A Appendix A: Full Regret Proof

A.0 Outline of Proof

1. Reduce path planner case to linear dynamical systems case.
2. Extension of Suggala and Natrapalli to Nonconvex FPL with memory (not ours; done in Ghai et al. (2021)).
3. Argue that our particular nonconvexity has a specific, efficient solver (e.g., Online Eig with Memory).
4. Justify truncated state history as being equivalent extension to memory setting (again, straight from Ghai, 2021 Ghai et al. (2021), which is from Agarwal, 2019 Agarwal et al. (2019a) for first result). There are some subtleties here that are unique to our particular instantiation.
5. Put everything together at the end.

A.1 Reduction of Path Planned case to Standard Controls case

Assume that the planner devises a nominal path (denoted with a $(\bar{\cdot})^0$ notation) in coordinates $\mathbf{x}$ and inputs $\mathbf{u}$: so the path $\mathcal{P}$ is fully specified as $\mathcal{P} = \{\bar{\mathbf{x}}_t^0, \bar{\mathbf{u}}_t^0\}_{t=0}^T$. Assume that the path is chosen so that at every $\mathbf{x}$ on or near the path, the following dynamics hold around perturbations of the path:

$$\mathbf{x}_t - \bar{\mathbf{x}}_t^0 = A(\mathbf{x}_{t-1} - \bar{\mathbf{x}}_{t-1}^0) + B(\mathbf{u}_{t-1} - \bar{\mathbf{u}}_{t-1}^0) + D\mathbf{w}_{t-1}. \tag{14}$$

Using this change of coordinates, we can essentially negate the path and study the relevant perturbation dynamics $\delta\mathbf{x}_t := \mathbf{x}_t - \bar{\mathbf{x}}_t^0$ and $\delta\mathbf{u}_t := \mathbf{u}_t - \bar{\mathbf{u}}_t^0$, we recover the desired equation:

$$\delta\mathbf{x}_t = A\delta\mathbf{x}_{t-1} + B\delta\mathbf{u}_{t-1} + D\mathbf{w}_{t-1}. \tag{15}$$

For shorthand, we will define $\mathbf{x} := \delta\mathbf{x}$ and $\mathbf{u} = \delta\mathbf{u}$ to ease exposition, remembering that they represent perturbations from the nominal path. Intuitively, this seems like a reasonable model for 'quasi-static' systems (e.g., drone or car or aircraft path planning when the maneuvers are non-aggressive).

A.2 Proof of Theorem 7

Recall that at time t, the algorithm A has access to the state trajectory $\{\mathbf{x}_\tau^A\}_{\tau=1}^t$, the disturbance history $\{\mathbf{w}_\tau\}_{\tau=1}^{t-1}$, and the sets of sensed obstacles $\{\mathbf{p}_\tau^j\}_{\tau=1}^t$. For any $\tau \in \{H, ..., t\}$, the loss function can be written as an instance of Alg. 2. Concretely, let $\mathbf{a}_\tau^j = \tilde{A}\mathbf{x}_{\tau-1} + D\mathbf{w}_{\tau-1} - \mathbf{p}_\tau^j$, $\mathbf{b}_\tau = \mathbf{w}_{\tau-H:\tau-1}$, $\mathbf{b}_{0,\tau} = \tilde{A}\mathbf{x}_{\tau-1} + D\mathbf{w}_{\tau-1}$. Then, for an appropriate $\mathbf{c}_\tau \in \Delta_{k_\tau}$, the optimization problems are equivalent. Concatenating over the τ-formulations, we have an instance of Eqn. (9). Specifically, for some choice of $\mathbf{c}$, we will have an equivalent problem as Eqn. (8); here $\mathbf{c}$ is an – unknown *a priori* – encoding of the relevant (nearest) obstacles.

Now, we need to demonstrate that Alg. 2 will converge to a pair $\{\mathbf{c}_N, M_N\}$ that corresponds to the optimal solution of Eqn. (8). This follows immediately from Theorem 7, Part II of Hazan (2006). We have an instance of a repeated game in which, by Lemma 6, an optimization oracle efficiently solves Eqn. (9), and then the low-regret exponentiated gradient algorithm iteratively updates $\mathbf{c}_n$ (Eqn. (10)).

A.3 Extension of Nonconvex FPL to Nonconvex Memory FPL

This result is from Ghai et al. (2021), Theorem 13 (Corollary 14 gives an equivalent result to our setting in the asymptotic regret behavior; our optimal choice of η and ϵ differs slightly).

