathcal{P}_{N_w}} \mathbb{E}_Q[C]$ is a $\sqrt{\rho + 1}/\sqrt{N_w}$-Lipschitz convex function of independent bounded random variables. Indeed, letting $q \in \mathbb{R}_+^{N_w}$ be the empirical probability mass function associated with $Q \in \mathcal{P}_{N_w}$, we have $\frac{1}{N_w}\sum_{i=1}^{N_w}(N_w q_i)^2 \le \rho + 1$ or $\|q\|_2 \le \sqrt{(1+\rho)/N_w}$. Using Samson's sub-Gaussian concentration inequality (Samson, 2000) for Lipschitz convex functions of bounded random variables, we have with probability at least $1 - \delta$ that

$$\sup_{Q \in \mathcal{P}_{N_w}} \mathbb{E}_Q[C] \in \mathbb{E}\left[\sup_{Q \in \mathcal{P}_{N_w}} \mathbb{E}_Q[C]\right] \pm 2\sqrt{2}\sqrt{\frac{(1+\rho)\log\frac{2}{\delta}}{N_w}}. \tag{8}$$

By the containment (8), we only need to consider convergence of

$$\mathbb{E}\left[\sup_{Q \in \mathcal{P}_{N_w}} \mathbb{E}_Q[C]\right] \text{ to } \sup_{Q \in \mathcal{P}} \mathbb{E}_Q[C],$$

which we do with the following lemma.

Lemma 1 (Sinha & Duchi (2016))**.** *Let $Z = (Z_1, \dots, Z_{N_w})$ be a random vector of independent random variables $Z_i \overset{\text{i.i.d.}}{\sim} P_0$, where $|Z_i| \le M$ with probability 1. Let $C_\rho = \frac{2(\rho+1)}{\sqrt{1+\rho}-1}$. Then*

$$\mathbb{E}\left[\sup_{Q \in \mathcal{P}_{N_w}} \mathbb{E}_Q[Z]\right] \ge \sup_{Q \in \mathcal{P}} \mathbb{E}_Q[Z] - 4C_\rho M\sqrt{\frac{\log(2N_w)}{N_w}}$$

and

$$\mathbb{E}\left[\sup_{Q \in \mathcal{P}_{N_w}} \mathbb{E}_Q[Z]\right] \le \sup_{Q \in \mathcal{P}} \mathbb{E}_Q[Z].$$

See Appendix B.3 for the proof.

Combining Lemma 1 with containment (8) gives the result.

B.3. Proof of Lemma 1

Before beginning the proof, we first state a technical lemma.

Lemma 2 (Ben-Tal et al. (2013))**.** *Let ϕ be any closed convex function with domain $\operatorname{dom}\phi \subset [0, \infty)$, and let $\phi^*(s) = \sup_{t \ge 0}\{ts - \phi(t)\}$ be its conjugate. Then for any distribution P and any function $g : \mathcal{W} \to \mathbb{R}$ we have*

$$\sup_{Q : D_\phi(Q\|P) \le \rho} \int g(w) dQ(w) = \inf_{\lambda \ge 0, \eta}\left\{\lambda \int \phi^*\left(\frac{g(w) - \eta}{\lambda}\right) dP(w) + \rho\lambda + \eta\right\}.$$

See Appendix B.4 for the proof.

We prove the result for general ϕ-divergences $\phi(t) = t^k - 1$, $k \ge 2$. To simplify algebra, we work with a scaled

version of the ϕ-divergence: $\phi(t) = \frac{1}{k}(t^k - 1)$, so the scaled population and empirical constraint sets we consider are defined by

$$\mathcal{P} = \left\{Q : D_\phi\left(Q \| P_0\right) \le \frac{\rho}{k}\right\} \text{ and } \mathcal{P}_{N_w} := \left\{q : D_\phi\left(q \| \mathbf{1}/N_w\right) \le \frac{\rho}{k}\right\}.$$

Then by Lemma 2, we obtain

$$\begin{aligned}
\mathbb{E}\left[\sup_{Q \in \mathcal{P}_{N_w}} \mathbb{E}_Q[Z]\right] &= \mathbb{E}_{P_0}\left[\inf_{\lambda \ge 0, \eta} \frac{1}{N_w}\sum_{i=1}^{N_w} \lambda\phi^*\left(\frac{Z_i - \eta}{\lambda}\right) + \eta + \frac{\rho}{k}\lambda\right] \\
&\le \inf_{\lambda \ge 0, \eta} \mathbb{E}_{P_0}\left[\frac{1}{N_w}\sum_{i=1}^{N_w} \lambda\phi^*\left(\frac{Z_i - \eta}{\lambda}\right) + \eta + \frac{\rho}{k}\lambda\right] \\
&= \inf_{\lambda \ge 0, \eta}\left\{\mathbb{E}_{P_0}\left[\lambda\phi^*\left(\frac{Z - \eta}{\lambda}\right)\right] + \frac{\rho}{k}\lambda + \eta\right\} \\
&= \sup_{Q \in \mathcal{P}} \mathbb{E}_Q[Z].
\end{aligned}$$

This proves the upper bound in Lemma 1.

Now we focus on the lower bound. For the function $\phi(t) = \frac{1}{k}(t^k - 1)$, we have $\phi^*(s) = \frac{1}{k^*}[s]_+^{k^*} + \frac{1}{k}$, where $1/k^* + 1/k = 1$, so that the duality result in Lemma 2 gives

$$\sup_{Q \in \mathcal{P}_{N_w}} \mathbb{E}_Q[Z] = \inf_\eta \left\{(1+\rho)^{1/k}\left(\frac{1}{N_w}\sum_{i=1}^{N_w}[Z_i - \eta]_+^{k^*}\right)^{\frac{1}{k^*}} + \eta\right\}.$$

Because $|Z_i| \le M$ for all i, we claim that any η minimizing the preceding expression must satisfy

$$\eta \in \left[-\frac{1 + (1+\rho)^{\frac{1}{k^*}}}{(1+\rho)^{\frac{1}{k^*}} - 1}, 1\right] \cdot M. \tag{9}$$

For convenience, we first define the shorthand

$$S_{N_w}(\eta) := (1+\rho)^{1/k}\left(\frac{1}{N_w}\sum_{i=1}^{N_w}[Z_i - \eta]_+^{k^*}\right)^{\frac{1}{k^*}} + \eta.$$

