Solving Stabilize-Avoid Optimal Control via Epigraph Form and Deep Reinforcement Learning - Appendix

Oswin So, Chuchu Fan
Massachusetts Institute of Technology
{oswinso, chuchu}@mit.edu

CONTENTS

APPENDIX A
PROOFS

A1 Proof that $V^{l,\pi}$ is a discrete-time Lyapunov function

Before we begin the statement of the theorem and its proof, define $\mathcal{K}$ to be the class of functions $\gamma : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ that is continuous, zero at zero and strictly increasing. Let $\mathcal{K}_\infty$ to be the class that are $\mathcal{K}$ and are also unbounded. We then have the following definition of a Lyapunov function for discrete-time systems, which we modify from [4, Ch.2] to use $\sigma(\cdot)$ instead of $\|\cdot\|_{x^{\text{ref}}}$.

Definition 1 (Discrete-Time Lyapunov Function)**.** *Suppose the system has discrete-time dynamics*

$$x_{k+1} = f(x_k, u_k), \tag{A.1}$$

for states $x_k \in \mathcal{X}$ and controls $u_k \in \mathcal{U}$. Consider a reference set $\mathcal{G}$ and a subset of the state space $\mathcal{R} \subseteq \mathcal{X}$. Let $\sigma : \mathcal{X} \to \mathbb{R}_{\geq 0}$ be a state measure (as in [3]) that is continuous and positive-definite. A function $V : \mathcal{R} \to \mathbb{R}_{\geq 0}$ is a uniform Lyapunov function on $\mathcal{R}$ if the following conditions are satisfied.

(i) There exist functions $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ such that

$$\alpha_1(\sigma(x)) \leq V(x) \leq \alpha_2(\sigma(x)) \tag{A.2}$$

holds for all $x \in \mathcal{R}$.

(ii) There exists a function $\alpha_V \in \mathcal{K}$ such that

$$V(x_{k+1}) \leq V(x_k) - \alpha_V(\sigma(x)) \tag{A.3}$$

holds for all $x_k \in \mathcal{R}$.

Theorem 1 (Policy Value Function is Lyapunov)**.** *Let $\pi : \mathcal{X} \to \mathcal{U}$ be an arbitrary deterministic policy, and define $V^{l,\pi} : \mathcal{X} \to \mathbb{R}_{\geq 0} \cup \{+\infty\}$ to be the policy value function*

$$V^{l,\pi}(x_0) \coloneqq \sum_{k=0}^{\infty} l(x_k), \quad x_{k+1} = f(x_k, \pi(x_k)) \tag{A.4}$$

for cost function $l : \mathcal{X} \to \mathbb{R}_{\geq 0}$ and discrete dynamics $f : \mathcal{X} \times \mathcal{U} \to \mathcal{X}$. Let $\mathcal{F}$ denote the set where $V^{l,\pi}$ is finite, i.e.,

$$\mathcal{F} \coloneqq \left\{ x \;\middle|\; V^{l,\pi}(x) < \infty \right\}, \tag{A.5}$$

and let $\sigma : \mathcal{X} \to \mathbb{R}_{\geq 0}$ be a state measure (as in [3]) that is continuous and positive-definite. Suppose that the following holds for the cost function l and the policy value function $V^{l,\pi}$.

(i) There exists $\overline{\alpha} \in \mathcal{K}_\infty$ such that, for any $x \in \mathcal{F}$,

$$V^{l,\pi}(x) \leq \overline{\alpha}(\sigma(x)) \tag{A.6}$$

(ii) There exists $\overline{\rho} \in \mathcal{K}_\infty$ such that, for any $x \in \mathcal{F}$,

$$l(x) \geq \overline{\rho}(\sigma(x)) \tag{A.7}$$

Then, $V^{l,\pi}$ is a Lyapunov function on $\mathcal{F}$.