A.4 Efficient Solution of Opt

Consider the relaxed optimization problem

$$\max_{M \in \mathcal{M}} \sum_j \lambda_j \|\mathbf{a}_j + BM\mathbf{b}\|_2^2 \tag{16}$$

We will first describe some useful quantities and (physically-motivated) assumptions. The physical quantities of interest have the following characteristics: $\mathbf{x} \in \mathbb{R}^{d_x}$, $\mathbf{u} \in \mathbb{R}^{d_u}$, and $\mathbf{w} \in \mathbb{R}^{d_w}$.

Assumption 11. *$B \in \mathbb{R}^{d_x \times d_u}$, with $d_u \leq d_x$, and rank $(B) = d_u$. This corresponds to the following physical assumptions: (1) there are no more inputs than states, and (2) there are no 'extraneous' inputs (if there are such inputs, then we can find a minimal realization of B and wlog set extraneous inputs to always be zero). Similarly, if (1) fails, then we can remove extraneous inputs by again setting them equal to zero uniformly (just as in (2)).*

The dimension of several other variables of interest are: $b \in \mathbb{R}^{Hd_w}$ and for the decision variable M, $M \in \mathbb{R}^{d_u \times Hd_w}$. We want to show that the optimization in Eqn. 16 is equivalent to a convex trust region problem, which is efficiently solvable.

Further, let the quantity $B^T B$ have a singular value decomposition denoted by:

$$B^T B = U^T \Lambda U.$$

We see that this decomposition has an orthogonal $U \in \mathbb{R}^{d_u \times d_u}$ and positive definite Λ because rank$(B) = d_u$ and thus the symmetric $B^T B \in \mathbb{R}^{d_u \times d_u}$ has $B^T B \succ 0$.

Now, consider the problem of Eqn. 16, assuming fixed $\lambda \in \Delta_p, B, \{\mathbf{a}_j\}_{j=1}^p, \mathbf{b}$. We rearrange the objective as follows:

$$\begin{aligned}
\text{OBJ} &= \sum_j \lambda_j \|\mathbf{a}_j + BM\mathbf{b}\|_2^2 \\
&= \sum_j \lambda_j (\mathbf{a}_j + BM\mathbf{b})^T (\mathbf{a}_j + BM\mathbf{b}) \\
&= \sum_j \lambda_j (\mathbf{a}_j^T \mathbf{a}_j + 2\mathbf{a}_j^T BM\mathbf{b} + \mathbf{b}^T M^T B^T BM\mathbf{b}) \\
&\equiv \sum_j \lambda_j (2\mathbf{c}_j^T M\mathbf{b} + \mathbf{b}^T M^T U^T \Lambda U M\mathbf{b})
\end{aligned}$$

Here, we are searching for the argmax M^*, so the $\mathbf{a}_j^T \mathbf{a}_j$ is irrelevant. Further, we have defined $\mathbf{c}_j^T = \mathbf{a}_j^T B$. Now, let $M_r := UM$, and decompose $M_r = [\mathbf{m}_1^T; \mathbf{m}_2^T; ...; \mathbf{m}_{d_u}^T]$. The optimization objective is now:

$$\begin{aligned}
\text{OBJ} &= \sum_j \lambda_j \|\mathbf{a}_j + BM\mathbf{b}\|_2^2 \\
&\equiv \sum_j \lambda_j (2\mathbf{c}_j^T M\mathbf{b} + \mathbf{b}^T M^T U^T \Lambda U M\mathbf{b}) \\
&= \sum_j \lambda_j (2\mathbf{c}_j^T M\mathbf{b} + \mathbf{b}^T M_r^T \Lambda M_r \mathbf{b}) \\
&= [\sum_j \lambda_j (2\mathbf{c}_j^T M\mathbf{b})] + \mathbf{b}^T M_r^T \Lambda M_r \mathbf{b}
\end{aligned}$$

The last simplification follows from the fact that $\sum_j (\lambda_j) = 1$ (because $\lambda \in \Delta_{d_u}$). Now, using our knowledge of the diagonal nature of Λ and the column partition of M_r, we can see that

$$M_r^T \Lambda M_r = \sum_{j=1}^{d_u} \sigma_j^2 \mathbf{m}_j \mathbf{m}_j^T$$

Substituting into the OPT formulation, we can further simplify all the way to the desired form:

$$\begin{aligned}
\text{OBJ} &= \sum_i \lambda_i \|\mathbf{a}_i + BM\mathbf{b}\|_2^2 \\
&\equiv \Big[\sum_i \lambda_i (2\mathbf{c}_i^T M\mathbf{b})\Big] + \mathbf{b}^T M_r^T \Lambda M_r \mathbf{b} \\
&= \Big[\sum_i \lambda_i (2\mathbf{c}_i^T M\mathbf{b})\Big] + \mathbf{b}^T (\sum_{j=1}^{d_u} \sigma_j^2 \mathbf{m}_j \mathbf{m}_j^T)\mathbf{b} \\
&= \Big[\sum_i \lambda_i (2\mathbf{c}_i^T U^T M_r\mathbf{b})\Big] + (\sum_{j=1}^{d_u} \sigma_j^2 \mathbf{b}^T \mathbf{m}_j \mathbf{m}_j^T \mathbf{b}) \\
&= 2\Big[\sum_i \lambda_i (\sum_j \tilde{c}_{i,j} \mathbf{m}_j^T \mathbf{b})\Big] + (\sum_{j=1}^{d_u} \sigma_j^2 \mathbf{m}_j^T \mathbf{b}\mathbf{b}^T \mathbf{m}_j) \\
&= \sum_j \Big[(2\sum_i \lambda_i \tilde{c}_{i,j})\mathbf{b}^T\Big]\mathbf{m}_j) + (\sum_{j=1}^{d_u} \mathbf{m}_j^T (\sigma_j^2 \mathbf{b}\mathbf{b}^T)\mathbf{m}_j) \\
&= \mathbf{m}^T P \mathbf{m} + \mathbf{p}^T \mathbf{m}.
\end{aligned}$$

where $\mathbf{m}$ is a vector concatenation of the transposed rows of M_r. Once we solve for $\mathbf{m}^*$ using a trust region solver, we unpack it into M_r^*, and get $M^* = U^T M_r^* (= U^T(UM^*) = M^*)$ as desired. Further, we can translate a norm bound on M into an equivalent one on $\mathbf{m}$ (at least, for appropriate choice of norm bound - e.g., the Frobenius norm on M becomes the 2-norm on $\mathbf{m}$).

A.4.1 Technical Notes: Continuity and Conditioning Parameters

We begin with an analysis of the Lipschitz constant for the approximate cost functions (this will follow a similar path to Ghai et al. (2021)).

First, note that the diameter of the decision set is $2D_M$ and that the gradient of the quadratic cost above is $\nabla_{\mathbf{m}} \ell_t = (P + P^T)\mathbf{m} + \mathbf{p}$. As such,

$$\begin{aligned}
L &:= \max_{\mathbf{m},t} \{\|\nabla_{\mathbf{m}} \ell_t(\mathbf{m})\|_\infty\} \\
&\leq \max_{\mathbf{m},t} \{(\|P\|_1 + \|P\|_\infty) D_M + \|\mathbf{p}\|_\infty\} \\
&\leq 2Hd_w RD + R
\end{aligned}$$

We consider as well a bound on the conditioning number of the optimization problem. Because the size of the optimization grows linearly in time, the condition number grows at most linearly as well. Therefore, the run-time of the algorithm is polynomial (neither the condition number nor the dimension grows too rapidly).

Finally, we note bounds on the elements of P and $\mathbf{p}$ in the trust region instance. The bounds on costs, states, inputs, and disturbances together imply that the elements of P_t are bounded by $C_u^2 \kappa^2 \xi$, and the elements of $\mathbf{p}$ are bounded by $C_u^2 \kappa^2 \xi \beta$ (this again follows Ghai et al. (2021)).

A.5 Truncated States

This follows directly from Ghai et al. (2021), *except* that the truncated state history affects the resulting vectors $\mathbf{a}_j$ in the optimization. We will need to show that this does not affect the resulting cost too much.

Interestingly, there is a bit of an added subtlety here, which arises from the observation that in certain scenarios, small perturbations in the observed relative obstacle positions could yield large changes in the optimal policy. For example, imagine that there is one obstacle, located directly on the centerline of the nominal planned motion. Then a small perturbation of the obstacle to the right makes the optimal action “Left," while a small perturbation of the obstacle to the left makes the optimal action “Right."

However, this is not a problem in the regret outline, because while the optimal decision is fragile, the loss incurred of choosing incorrectly is bounded by the quadratic (and therefore, continuous) nature of the costs. Intuitively, take wlog the case where the vehicle's maximal deviation from the centerline is 1 on each side of a centered obstacle. Then, for a ε-error in the estimate of the obstacle position, By choosing optimally we could have achieved $(1+\varepsilon)^2$ reward and instead perhaps will get $(1-\varepsilon)^2$ reward. This is not a problem for the regret bound because we can accurately control ε via the choice of history length H (and because of the speed that ε decays as a function of H); therefore, this induced regret can be kept small.