Then it is clear that $\eta \le M$, because otherwise we would have $S_{N_w}(\eta) > M \ge \inf_\eta S_{N_w}(\eta)$. Let the lower bound be of the form $\eta = -cM$ for some $c > 1$. Taking derivatives of the objective $S_{N_w}(\eta)$ with respect to η, we have

$$\begin{aligned}
S'_{N_w}(\eta) &= 1 - (1+\rho)^{1/k}\frac{\frac{1}{N_w}\sum_{i=1}^{N_w}[Z_i - \eta]_+^{k^*-1}}{\left(\frac{1}{N_w}\sum_{i=1}^{N_w}[Z_i - \eta]_+^{k^*}\right)^{1-\frac{1}{k^*}}} \\
&\le 1 - (1+\rho)^{1/k}\left(\frac{(c-1)M}{(c+1)M}\right)^{k^*-1} \\
&= 1 - (1+\rho)^{1/k}\left(\frac{c-1}{c+1}\right)^{k^*-1}.
\end{aligned}$$

For any $c > c_{\rho,k} := \frac{(1+\rho)^{\frac{1}{k^*}}+1}{(1+\rho)^{\frac{1}{k^*}}-1}$, the preceding display is negative, so we must have $\eta \ge -c_{\rho,k}M$. For the remainder of the proof, we thus define the interval

$$U := \left[-Mc_{\rho,k}, M\right], \quad c_{\rho,k} = \frac{(1+\rho)^{\frac{1}{k^*}}+1}{(1+\rho)^{\frac{1}{k^*}}-1},$$

and we assume w.l.o.g. that $\eta \in U$.

Again applying the duality result of Lemma 2, we have that

$$\begin{aligned}
\mathbb{E}\left[\sup_{Q \in \mathcal{P}_{N_w}} \mathbb{E}_Q[Z]\right] &= \mathbb{E}\left[\inf_{\eta \in U} S_{N_w}(\eta\right] \\
&= \mathbb{E}\left[\inf_{\eta \in U}\{S_{N_w}(\eta) - \mathbb{E}[S_{N_w}(\eta)] + \mathbb{E}[S_{N_w}(\eta)]\}\right] \\
&\ge \inf_{\eta \in U}\mathbb{E}[S_{N_w}(\eta)] \\
&\quad - \mathbb{E}\left[\sup_{\eta \in U}|S_{N_w}(\eta) - \mathbb{E}[S_{N_w}(\eta)]|\right]. \quad (10)
\end{aligned}$$

To bound the first term in expression (10), we use the following lemma.

Lemma 3 (Sinha & Duchi (2016)). *Let $Z \ge 0, Z \not\equiv 0$ be a random variable with finite $2p$-th moment for $1 \le p \le 2$. Then we have the following inequality:*

$$\mathbb{E}\left[\left(\frac{1}{n}\sum_{i=1}^n Z_i^p\right)^{\frac{1}{p}}\right] \ge \|Z\|_p - \frac{p-1}{p}\sqrt{\frac{2}{n}}\sqrt{\mathrm{Var}(Z^p/\mathbb{E}[Z^p])}\|Z\|_2, \tag{11a}$$

and if $\|Z\|_\infty \le C$, then

$$\mathbb{E}\left[\left(\frac{1}{n}\sum_{i=1}^n Z_i^p\right)^{\frac{1}{p}}\right] \ge \|Z\|_p - C\frac{p-1}{p}\sqrt{\frac{2}{n}}. \tag{11b}$$

See Appendix B.5 for the proof. Now, note that $[Z - \eta]_+ \in [0, 1 + c_{\rho,k}]M$ and $(1+\rho)^{1/k}(1 + c_{\rho,k}) =: C_{\rho,k}$. Thus, by Lemma 3 we obtain that

$$\begin{aligned}
\mathbb{E}[S_{N_w}(\eta)] &\ge (1+\rho)^{1/k}\mathbb{E}\left[[Z - \eta]_+^{k^*}\right]^{1/k^*} \\
&\quad + \eta - C_{\rho,k}M\frac{k^*-1}{k^*}\sqrt{\frac{2}{N_w}}.
\end{aligned}$$

Using that $\frac{k^*-1}{k^*} = \frac{1}{k}$, taking the infimum over η on the right hand side and using duality yields

$$\inf_\eta \mathbb{E}[S_{N_w}(\eta)] \ge \sup_{Q \in \mathcal{P}}\mathbb{E}_Q[Z] - C_{\rho,k}\frac{M}{k}\sqrt{\frac{2}{N_w}}.$$

To bound the second term in expression (10), we use concentration results for Lipschitz functions. First, the function $\eta \mapsto S_{N_w}(\eta)$ is $\sqrt{1+\rho}$-Lipschitz in η. To see this, note that for $1 \le k^\star \le 2$ and $X \ge 0$, by Jensen's inequality,

$$\frac{\mathbb{E}[X^{k^\star-1}]}{(\mathbb{E}[X^{k^\star}])^{1-1/k^\star}} \le \frac{\mathbb{E}[X]^{k^\star-1}}{(\mathbb{E}[X^{k^\star}])^{1-1/k^\star}} \le \frac{\mathbb{E}[X]^{k^\star-1}}{\mathbb{E}[X]^{k^\star-1}} = 1,$$

so $S'_{N_w}(\eta) \in [1 - (1+\rho)^{\frac{1}{k}}, 1]$ and therefore S_{N_w} is $(1+\rho)^{1/k}$-Lipschitz in η. Furthermore, the mapping $T : z \mapsto (1+\rho)^{\frac{1}{k}}(\frac{1}{N_w}\sum_{i=1}^{N_w}[z_i - \eta]_+^{k^*})^{\frac{1}{k^*}}$ for $z \in \mathbb{R}^{N_w}$ is convex and $(1+\rho)^{\frac{1}{k}}/\sqrt{N_w}$-Lipschitz. This is verified by the following:

$$\begin{aligned}
|T(z) - T(z')| &\le (1+\rho)^{1/k}\left|\left(\frac{1}{N_w}\sum_{i=1}^{N_w}\left|[z_i - \eta]_+ - [z'_i - \eta]_+\right|^{k^*}\right)^{\frac{1}{k^*}}\right| \\
&\le \frac{(1+\rho)^{1/k}}{{N_w}^{1/k^*}}\left|\left(\sum_{i=1}^{N_w}|z_i - z'_i|^{k^*}\right)^{\frac{1}{k^*}}\right| \\
&\le \frac{(1+\rho)^{1/k}}{\sqrt{N_w}}\|z - z'\|_2,
\end{aligned}$$

where the first inequality is Minkowski's inequality and the third inequality follows from the fact that for any vector $x \in \mathbb{R}^n$, we have $\|x\|_p \leq n^{\frac{2-p}{2p}} \|x\|_2$ for $p \in [1,2]$, where these denote the usual vector norms. Thus, the mapping $Z \mapsto S_{N_w}(\eta)$ is $(1+\rho)^{1/k}/\sqrt{N_w}$-Lipschitz continuous with respect to the ℓ_2-norm on Z. Using Samson's sub-Gaussian concentration result for convex Lipschitz functions, we have