Proof. By the definition of $V^{l,\pi}$, use of dynamic programming shows that

$$V^{l,\pi}(x_k) = l(x_k) + V^{l,\pi}(x_{k+1}). \tag{A.8}$$

Since $V^{l,\pi} \geq 0$ by definition (A.4), by using (A.7) and (A.8) we can conservatively lower bound $V^{l,\pi}$ in terms of σ on $\mathcal{F}$ as

$$V^{l,\pi}(x) \geq l(x) \geq \overline{\rho}(\sigma(x)). \tag{A.9}$$

Combining the same two equations again without dropping $V^{l,\pi}(x_{k+1})$, we can also show that for $x \in \mathcal{F}$,

$$V^{l,\pi}(x_{k+1}) = V^{l,\pi}(x_k) - l(x_k) \leq V^{l,\pi}(x_k) - \overline{\rho}(\sigma(x)). \tag{A.10}$$

Combining (A.6), (A.9) and (A.10) then gives us that for $x_k \in \mathcal{F}$,

$$\overline{\rho}(\sigma(x)) \leq V^{l,\pi}(x) \leq \overline{\alpha}(\sigma(x)), \tag{A.11a}$$

$$V^{l,\pi}(x_{k+1}) \leq V^{l,\pi}(x_k) - \overline{\rho}(\sigma(x)). \tag{A.11b}$$

Since $\overline{\rho} \in \mathcal{K}_\infty$, $\overline{\alpha} \in \mathcal{K}_\infty$, (A.11a) and (A.11b) thus show that $V^{l,\pi}$ is a Lyapunov function on $\mathcal{F}$ by Definition 1. □

From Theorem 2, we can then apply the standard proof of local asymptotic stability using Lyapunov functions [4] to show asymptotic stability.

Corollary 1. *Define the set $\mathcal{Z}$ to be the zero-set of $V^{l,\pi}$, i.e.,*

$$\mathcal{Z} \coloneqq \left\{ x \;\middle|\; V^{l,\pi}(x) = 0 \right\}. \tag{A.12}$$

Then, $\mathcal{Z}$ is also the zero-set of σ, i.e.,

$$\mathcal{Z} = \left\{ x \mid \sigma(x) = 0 \right\}. \tag{A.13}$$

Moreover, $\mathcal{Z}$ is locally asymptotically stable within $\mathcal{F}$ under the controller π on $\mathcal{F}$.

Proof. First, note that by the definition of $\mathcal{K}$, (A.9) implies that for $x \in \mathcal{Z}$,

$$0 \leq \overline{\rho}(x) \leq V^{l,\pi}(x) = 0. \tag{A.14}$$

Moreover, since $\overline{\rho}$ is strictly increasing,

$$\overline{\rho}(x) > 0 \implies V^{l,\pi}(x) > 0. \tag{A.15}$$

Hence, the zero-set $\mathcal{Z}$ of $V^{l,\pi}$ is also the zero-set of σ. Applying Theorem 2.19 from [4] using the policy value function as the Lyapunov function as shown in Theorem 2 then gives us the result. □

Note. As noted in the main paper, while Theorem 2 and Corollary 1 show that we can use $V^{l,\pi}$ to show stability for *any* policy π within the region $\mathcal{F}$ under assumptions (A.7) and (A.6), we note that $\mathcal{F}$ may be a tiny set or even empty. Hence, the theorems above do not give us a direct method of constructing stable controllers. Nevertheless, the above theorems provide intuition on the relationship between the optimality of a policy (measured by the size of $\mathcal{F}$) and its stability, which we use when solving the infinite-horizon constrained OCP in the main paper.

A2 Equivalence of (11) and (12)

Theorem 2. *Let $x \in \mathbb{R}^n$ and $z \in \mathbb{R}$, and let $g : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ be a continuous (potentially non-differentiable) function. Then, if a solution exists (i.e., an optimal x^*, z^* exist, are finite), then the following optimization problems are equivalent.*

$$\begin{aligned} \min_{x,\,z} \quad & z \\ \text{s.t.} \quad & g(x,z) \le 0, \end{aligned} \tag{A.16}$$

$$\begin{aligned} \min_{z} \quad & z \\ \text{s.t.} \quad & \Big[\min_{x} g(x,z)\Big] \le 0, \end{aligned} \tag{A.17}$$

Proof. We begin by comparing the Lagrangian primal problem of eq. (A.16) and eq. (A.17).