For the dynamics and control we have assumed

$$\begin{aligned} x_{t+1} &= A\mathbf{x}_t + B\mathbf{u}_t + D\mathbf{w}_t \\ \mathbf{u}_t &= K\mathbf{x}_t + M_t\tilde{\mathbf{w}}_t \\ &= K\mathbf{x}_t + \sum_{i=1}^{H} M_t^{[i]}\mathbf{w}_{t-i}, \end{aligned} \tag{17}$$

and we can show (as in Ghai et al. (2021)) that the state can be expressed as the sum of disturbance-to-state transfer function matrices $\Psi_{t,i}$:

$$\mathbf{x}_{t+1}^{\mathcal{A}} = \tilde{A}^{H+1}\mathbf{x}_{t-H}^{\mathcal{A}} + \sum_{i=0}^{2H}\Psi_{t,i}\mathbf{w}_{t-i}, \text{ where}$$

$$\tilde{A} = A + BK \text{ and}$$

$$\Psi_{t,i} = \tilde{A}^i D\mathbb{1}[i \le H] + \sum_{j=0}^{H}\tilde{A}^j B M_{t-j}^{[i-j]}\mathbb{1}[i-j \in \{1,...,H\}].$$

We define the state estimate and cost as

$$\mathbf{y}_{t+1} := \sum_{i=0}^{2H}\Psi_{t,i}\mathbf{w}_{t-i}$$

$$\ell_t(M_{t-H:t}) = c_t(\mathbf{y}_{t+1}(M_{t-H:t}), \tilde{\mathbf{u}}_t)$$

where $\tilde{\mathbf{u}}_t = M_t\tilde{\mathbf{w}}_t$ (the residual input on top of the closed-loop controller).

Now, assume that $\|\tilde{A}\| \le 1-\gamma$, that $\|\tilde{A}\|, \|B\|, \|D\|, \|K\| \le \beta$, and that for all t it holds that $\|w_t\| \le C_w$, $\|u_t\| \le C_u$, and $\|Q_t\|, \|R_t\| \le \xi$. Then we can show that the approximation error of the costs is sufficiently small. Let the condition number be defined as $k = \|\tilde{A}\|\|\tilde{A}^{-1}\|$.

A.5.1 Bound the States

Note that $\tilde{\mathbf{u}}_t = M_t\tilde{\mathbf{w}}_t$; this implies that $\|\tilde{\mathbf{u}}_t\| \le HDC_w$. This implies further that $\|B\tilde{\mathbf{u}}_t + D\mathbf{w}_t\| \le 2\beta HDC_w$ by the triangle inequality. Assuming that there exists τ such that

$$\|\mathbf{x}_\tau\|_2 \le \frac{2\beta HDC_w}{\gamma},$$

we have that for every $t > H + \tau + 1$, $\|\mathbf{x}_{t-H-1}^{\mathcal{A}}\|_2 \le \frac{2\beta HDC_w}{\gamma}$. (WLOG, we can assume the initial state $\mathbf{x}_0$ is bounded in this domain - that is, that the assumption is satisfied with $\tau = 0$; the region defined above is the long-term reachable set of the state $\mathbf{x}_t$ driven by bounded disturbances $\mathbf{w}_t$ and (implicitly bounded) residual inputs $\tilde{\mathbf{u}}_t$ [the norm is limited by the stability parameter γ of the closed-loop $\tilde{A}$-matrix]).

A.5.2 Bound the Change in Costs

Now, we analyze the change in costs

$$|c_t(\mathbf{x}_{t+1}^{\mathcal{A}}, \tilde{\mathbf{u}}_t) - \ell_t(M_{t-H:t})| = |\min_{j\in[p]}\|\mathbf{a}_{j,t} + BM_t\tilde{\mathbf{w}}_t\|_2^2 - \min_{j\in[p]}\|\hat{\mathbf{a}}_{j,t} + BM_t\tilde{\mathbf{w}}_t\|_2^2|$$

Noting the definition of $\hat{\mathbf{a}}_{j,t}$ and of $\mathbf{a}_{j,t}$, we can bound the difference between them as a function of the error in approximation of $\mathbf{x}_t$ (see Ghai et al. (2021)):

$$\begin{aligned}
&\mathbf{a}_{j,t} := \mathbf{p}_{j,t} - \mathbf{x}_t \\
\implies &\hat{\mathbf{a}}_{j,t} - \mathbf{a}_{j,t} = (\mathbf{p}_{j,t} - \hat{\mathbf{x}}_t) - (\mathbf{p}_{j,t} - \mathbf{x}_t) \\
&= \mathbf{x}_t - \hat{\mathbf{x}}_t \\
\implies &\|\hat{\mathbf{a}}_{j,t} - \mathbf{a}_{j,t}\|_2 = \|\mathbf{x}_t - \hat{\mathbf{x}}_t\|_2 \\
&\leq kC_x e^{-\gamma H}
\end{aligned}$$