$$\mathbb{P}\left(|S_{N_w}(\eta) - \mathbb{E}[S_{N_w}(\eta)]| \geq \delta\right) \leq 2\exp\left(-\frac{N_w \delta^2}{2C_{\rho,k}^2 M^2}\right)$$

for any fixed $\eta \in \mathbb{R}$ and any $\delta \geq 0$. Now, let $\mathcal{N}(U,\epsilon) = \{\eta_1, \ldots, \eta_{N(U,\epsilon)}\}$ be an ϵ cover of the set U, which we may take to have size at most $N(U,\epsilon) \leq M(1+c_{\rho,k})\frac{1}{\epsilon}$. Then we have

$$\begin{aligned}&\sup_{\eta \in U} |S_{N_w}(\eta) - \mathbb{E}[S_{N_w}(\eta)]| \\ &\leq \max_{i \in \mathcal{N}(U,\epsilon)} |S_{N_w}(\eta_i) - \mathbb{E}[S_{N_w}(\eta_i)]| + \epsilon(1+\rho)^{1/k}.\end{aligned}$$

Using the fact that $\mathbb{E}[\max_{i \leq n} |X_i|] \leq \sqrt{2\sigma^2 \log(2n)}$ for X_i all σ^2-sub-Gaussian, we have

$$\begin{aligned}&\mathbb{E}\left[\max_{i \in \mathcal{N}(U,\epsilon)} |S_{N_w}(\eta_i) - \mathbb{E}[S_{N_w}(\eta_i)]|\right] \\ &\leq C_{\rho,k}\sqrt{2\frac{M^2}{N_w}\log 2N(U,\epsilon)}.\end{aligned}$$

Taking $\epsilon = M(1+c_{\rho,k})/N_w$ gives that

$$\begin{aligned}&\mathbb{E}\left[\sup_{\eta \in U} |S_{N_w}(\eta) - \mathbb{E}[S_{N_w}(\eta)]\right] \\ &\leq \sqrt{2}MC_{\rho,k}\sqrt{\frac{1}{N_w}\log(2N_w)} + \frac{C_{\rho,k}M}{N_w}.\end{aligned}$$

Then, in total we have (using $C_\rho \geq C_{\rho,k}$, $k \geq 2$, and $N_w \geq 1$),

$$\begin{aligned}\mathbb{E}\left[\sup_{Q \in \mathcal{P}_{N_w}} \mathbb{E}_Q[Z]\right] &\geq \sup_{Q \in \mathcal{P}} \mathbb{E}_Q[Z] - \\ &\frac{C_\rho M\sqrt{2}}{\sqrt{N_w}}\left(\frac{1}{k} + \sqrt{\log(2N_w)} + \frac{1}{\sqrt{2N_w}}\right) \\ &\geq \sup_{Q \in \mathcal{P}} \mathbb{E}_Q[Z] - 4C_\rho M\sqrt{\frac{\log(2N_w)}{N_w}}.\end{aligned}$$

This gives the desired result of the lemma.

B.4. Proof of Lemma 2

Let $L \geq 0$ satisfy $L(w) = dQ(w)/dP(w)$, so that L is the likelihood ratio between Q and P. Then we have

$$\begin{aligned}&\sup_{Q: D_\phi(Q\|P) \leq \rho} \int g(w) dQ(w) = \sup_{\int \phi(L) dP \leq \rho, \mathbb{E}_P[L]=1} \int g(w)L(w)dP(w) \\ &= \sup_{L \geq 0} \inf_{\lambda \geq 0, \eta} \left\{\int g(w)L(w)dP(w) - \lambda\left(\int f(L(w))dP(w) - \rho\right)\right. \\ &\left. - \eta\left(\int L(w)dP(w) - 1\right)\right\} \\ &= \inf_{\lambda \geq 0, \eta} \sup_{L \geq 0} \left\{\int g(w)L(w)dP(w) - \lambda\left(\int f(L(w))dP(w) - \rho\right)\right. \\ &\left. - \eta\left(\int L(w)dP(w) - 1\right)\right\},\end{aligned}$$

where we have used that strong duality obtains because the problem is strictly feasible in its non-linear constraints (take $L \equiv 1$), so that the extended Slater condition holds (Luenberger, 1969, Theorem 8.6.1 and Problem 8.7). Noting that L is simply a positive (but otherwise arbitrary) function, we obtain

$$\begin{aligned}&\sup_{Q: D_\phi(Q\|P) \leq \rho} \int g(w) dQ(w) \\ &= \inf_{\lambda \geq 0, \eta} \int \sup_{\ell \geq 0} \left\{(g(w) - \eta)\ell - \lambda\phi(\ell)\right\} dP(w) + \lambda\rho + \eta \\ &= \inf_{\lambda \geq 0, \eta} \int \lambda\phi^*\left(\frac{g(w) - \eta}{\lambda}\right) dP(w) + \eta + \rho\lambda.\end{aligned}$$

Here we have used that $\phi^*(s) = \sup_{t \geq 0}\{st - \phi(t)\}$ is the conjugate of ϕ and that $\lambda \geq 0$, so that we may take divide and multiply by λ in the supremum calculation.

B.5. Proof of Lemma 3

For $a > 0$, we have

$$\inf_{\lambda \geq 0}\left\{\frac{a^p}{p\lambda^{p-1}} + \lambda\frac{p-1}{p}\right\} = a,$$

(with $\lambda = a$ attaining the infimum), and taking derivatives yields

$$\frac{a^p}{p\lambda^{p-1}} + \lambda\frac{p-1}{p} \geq \frac{a^p}{p\lambda_1^{p-1}} + \lambda_1\frac{p-1}{p} + \frac{p-1}{p}\left(1 - \frac{a^p}{\lambda_1^p}\right)(\lambda - \lambda_1).$$

Using this in the moment expectation, by setting $\lambda_n = \sqrt[p]{\frac{1}{n}\sum_{i=1}^n Z_i^p}$, we have for any $\lambda \geq 0$ that

$$\begin{aligned}\mathbb{E}\left[\left(\frac{1}{n}\sum_{i=1}^n Z_i^p\right)^{\frac{1}{p}}\right] &= \mathbb{E}\left[\frac{\sum_{i=1}^n Z_i^p}{pn\lambda_n^{p-1}} + \lambda_n\frac{p-1}{p}\right] \\ &\geq \mathbb{E}\left[\frac{\sum_{i=1}^n Z_i^p}{pn\lambda^{p-1}} + \lambda\frac{p-1}{p}\right] \\ &+ \frac{p-1}{p}\mathbb{E}\left[\left(1 - \frac{\sum_{i=1}^n Z_i^p}{n\lambda^p}\right)(\lambda_n - \lambda)\right].\end{aligned}$$