$$\begin{aligned} &\min_z \min_x \max_{\lambda \ge 0} z + \lambda\, g(x,z) \\ &= \min_z \left\{ z + \min_x \max_{\lambda \ge 0} \lambda\, g(x,z) \right\} \end{aligned} \tag{A.18}$$

$$\begin{aligned} &\min_z \max_{\lambda \ge 0} z + \lambda \Big[\min_x g(x,z)\Big] \\ &= \min_z \left\{ z + \max_{\lambda \ge 0} \min_x \lambda\, g(x,z) \right\} \end{aligned} \tag{A.19}$$

Comparing the two, the only difference is that the order of $\min_x$ and $\max_\lambda$ are flipped. Hence, it is sufficient to show that, for any z where $\min_x g(x,z) < 0$,

$$p^* \coloneqq \min_x \max_{\lambda \ge 0} \lambda\, g(x,z) = \max_{\lambda \ge 0} \min_x \lambda\, g(x,z) =: d^*. \tag{A.20}$$

Note that this is exactly equivalent to showing that strong duality holds for the following constraint satisfaction problem.

$$\begin{aligned} \min_{x} \quad & 0 \\ \text{s.t.} \quad & g(x,z) \le 0, \end{aligned} \tag{A.21}$$

We now prove that strong duality holds for the above problem in a similar fashion to the proof that Slater's condition is a sufficient condition for strong duality to hold in convex optimization problems [1].

Define the set $\mathcal{A} \subseteq \mathbb{R}^n \times \mathbb{R}$ as

$$\mathcal{A} \coloneqq \{\, (u,t) \mid \exists x,\ g(x,z) \le u, \quad 0 \le t \,\}, \tag{A.22}$$

$$= \left\{\, u \;\middle|\; \inf_x g(x,z) \le u \,\right\} \times [0,\infty). \tag{A.23}$$

Note that $\mathcal{A}$ is convex. Furthermore, since a feasible solution exists by assumption, we have that

$$p^* = \min_x \max_{\lambda \ge 0} \lambda\, g(x,z) = \min_x \begin{cases} \infty & g(x,z) > 0 \\ 0 & g(x,z) \le 0 \end{cases} = 0. \tag{A.24}$$

We now define a second set $\mathcal{B} \subseteq \mathbb{R}^n \times \mathbb{R}$ as

$$\mathcal{B} \coloneqq \{\, (0,s) \mid s < p^* \,\}, \tag{A.25}$$

$$= \{0\} \times (-\infty, 0). \tag{A.26}$$

Note that $\mathcal{B}$ is also convex, and that the sets $\mathcal{A}$ and $\mathcal{B}$ do not intersect. We can then invoke the separating hyperplane theorem to show that there exists a $(\tilde{\lambda}, \mu) \neq 0$ and a α that defines a hyperplane which separates the two sets, i.e.,

$$(u,t) \in \mathcal{A} \implies \tilde{\lambda} u + \mu t \ge \alpha \tag{A.27}$$

$$(u,s) \in \mathcal{B} \implies \tilde{\lambda} u + \mu s \le \alpha \tag{A.28}$$

In (A.27), since both u and t are unbounded above, we must have $\tilde{\lambda} \ge 0$ and $\mu \ge 0$. Furthermore, in (A.28), since $s < p^*$, we have that $\mu p^* \le \alpha$. Combining both then gives us that for all x,

$$0 = p^* = \mu p^* \le \alpha \le \tilde{\lambda} g(x,z). \tag{A.29}$$

Minimizing the RHS over x then maximizing over $\tilde{\lambda}$ then gives us that

$$p^* \le \min_x \tilde{\lambda} g(x,z) \le \max_{\lambda} \min_x \lambda g(x,z) = d^*. \tag{A.30}$$

Finally, by weak duality, we have that

$$p^* \ge d^*. \tag{A.31}$$

Combining the two then allows us to conclude that $p^* = d^*$. □

APPENDIX B
UNDERSTANDING THE ROLE OF z IN THE EFCOCP INNER PROBLEM

In this section, we provide more intuition about the role of z on the learned policy π and the learned value function V^π. We first restate the EFCOCP inner problem below.

$$\tilde{J}^\pi(x_0, z) \coloneqq \max\left\{\max_{k\geq 0} h(x_k), \sum_{k=0}^{\infty} l(x_k) - z\right\}. \tag{B.1}$$