Now, we argue that the loss incurred due to the noise in $\hat{\mathbf{x}}_t$ is less than simply twice the change in cost due to the error in $\hat{\mathbf{a}}_{j,t}$. Let $\hat{j}^* := \arg\min_{j\in[p]}\{\|\hat{\mathbf{a}}_{j,t} - BM_t\tilde{\mathbf{w}}_t\|_2^2\}$. Let j^* be defined analogously. If $j^* = \hat{j}^*$, then the difference in cost is less than or equal to the extra loss incurred by the error in $\hat{\mathbf{a}}$. If $j^* \neq \hat{j}^*$, then it is possible that the true 'binding obstacle' was biased away, and that the 'guessed' binding obstacle was 'biased towards'; therefore, the cost error is possibly due to deviations up to twice the error in the $\hat{\mathbf{a}}_{j,t}$ vectors. This means that, defining δ_t such that $\|\delta_t\|_2 = 2\|\mathbf{x}_t - \hat{\mathbf{x}}_t\|_2$, we have that the following holds:

$$\Delta = |c_t(\mathbf{x}_{t+1}^{\mathcal{A}}, \tilde{\mathbf{u}}_t) - \ell_t(M_{t-H:t})| = |\min_{j\in[p]}\|\mathbf{a}_{j,t} + BM_t\tilde{\mathbf{w}}_t\|_2^2 - \min_{j\in[p]}\|\hat{\mathbf{a}}_{j,t} + BM_t\tilde{\mathbf{w}}_t\|_2^2|$$

$$\leq \max_{\delta_t:\|\delta_t\|_2\leq 2kC_x e^{-\gamma H}} \left\{\|(\hat{\mathbf{a}}_{j,t} + \delta_t) + BM_t\tilde{\mathbf{w}}_t\|_2^2 - \|\hat{\mathbf{a}}_{j,t} + BM_t\tilde{\mathbf{w}}_t\|_2^2\right\}$$

$$= \delta_t^T\delta_t + 2\delta_t^T\hat{\mathbf{a}}_{j,t} - 2\delta_t^T(BM_t\tilde{\mathbf{w}}_t)$$

$$\leq \|\delta_t\|_2^2 + 2(C_x + \|\delta_t\|_2)\|\delta_t\|_2 + 2\|\delta_t\|_2 C_w\beta D_M$$

$$= 3\|\delta_t\|_2^2 + 2C_x\|\delta_t\|_2 + 2C_w\beta D_M\|\delta_t\|_2$$

$$\leq 5C_x\|\delta_t\|_2 + 2C_w\beta D_M\|\delta_t\|_2$$

$$\leq 5(k^2C_x^2e^{-\gamma H}(1 + \beta D_M C_w))$$

.

Letting $H = \lceil\gamma^{-1}\log(5k^2C_x(1+\beta D_M C_w)T)\rceil$, we have that

$$\Delta \leq \frac{C_x}{T}$$

Remark 12. ***Recursive Definition of*** H ***and*** C_x***:***
Currently, there is a recursive nature to the definition of H and C_x; $H := \lceil\gamma^{-1}\log(5k^2C_x(1+\beta DC_w)T)\rceil$ and $C_x := \frac{2\beta HDC_w}{\gamma}$. However, this is not problematic because the definitions will have a solution (that can be found efficiently); namely:

$$\begin{aligned}
H &\geq c_1\log(c_2C_x) \\
C_x &= k_1H \\
\implies H &\geq c_1\log(c_2k_1H)
\end{aligned}$$

And for any $c_1, c_2, k_1 \in \mathbb{R}^+$ and fixed $T > 0$, there exists a positive integer H such that the above result holds (e.g., following from the fact that $\log H = o(H)$). Further, the resulting H will not be too large wrt T for sufficiently large T (e.g., large enough T to overcome the constants).

A.6 PUTTING EVERYTHING TOGETHER

Finally, we use the OTR Solver Algorithm (which acts as an efficient ε-oracle) with an approximate trust region implementation of our desired optimization problem in order to compose the regret components into a total bound.

$$\begin{aligned}
\text{Regret}(\mathcal{A}) &:= \max_{M\in\Pi} \sum_{t=H}^{T} c_t(x_t^M, \tilde{u}_t(M)) - \sum_{t=H}^{T} c_t(x_t^{\mathcal{A}}, \tilde{u}_t(\mathcal{A})) \\
&\leq \max_{M\in\Pi} \sum_{t=H}^{T} (f_t(M, M, ..., M) + \frac{C_x}{T}) - \sum_{t=H}^{T} (f_t(M_{t-H:t}) + \frac{C_x}{T}) \\
&= \Big[\max_{M\in\Pi} \sum_{t=H}^{T} f_t(M, M, ..., M) - \sum_{t=H}^{T} f_t(M_{t-H:t})\Big] + \mathcal{O}(\log T) \\
&\leq \tilde{\mathcal{O}}(\text{poly}(\mathcal{L})\sqrt{T})
\end{aligned} \tag{18}$$

To clarify the steps: the second line incorporates the approximation error from Section A.5 (which is logarithmic in T, as noted in the third line) and the final line follows from the Suggala and Natrapalli regret bound Suggala & Netrapalli (2020) extended to the memory setting in Section A.3.