Now we take $\lambda = \|Z\|_p$, and we apply the Cauchy-Schwarz

inequality to obtain

$$\begin{aligned}
&\mathbb{E}\left[\left(\frac{1}{n}\sum_{i=1}^n Z_i^p\right)^{\frac{1}{p}}\right] \geq \|Z\|_p \\
&-\frac{p-1}{p}\mathbb{E}\left[\left(1-\frac{\frac{1}{n}\sum_{i=1}^n Z_i^p}{\|Z\|_p^p}\right)^2\right]^{\frac{1}{2}}\mathbb{E}\left[\left(\left(\frac{1}{n}\sum_{i=1}^n Z_i^p\right)^{\frac{1}{p}}-\|Z\|_p\right)^2\right]^{\frac{1}{2}} \\
&=\|Z\|_p-\frac{p-1}{p\sqrt{n}}\sqrt{\mathrm{Var}(Z^p/\mathbb{E}[Z^p])}\mathbb{E}\left[\left(\left(\frac{1}{n}\sum_{i=1}^n Z_i^p\right)^{\frac{1}{p}}-\mathbb{E}[Z^p]^{\frac{1}{p}}\right)^2\right]^{\frac{1}{2}} \\
&\geq \|Z\|_p-\frac{p-1}{p\sqrt{n}}\sqrt{\mathrm{Var}(Z^p/\mathbb{E}[Z^p])}\mathbb{E}\left[\left(\frac{1}{n}\sum_{i=1}^n Z_i^p\right)^{\frac{2}{p}}+\mathbb{E}[Z^p]^{\frac{2}{p}}\right]^{\frac{1}{2}} \\
&\geq \|Z\|_p-\frac{p-1}{p}\sqrt{\frac{2}{n}}\sqrt{\mathrm{Var}(Z^p/\mathbb{E}[Z^p])}\|Z\|_2,
\end{aligned}$$

where the last inequality follows by the fact that the norm is non-decreasing in p.

In the case that we have the unifom bound $\|Z\|_\infty \leq C$, we can get tighter guarantees. To that end, we state a simple lemma.

Lemma 4. *For any random variable $X \geq 0$ and $a \in [1,2]$, we have*

$$\mathbb{E}[X^{ak}] \leq \mathbb{E}[X^k]^{2-a}\mathbb{E}[X^{2k}]^{a-1}$$

Proof For $c \in [0,1]$, $1/p + 1/q = 1$ and $A \geq 0$, we have by Holder's inequality,

$$\mathbb{E}[A] = \mathbb{E}[A^c A^{1-c}] \leq \mathbb{E}[A^{pc}]^{1/p}\mathbb{E}[A^{q(1-c)}]^{1/q}$$

Now take $A := X^{ak}$, $1/p = 2-a$, $1/q = a-1$, and $c = \frac{2}{a} - 1$. □

First, note that $\mathbb{E}[Z^{2p}] \leq C^p\mathbb{E}[Z^p]$. For $1 \leq p \leq 2$, we can take $a = 2/p$ in Lemma 4, so that we have

$$E[Z^2] \leq \mathbb{E}[Z^p]^{2-\frac{2}{p}}\mathbb{E}[Z^{2p}]^{\frac{2}{p}-1} \leq \|Z\|_p^p C^{2-p}.$$

Now, we can plug these into the expression above (using $\mathrm{Var}Z^p \leq \mathbb{E}[Z^{2p}] \leq C^p\|Z\|_p^p$), yielding

$$\mathbb{E}\left[\left(\frac{1}{n}\sum_{i=1}^n Z_i^p\right)^{\frac{1}{p}}\right] \geq \|Z\|_p - C\frac{p-1}{p}\sqrt{\frac{2}{n}}$$

as desired.

B.6. Proof of Proposition 2

We utilize the following lemma for regret of online mirror descent.

Lemma 5. *The expected regret for online mirror descent with unbiased stochastic subgradient $\gamma(t)$ and stepsize η is*

$$\sum_{t=1}^T \mathbb{E}\left[\gamma(t)^T(w(t)-w^\star)\right] \leq \frac{\log(d)}{\eta} + \frac{\eta}{2}\mathbb{E}\left[\sum_{t=1}^T\sum_{j=1}^d w_j(t)\gamma_j(t)^2\right] \tag{12}$$

See Appendix B.7 for the proof. Now we bound the right-hand term of the regret bound (12) in our setting. For this we utilize the following:

$$\begin{aligned}
\mathbb{E}\left[\gamma_i(t)^2|w(t)\right] &= \frac{1}{N_w^2}\frac{L_i^2(t)}{w_i^2(t)}\mathbb{E}\left[\left(\sum_{k=1}^{N_w}\mathbf{1}\left\{J_k=i\right\}\right)^2\middle| w(t)\right] \\
&= \frac{1}{N_w^2}\frac{L_i^2(t)}{w_i^2(t)}\left(N_w(N_w-1)w_i(t)^2 + N_w w_i(t)\right),
\end{aligned}$$

where the latter fact is simply the second moment for the sum of N_w random variables $\overset{\text{i.i.d.}}{\sim}$ Bernoulli$(w_i(t))$. Then,

$$\begin{aligned}
\sum_{i=1}^d w_i(t)\mathbb{E}\left[\gamma_i(t)^2|w(t)\right] &= \sum_{i=1}^d L_i(t)^2\left(\frac{N_w-1}{N_w}w_i(t)+\frac{1}{N_w}\right) \\
&\leq \sum_{i=1}^d\left(\frac{N_w-1}{N_w}w_i(t)+\frac{1}{N_w}\right) \\
&= \frac{N_w-1}{N_w}+\frac{d}{N_w} \\
&=: z.
\end{aligned}$$

Plugging in the prescribed $\eta = \sqrt{\frac{2\log(d)}{zT}}$ into the bound (12) yields the result.

B.7. Proof of Lemma 5

We first show the more general regeret of online mirror descent with a Bregman divergence and then specialize to the entropic regularization case. Let $\psi(w)$ be a convex fuction and $\psi^*(\theta)$ its Fenchel conjugate. Define the Bregman divergence $B_\psi(w,w') = \psi(w) - \psi(w') - \nabla\psi(w')^T(w-w')$. In the following we use the subscript $\cdot_t$ instead of $(\cdot)(t)$ for clarity. The standard online mirror descent learner sets

$$w_t = \operatorname*{argmin}_w\left(\gamma_t^T w + \frac{1}{\eta}B_\psi(w,w_t)\right).$$

Using optimality of w_{t+1} in the preceding equation, we have

$$\begin{aligned}
\gamma_t^T(w_t-w^*) &= \gamma_t^T(w_{t+1}-w^*) + \gamma_t^T(w_t-w_{t+1}) \\
&\leq \frac{1}{\eta}(\nabla\psi(w_{t+1})-\nabla\psi(w_t))^T(w^*-w_{t+1}) \\
&\quad + \gamma_t^T(w_t-w_{t+1}) \\
&= \frac{1}{\eta}\left(B_\psi(w^*,w_t) - B_\psi(w^*,w_{t+1}) - B_\psi(w_{t+1},w_t)\right) \\
&\quad + \gamma_t^T(w_t-w_{t+1}).
\end{aligned}$$

Summing this preceding display over iterations t yields

$$\begin{aligned}
\sum_{t=1}^T\gamma_t^T(w_t-w^*) &\leq \frac{1}{\eta}B_\psi(w^*,w_1) \\
&\quad + \sum_{t=1}^T\left(-\frac{1}{\eta}B_\psi(w_{t+1},w_t) + \gamma_t^T(w_t-w_{t+1})\right)
\end{aligned}$$

Now let $\psi(w) = \sum_i w_i\log w_i$. Then, with $w_1 = \mathbf{1}/d$, $B\psi(w^*,w_1) \leq \log(d)$. Now we bound the second term with the following lemma.

Lemma 6. *Let $\psi(x) = \sum_j x_j\log x_j$ and $x, y \in \Delta$ be defined by: $y_i = \frac{x_i\exp(-\eta g_i)}{\sum_j x_j\exp(-\eta g_j)}$ where $g \in \mathbb{R}_+^d$ is non-negative. Then*

$$-\frac{1}{\eta}B_\psi(y,x) + g^T(x-y) \leq \frac{\eta}{2}\sum_{i=1}^d g_i^2 x_i.$$

See Appendix B.8 for the proof. Setting $y = w_{t+1}$, $x = w_t$,

and $g = \gamma_t$ in Lemma 6 yields

$$\sum_{t=1}^{T} \gamma_t^T (w_t - w^*) \leq \frac{\log(d)}{\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \sum_{j=1}^{d} w_j(t) \gamma_j(t)^2.$$

Taking expectations on both sides yields the result.

B.8. Proof of Lemma 6

Note that $B_\psi(y, x) = \sum_i y_i \log \frac{y_i}{x_i}$. Substituting the values for x and y into this expression, we have

$$\sum_i y_i \log \frac{y_i}{x_i} = -\eta g^T y - \sum_i y_i \log \left(\sum_j x_j e^{-\eta g_j} \right)$$

Now we use a Taylor expansion of the function $g \mapsto \log \left(\sum_j x_j e^{-\eta g_j} \right)$ around the point 0. If we define the vector $p_i(g) = x_i e^{-\eta g_i} / \left(\sum_j x_j e^{-\eta g_j} \right)$, then

$$\log \left(\sum_j x_j e^{-\eta g_j} \right) = \log(\mathbf{1}^T x) - \eta p(0)^T g + \frac{\eta^2}{2} g^\top \left(\operatorname{diag}(p(\widetilde{g})) - p(\widetilde{g}) p(\widetilde{g})^\top \right) g$$

where $\widetilde{g} = \lambda g$ for some $\lambda \in [0, 1]$. Noting that $p(0) = x$ and $\mathbf{1}^T x = \mathbf{1}^T y = 1$, we obtain

$$B_\psi(y, x) = \eta g^T (x - y) - \frac{\eta^2}{2} g^\top \left(\operatorname{diag}(p(\widetilde{g})) - p(\widetilde{g}) p(\widetilde{g})^\top \right) g,$$

whereby

$$-\frac{1}{\eta} B_\psi(y, x) + g^T (x - y) \leq \frac{\eta}{2} \sum_{i=1}^{d} g_i^2 p_i(\widetilde{g}). \tag{13}$$

Lastly, we claim that the function

$$s(\lambda) = \sum_{i=1}^{d} g_i^2 \frac{x_i e^{-\lambda g_i}}{\sum_j x_j e^{-\lambda g_j}}$$

is non-increasing on $\lambda \in [0, 1]$. Indeed, we have

$$s'(\lambda) = \frac{\left(\sum_i g_i x_i e^{-\lambda g_i} \right) \left(\sum_i g_i^2 x_i e^{-\lambda g_i} \right)}{\left(\sum_i x_i e^{-\lambda g_i} \right)^2} - \frac{\sum_i g_i^3 x_i e^{-\lambda g_i}}{\sum_i x_i e^{-\lambda g_i}}$$
$$= \frac{\sum_{ij} g_i g_j^2 x_i x_j e^{-\lambda g_i - \lambda g_j} - \sum_{ij} g_i^3 x_i x_j e^{-\lambda g_i - \lambda g_j}}{\left(\sum_i x_i e^{-\lambda g_i} \right)^2}$$

Using the Fenchel-Young inequality, we have $ab \leq \frac{1}{3}|a|^3 + \frac{2}{3}|b|^{3/2}$ for any a, b so $g_i g_j^2 \leq \frac{1}{3} g_i^3 + \frac{2}{3} g_j^3$. This implies that the numerator in our expression for $s'(\lambda)$ is non-positive. Thus, $s(\lambda) \leq s(0) = \sum_{i=1}^{d} g_i^2 x_i$ which gives the result when combined with inequality (13).

C. Hardware

The major components of the vehicle used in experiments are shown in Figure 7. The chassis of the 1/10-scale vehicles used in experiments are based on a Traxxas Rally 1/10-scale radio-controlled car with an Ackermann steering mechanism. An electronic speed controller based on an open source design (Vedder) controls the RPM of a brushless DC motor and actuates a steering servo. A power distribution board manages the power delivery from a lithium polymer (LiPo) battery to the onboard compute unit and sensors. The onboard compute unit is a Nvidia Jetson Xavier, a system-on-a-chip that contains 8 ARM 64 bit CPU cores and a 512 core GPU. The onboard sensor for localization is a planar LIDAR that operates at 40Hz with a maximum range of 30 meters. The electronic speed controller also provides odometry via the back EMF of the motor.

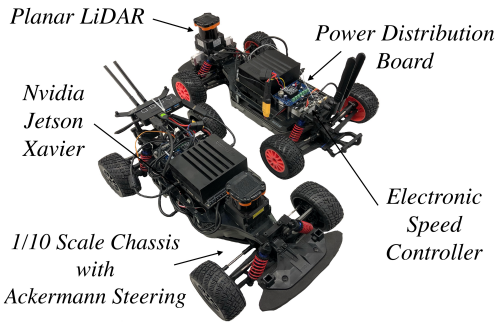

Figure 7. Components of the 1/10 Scale Vehicle

D. Vehicle Software Stack

This section gives a detailed overview of the software used onboard the vehicles. Figure 8 gives a graphical overview.