B APPENDIX B: EXPERIMENT HYPERPARAMETERS

In this section, we report the hyperparameters used for the experiments results in the main text. We implemented our algorithm and environments in JAX. All experiments were carried out on a single CPU in minutes.

We set the full horizon T to 100 and the history length H to 10. For random perturbation across environments, we sample noise from Gaussian distribution with mean 0 and standard deviation 0.5. For directional perturbation, we sample Gaussian noise with mean 0.5 and standard deviation 0.5. A random seed of 0 is used for all experiments. We obtain the nominal control from LQR with Q set to 0.001 and R set to 1. We then learn the residual obstacle-avoiding parameter M via gradient descent. The learning rate of gradient descent is 0.008 in the centerline environment and is 0.001 for the other environments.

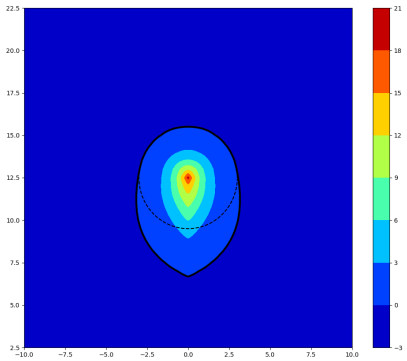

Figure 3: Racer backwards reachable set (inside thick black line) and the obstacle (dashed black line).

C Appendix C: Additional Experimental Resources

C.1 RRT* Implementation

An important note for these experiments: we do not implement existing heuristic techniques like obstacle padding to improve the RRT* collision-avoidance performance. As such, this performance is not meant to suggest that RRT* *cannot* work robustly in these settings, only that its nominal (and theoretically grounded) form does not account for disturbances or uncertainty and is therefore "optimistic" as compared to HJ methods, etc.

C.2 Additional Figures and Trajectories

This appendix includes sample trajectories and other relevant visualizations for each algorithm.

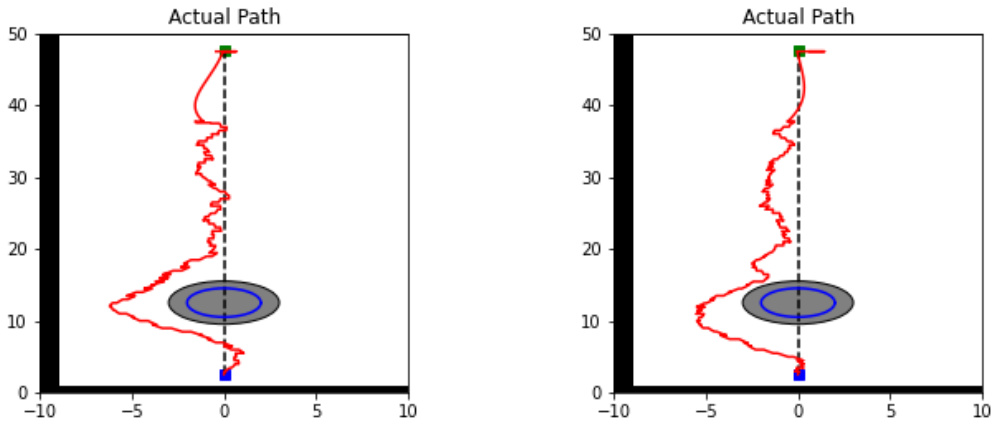

Figure 4: RRT* Planner trajectories against uniform random disturbances. Obstacle is the gray sphere, with the nominal trajectory a dashed black (vertical) line.

C.2.1 HJ Reachability Planner

For the centerline example, the HJ Reachability planner constructs in Fig. 3 the backwards-reachable set for a given obstacle (dashed line), subject to the dynamics constraints imposed on the racer. Note that every positive-value region denotes an unsafe region. The interpretation is that there is a "pseudo-cone" in front of the obstacle from which the vehicle cannot escape hitting the obstacle *if the disturbances are sufficiently adversarial*. Note that this means that HJ planning is independent of the *actual* disturbances. For each of the disturbance patterns (random, sinusoid, adversarial), we plot a collapsed view of sample trajectories around an obstacle for the HJ planner in Fig. 9. Note how similar each plot is, due to this independence of the control from the actual observed disturbances.

C.2.2 RRT* Planner

Here, we demonstrate some sample paths for RRT* in each disturbance regime. Fig. 4 shows uniform random noise, Fig. 5 shows sinusoidal noise, and Fig. 6 shows adversarial noise. In each case, the disturbance at the final time (goal position, top of image) causing a horizontal shift in the path should be ignored.