D.1. Mapping

We create occupancy grid maps of tracks using Google Cartographer (Hess et al., 2016). The map's primary use is as an efficient prior for vehicle localization algorithms. In addition, maps serve as a representation of the static portion of the simulation environment describing where the vehicle may drive and differentiating which (if any) portions of the LIDAR scan have line-of-sight to other agents. A feature of our system useful to other researchers is that any environment which can be mapped may be trivially added to the simulator described in Appendix E.

D.2. Localization

Due to the speeds at which the vehicles travel, localization must provide pose estimates at a rate of at least 20 Hz. Thus, to localize the vehicle we use a particle filter (Walsh & Karaman, 2017) that implements a ray-marching scheme on the GPU in order to efficiently simulate sensor observations in parallel. We add a small modification which captures the covariance of the pose estimate. We do not use external localization systems (*e.g.* motion capture cameras) in any experiment.

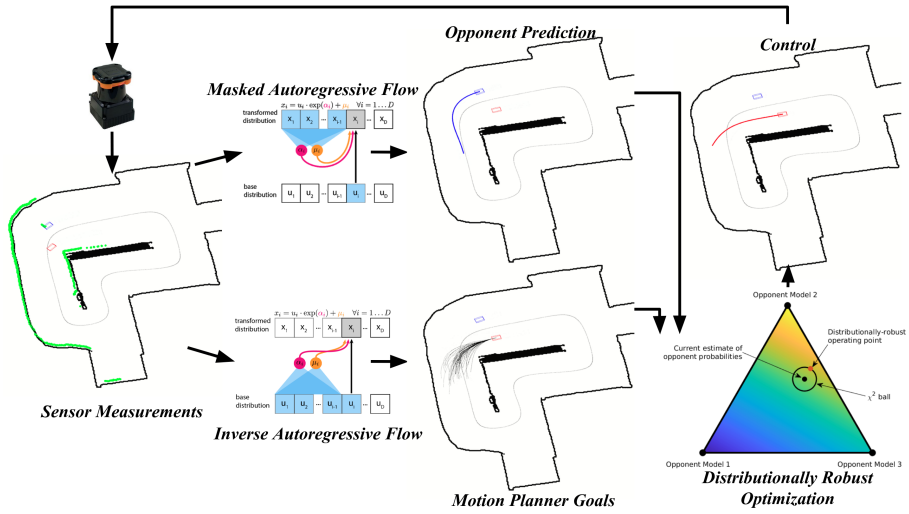

Figure 8. FormulaZero implementation on vehicle. Online each agent measures the world using onboard sensors such as a planar LIDAR. Given the sensor measurement the vehicle performs opponent prediction via the use of a masked autoregressive flow and simultaneously selects motion planner goals using an inverse autoregressive flow. Given the set of goals each is evaluated within our DRO framework, the best goal is chosen, and a new control command is applied to the vehicle. Then, the process occurs again.

D.3. Planning

The vehicle software uses a hierarchical planner (Gat et al., 1998) similar to that of Ferguson et al. (2008). At the top level the planner receives a map and waypoints representing the centerline of the track; the goal is to traverse the track from start to finish. Unlike route planning in road networks, there are no routing decisions to be made. In more complex instances of our proposed environment, this module could be necessary for making strategic decisions such as pit stops. The second key difference is the mid-level planner. Whereas Ferguson et al. (2008) uses a deterministic lattice of points, our vehicle draws samples from a neural autoregressive flow. Each sample contains a goal pose and speed profile. Given this specification, the local planner calculates a trajectory parameterized as a cubic spline, evaluates static and dynamic costs of the proposed plan in belief space, and selects the lowest cost option.

D.3.1. Sampling behavior proposals

There are two advantages to using a neural autoregressive flow in our planning framework. First, each agent in the population weights the individual components of its cost function differently; the flow enables the goal generation mechanism to learn a distribution which places more probability mass on the agent's preferences. Second, as planning takes place in the context of the other agent's actions, the ego-agent's beliefs can be updated by inverting the flow and estimating the likelihood of the other agent's actions under a given configuration of the cost function.

The goal-generation process utilizes an inverse autoregressive flow (IAF) (Kingma et al., 2016). The IAF samples are drawn from a density conditioned on a 101-dimensional observation vector composed of a subsampled LIDAR scan and current speed. Each sample is a 6 dimensional vector: Δt, the perpendicular offset of the goal pose from the track's centerline; Δs, the arc-length along the track's centerline relative to the vehicle's current pose; $\Delta\theta$, the difference between the goal pose's heading angle and the current heading angle; three velocity offsets from the vehicle's current velocity at three equidistant knot points along the trajectory.

The second benefit of using a generative model for sampling behavior proposals is the ability to update an agent's beliefs about the opponent's policy type. As noted in Section 4, masked (Papamakarios et al., 2017) and inverse autoregressive flows (MAF and IAF respectively) have complementary strengths. While sampling from a MAF is slow, density estimation using this architecture is fast. Thus, we use a MAF network trained to mimic the samples produced by the IAF for this task. The architectures of each network are the same, and we describe this architecture below.

The IAF and MAF networks used in this paper have 5 MADE layers (Papamakarios et al., 2017) each containing: a masked linear mapping ($\mathbb{R}^6 \to \mathbb{R}^{100}$), RELU layer, masked linear mapping ($\mathbb{R}^{100} \to \mathbb{R}^{100}$), RELU layer, and a final masked linear layer ($\mathbb{R}^{100} \to \mathbb{R}^{12}$). Note that output of a MADE layer includes both the transformed sample and the logarithm of the absolute value of the determinant of the Jacobian of the transformation. For sampling, the latter is discarded, and the transformed sample is passed to the next layer. In addition, the masking pattern is sequential and held constant during both training and inference. This

choice was made to aid in debugging of experiments and to simplify communication during distributed training.

Each population member has a dedicated IAF model, which is trained iteratively according to the AADAPT algorithm described in Section 2 using the hyperparameters given in Section 4. We initialize each IAF with a set of weights which approximate an identity transformation for random pairs of samples from a normal distribution and simulated observations. In addition each population member also has a MAF model, which is trained using the same hyperparameters as the IAF but only after AADAPT has finished. The code submitted in the supplementary materials extends an existing library[5] created by other authors; we add support for the IAF architecture as well as generalize the network architecture to 3-dimensional tensors. The latter extension enables sampling from multiple agents' IAF models simultaneously and efficiently.

D.3.2. Model Predictive Control

The goal of the trajectory generator is to compute kinematically and dynamically feasible trajectories that take the vehicle from its current pose to a set of sampled poses from the IAF. The trajectory generator combines approaches from (Howard, 2009; Nagy & Kelly, 2001; Kelly & Nagy, 2003; McNaughton, 2011). Each trajectory is represented by a cubic spiral with five parameters $p = [s, a, b, c, d]$ where s is the arc length of the spiral, and (a, b, c, d) encode the curvature at equispaced knot points along the trajectory. Powell's method or gradient descent can be used to find the spline parameters that (locally) minimize the sum of the Euclidean distance between the desired endpoint pose and the forward simulated pose. Offline, a lookup table of solutions for a dense grid of goal poses is precomputed, enabling fast trajectory generation online. Each trajectory is associated with an index which selects the Δx, Δy, and the $\Delta\theta$ of the goal pose relative to the current pose (where positive x is ahead of the vehicle and postiive y is to the left), and κ_0, the initial curvature of the trajectory. The resolution and the range of the table is listed in Table 5. Figure 9 shows a selection of trajectories. The point on the left of the figure is the starting pose of the vehicle, and the collection of goal poses is shown as the points on the right of the figure.

D.3.3. Trajectory Cost Functions

Each of the generated trajectories is evaluated with the weighted sum of the following cost functions. Note, in order to ensure safety, goals which would result in collision result in infinite cost and are automatically rejected prior to computing the robust cost, which operates only on finite-cost proposals.

[5] https://github.com/kamenbliznashki/normalizing_flows

Table 5. The resolution and ranges of the Trajectory Generator Look-up Table

Index	Resolution	Min	Max
Δx	0.1 m	-1.0 m	10.0 m
Δy	0.1 m	-8.0 m	8.0 m
$\Delta\theta$	$\pi/32$ rad	$-\pi/2$ rad	$\pi/2$ rad
κ_0	0.2 rad/m	-1.0 rad/m	1.0 rad/m

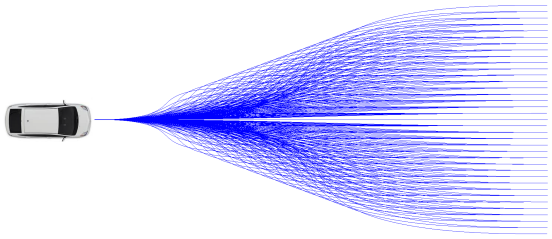

Figure 9. Sample trajectories from the look-up table

1. **Trajectory length:** $c_{al} = s$, where $1/s$ is the arc length of each trajectory. Short and myopic trajectories are penalized.

2. **Maximum absolute curvature:** $c_{mc} = \max_i\{|\kappa_i|\}$, where κ_i are the curvatures at each point on a trajectory. Large curvatures are penalized to preserve smoothness of trajectories.

3. **Mean absolute curvature:** $c_{ac} = \frac{1}{N}\sum_{i=0}^{N}|\kappa_i|$, the notation is the same as c_{mc} and the effect of this feature is similar, but less myopic.

4. **Hysteresis loss:** Measured between the previous chosen trajectory and each of the sampled trajectories, $c_{hys} = ||\theta_{prev}^{[n_1,n_2]} - \theta^{[0,n_2-n_1]}||_2^2$, where θ_{prev} is the array of heading angles of each pose on the previous selected trajectory by the vehicle, θ is the array of heading angles of each pose on the trajectory being evaluated, and the ranges $[n_1, n_2]$ and $[0, n_2 - n_1]$ define contiguous portions of trajectories that are compared. Trajectories dissimilar to the previously selected trajectory are penalized.

5. **Lap progress:** Measured along the track from the start to the end point of each trajectory in the normal and tangential coordinate system, $c_p = \frac{1}{s_{end}-s_{start}}$, where s_{end} is the corresponding position in the tangential coordinate along the track of the end point of a trajectory, and s_{start} is that of the start point of a trajectory. Shorter progress in distance is penalized.

6. **Maximum acceleration:** $c_{ma} = \max_i|\frac{\Delta v_i}{\Delta t_i}|$ where Δv is the array of difference in velocity between adjacent points on a trajectory, and Δt is the array of

corresponding time intervals between adjacent points. High maximum acceleration is penalized.

7. **Maximum absolute curvature change:** Measured between adjacent points along each trajectory, $c_{dk} = \max_i |\frac{\Delta\kappa_i}{\Delta t_i}|$. High curvature changes are penalized.
8. **Maximum lateral acceleration:** $c_{la} = \max_i\{|\kappa|_i v_i^2\}$, where κ and v are the arrays of curvature and velocity of all points on a trajectory. High maximum lateral accelerations are penalized.
9. **Minimum speed:** $c_{ms} = \frac{1}{(\min_i\{v_i\})_+}$. Low minimum speeds are penalized.
10. **Minimum range:** $c_{mr} = \min_i\{r_i\}$, where r is the array of range measurements (distance to static obstacles) generated by the simulator. Smaller minimum range is penalized, and trajectories with minimum ranges lower than a threshold are given infinite cost and therefore discarded.
11. **Cumulative inter-vehicle distance short:** $c_{dyshort} = \begin{cases} \infty, \text{ if } d(\texttt{ego}_i, \texttt{opp}_i) \leq \texttt{thresh} \\ \sum_{i=0}^{N_{short}} d(\texttt{ego}_i, \texttt{opp}_i), \text{ otherwise} \end{cases}$ Where the function $d()$ returns the instantaneous minimum distance between the two agents at point i, N_{short} is a point that defines the shorter time horizon for a trajectory of N points. Trajectories with infinite cost on the shorter time horizon are considered infeasible and discarded.
12. **Discounted cumulative inter-vehicle distance long:** $c_{dylong} = \sum_{i=N_{short}}^{N_{long}} 0.9^{i-N_{short}} \frac{1}{d(\texttt{ego}_i, \texttt{opp}_i)}$, where N_{long} is a point that defines the longer time horizon for a trajectory of N points. Note that $N_{short} < N_{long} < N$. Lower minimum distances between agents on the longer time horizon are penalized.
13. **Relative progress:** Measured along the track between the sampled trajectories' endpoints and the opponent's selected trajectory's endpoint, $c_{dp} = (s_{opp\_end} - s_{end})_+$, where s_{opp_end} is the position along the track in tangential coordinates of the endpoint of the opponent's chosen trajectory. Lagging behind the opponent is penalized.

D.3.4. PATH TRACKER

Once a trajectory has been selected it is given to the path-tracking module. The goal of the path tracker is to compute a steering input which drives the vehicle to follow the desired trajectory. Our implementation uses a simple and industry-standard geometrical tracking method called pure pursuit (Coulter, 1992; Snider et al., 2009). Due to the decoupling of the trajectory generation and tracking modules it is possible for the tracker to run at a much higher frequency than the trajectory generator; this is essential for good performance.