C.2.3 Online Learned Planner

The key illustration here is that the trajectories of the online planner follow the structure of the disturbances, as illustrated by the following comparison of the uniform random and sinusoidal disturbances in Fig. 7 and Fig. 8.

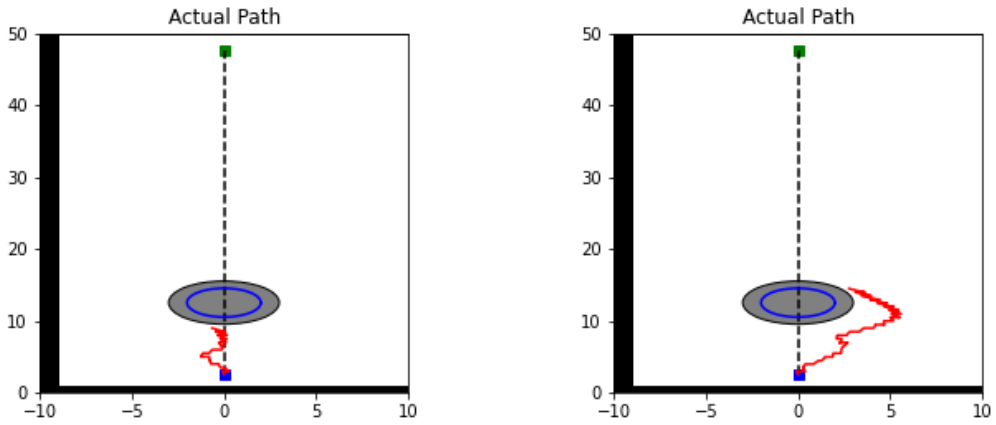

Figure 5: RRT* Planner trajectories against sinusoidal disturbances. Obstacle is the gray sphere, with the nominal trajectory a dashed black (vertical) line.

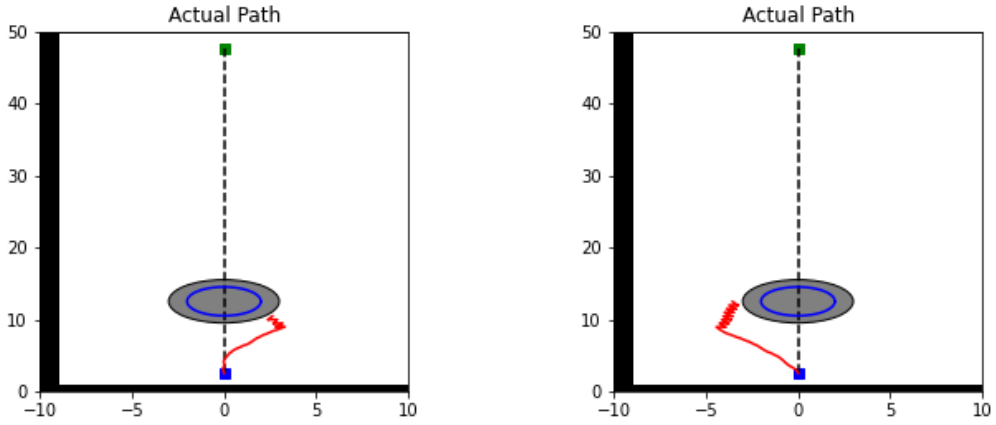

Figure 6: RRT* Planner trajectories against adversarial disturbances. Obstacle is the gray sphere, with the nominal trajectory a dashed black (vertical) line.

C.3 The Slalom Setting

The final key challenge of this work is discussing the effects of the slalom setting: why the online planner fails on our examples, and what this suggests about our obstacle avoidance framework.

The first answer is relatively direct: in all of our examples, we are implicitly acting in a kind of Frenet frame, where all obstacle positions and other referencing is to the ego vehicle (racer) position. As such, the nominal planned trajectory can always be thought of as mapped to a straight line ahead of the racer. In this context, the second slalom gate in Fig. 1 represents a 20m deviation *from the nominal trajectory.* However, this flies in the face of the central modeling intuition of the online framework – that obstacle avoidance is local, with local sensing, local deviations from the nominal trajectory, and "reactive" control to disturbances as they arise. In this vein, the nominal slalom is a challenging task, precisely because it stretches the limits of what can be met by our setup. Concretely: limited sensing makes each slalom wall a kind of "gradient-less" observation (shifting left and right yields only a continuation of the wall *unless the gap is already sensed*), meaning that choosing the correct Left/Right action is difficult. Additionally, the map displays memory, because going the wrong way early through one gate can render the next gate infeasible.

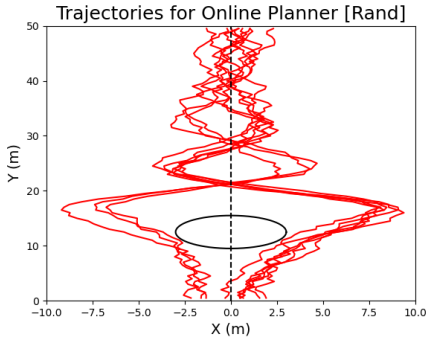

Figure 7: Collapsed trajectories of the racer using the online planner with random disturbances. Note that the optimal passing side is essentially random and evenly distributed.

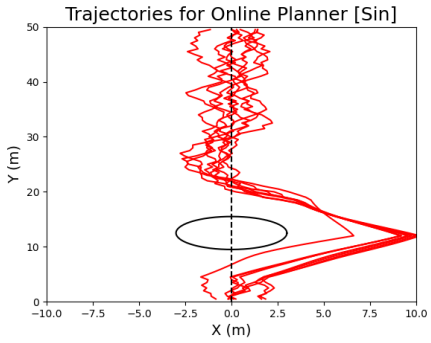

Figure 8: Collapsed trajectories of the racer using the online planner with sinusoidal disturbances. Note how the racer learns quickly to always pass on the right, which is the easier route.

It is in light of these considerations that we argue that the slalom case is actually a case for our model, because it interpretably creates a setting in which the key assumptions are broken. Just like an actual skier who overshoots through one gate and cannot recover for the next gate, so too does our obstacle avoidance algorithm run the risk of “dooming" itself due to a wrong turn – but this is, as described, fundamental to the hardness of the obstacle avoidance problem! As such, we consider the slalom gate as a fundamentally hard problem, and consider a case for future work a fuller characterization of how our planner works for slaloms of varying difficulty, as measured by the sensor range, the distance between gates (both laterally and longitudinally), and the fundamental “cost memory" as it depends on these and other parameters.

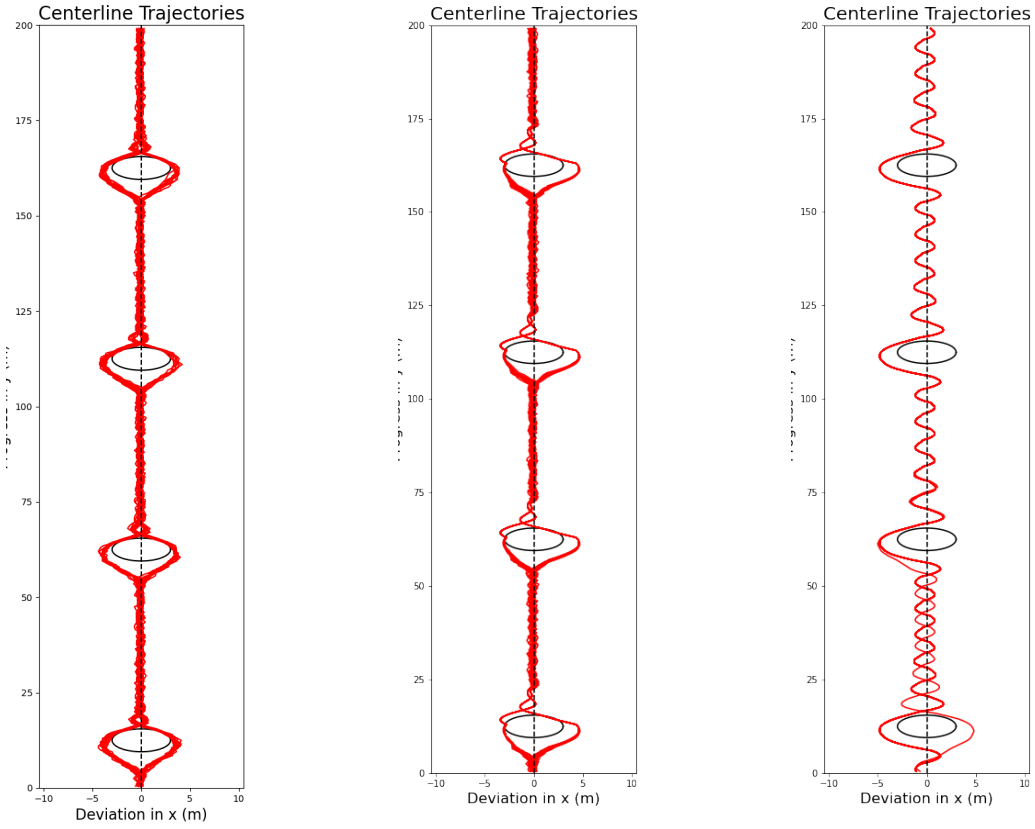

Figure 9: HJ Planner trajectories against (L) uniform random, (C) sinusoid, and (R) adversarial disturbances. Obstacles are black spheres, with the nominal trajectory a dashed black (vertical) line.