D.4. Communication and system architecture

The ZeroMQ (Hintjens, 2013) messaging library is used to create interfaces between the FormulaZero software stack and the underlying ROS nodes that control and actuate the vehicle test bed. Unlike in the simulator, some aspects of the FormulaZero planning function operate non-deterministicaly and asynchronously. In particular we use a sink node to collect observations from ROS topics related to the various sensors on the vehicle in order to approximate the step-function present in the Gym API. When a planning cycle is complete, the trajectory is published back to ROS and tracked asynchronously using pure-pursuit as new pose estimates become available. Because perception is not the primary focus of this project we simplify the problem of detecting and tracking the other vehicle. In particular, each vehicle estimates its current pose in the map obtained by its onboard particle filter, and this information is communicated to the other vehicle via ZeroMQ over a local wireless network. Since tracking and detection has been well studied in robotics, solutions which rely less on communication could be explored by other future work which builds upon this paper.

E. Simulation Stack

The simulation stack includes a lightweight 2D physics engine with a dynamical vehicle model. Then on top of the physics engine, a multi-agent simulator with an OpenAI Gym (Brockman et al., 2016) API is used to perform rollouts of the experiments.

E.1. Vehicle Dynamics

The single-track model in Althoff et al. (2017) is chosen because it considers tire slip influences on the slip angle, which enables accurate simulation at physical limits of the vehicle test bed. It is also easily enables changes to the driving surface friction coefficient in simulation which allows the simulator to model a variety of road surfaces.

E.2. System Identification

Parameter identification was performed to derive the following vehicle parameters: mass, center of mass, moment of inertia, surface friction coefficient, tire cornering stiffness, and maximum acceleration/deceleration rates following the methods described in O'Kelly et al. (2019).

E.3. Distributed Architecture

Due to the nature of the AADAPT algorithm, the rollouts in a single vertical step do not need to be in sequence. The ZeroMQ messaging library is used to create a MapReduce (Dean & Ghemawat, 2008) pattern between the task distributor, result collector, and the workers. Each worker receives the description of the configuration to be simualted, *e.g.* (x, θ). Then the workers asynchronously perform simulations and send results to the collector.

E.4. Addressing the simulation/reality gap

As noted in Section 4 there are several differences between the observations in simulated rollouts and reality. First, pose estimation errors are not present in the simulator. A simple fix would be to add Gaussian white noise to the pose observations returned by the simulator. We avoided this and other domain randomization techniques in order to preserve the determinism of the simulator, but we will investigate its effect in further experiments. Second, the LIDAR simulation does not account for material properties of the environment. In particular, surfaces such as glass do not produce returns, causing subsets of the LIDAR beams to be dropped. We hypothesize that simple data augmentation schemes which select a random set of indices to drop from simulated LIDAR observations would improve the robustness to such artifacts when the system is deployed on the real car; we are currently investigating this hypothesis.

F. Experiments

Additional videos of simulation runs are available.[6]

F.1. Instantaneous time-to-collision (iTTC)

Let $T_i(t)$ be the instantaneous time-to-collision between the ego vehicle and the i-th environment vehicle at time step t. The value $T_i(t)$ can be defined in multiple ways (see *e.g.* Sontges et al. (2018)). Norden et al. (2019) define it as the amount of time that would elapse before the two vehicles' bounding boxes intersect assuming that they travel at constant fixed velocities from the snapshot at time t. Time-to-collision captures directly whether or not the ego-vehicle was involved in a crash. If it is positive no crash occurred, and if it is 0 or negative there was a collision.

F.2. Out-of-distribution agent strategies

In the following sections, we describe the human-created algorithms used in our out-of-distribution analysis.

[6] https://youtu.be/8q0lZssbEI4

F.2.1. OOD1: RRT* with MPC-based Opponent Prediction

This approach exploits the fact that the two-car racing scenario is similar to driving alone on the track with the only exception being during overtaking the opponent. This approach uses a costmap-based RRT* (Karaman & Frazzoli, 2011) planning algorithm. The agent first uses the opponent's current pose and velocity in the world, and uses Model-Predictive Control to calculate an open loop trajectory of N optimal inputs resulting in N+1 states based on a given cost function and constraints. Specifically, the optimization problem is constrained by a linearized version of the single track model described in Althoff et al. (2017), and by the boundary values of the inputs and states of the vehicle. The cost function that the optimization tries to minimize consists of the trajectory length and input power requirement. The costmap used by RRT* also incorporates this predicted trajectory of the opponent vehicle by inflating the two-dimensional spline representing the prediction, and weighting the portion of the spline closer to the ego vehicle higher. RRT* samples the two dimensional space that the vehicle lies in. The path generated by RRT* is then tracked with the Pure Pursuit controller (Coulter, 1992).

F.2.2. OOD2: RL-based Lane Switching

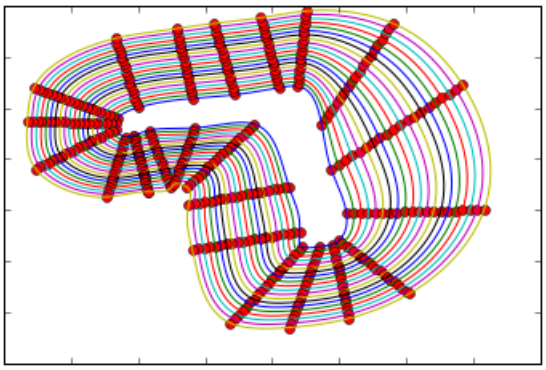

Figure 10. Lanes that cover the track

The second algorithm is based on a lane-switching planning strategy that uses an RL algorithm to make lane switching decisions, and filters out unsafe decisions using a collision indicator. First, as shown in 10, different lanes going through numerous checkpoints on the track are created to cover the entirety of the race track. Then a network is trained to make lane switching decisions. The state of the RL problem consists of the sub-sampled LIDAR scans of the ego vehicle; the pose (x, y, θ) of the opponent car with respect to the ego vehicle; velocity (v_x, v_y) of the opponent vehicle with respect of the ego vehicle; projected distance from the ego vehicle's current position to all pre-defined paths. The reward of a rollout is zero in the beginning. At

each timestep, the timestep itself is subtracted from the total reward. A rollout receives -100 as the reward when the ego agent collide with the environment or the other agent. And finally, if both agents finish 2 laps, the difference between lap times (positive if the ego agent wins) of the two agents are added to the reward. Clipped Double Q-Learning (Fujimoto et al., 2018) is used to estimate the Q function and make the lane switching decisions. iTTC defined in Appendix F.1 is used as an indicator for future collisions. If any decisions made by the RL network would result in a collision indicated by the iTTC value, the safety function kicks in and makes the lane switching decision based on the collision indicator. Finally, ego vehicle actuation is provided by the same Pure Pursuit controller (Coulter, 1992) tracking the selected lane. We used an existing implementation[7] of this algorithm.

[7] https://github.com/pnorouzi/rl-path-racing