As $z \to -\infty$, the cost (i.e., stability) related term dominates the $\max$. Consequently, we should see that the optimal policy will prioritize stability. On the other hand, as $z \to \infty$, the constraint (i.e., safety) related term dominates the $\max$. In this case, the optimal policy will prioritize safety. Moreover, if the optimal policy is safe under the unconstrained minimizer, i.e.,

$$h_{\max} \coloneqq \max_{k\geq 0} h(x_k) \leq 0, \tag{B.2}$$

then the second term will be larger than the first. Since the second term is non-negative, we have that for any $z \in (-h_{\max}, 0]$,

$$\sum_{k=0}^{\infty} l(x_k) - z \geq -z > h_{\max} \tag{B.3}$$

Consequently, for such a choice of z,

$$\tilde{J}^\pi(x_0, z), = \max\left\{\max_{k\geq 0} h(x_k), \sum_{k=0}^{\infty} l(x_k) - z\right\}, \tag{B.4}$$

$$= \max\left\{h_{\max}, \sum_{k=0}^{\infty} l(x_k) - z\right\}, \tag{B.5}$$

$$= \sum_{k=0}^{\infty} l(x_k), \tag{B.6}$$

and we recover the unconstrained optimizer.

We now compare the policy rollouts for different values of z on different systems. The policy rollouts for the 1D double-integrator system are shown in Figure 1. Note that for $z = 0$, states whose unconstrained minimizer are safe follow the unconstrained optimal trajectory and converge to the goal region. As z increases, the policy focuses more on constraint satisfaction. Consequently, trajectories that were originally unsafe (red) become safe (blue). However, as z increases further, the policy focuses too much on minimizing constraint function (i.e., $\max_{k\geq 0} h(x_k)$) and converges to the minimizer of h instead of the goal region (olive).

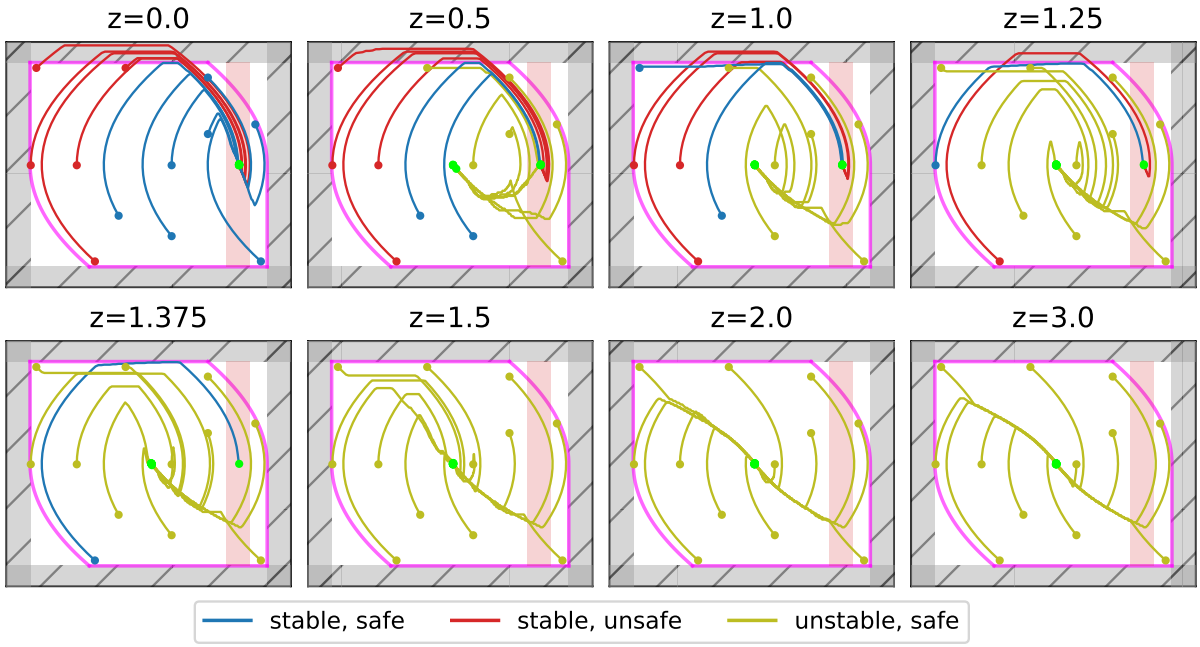
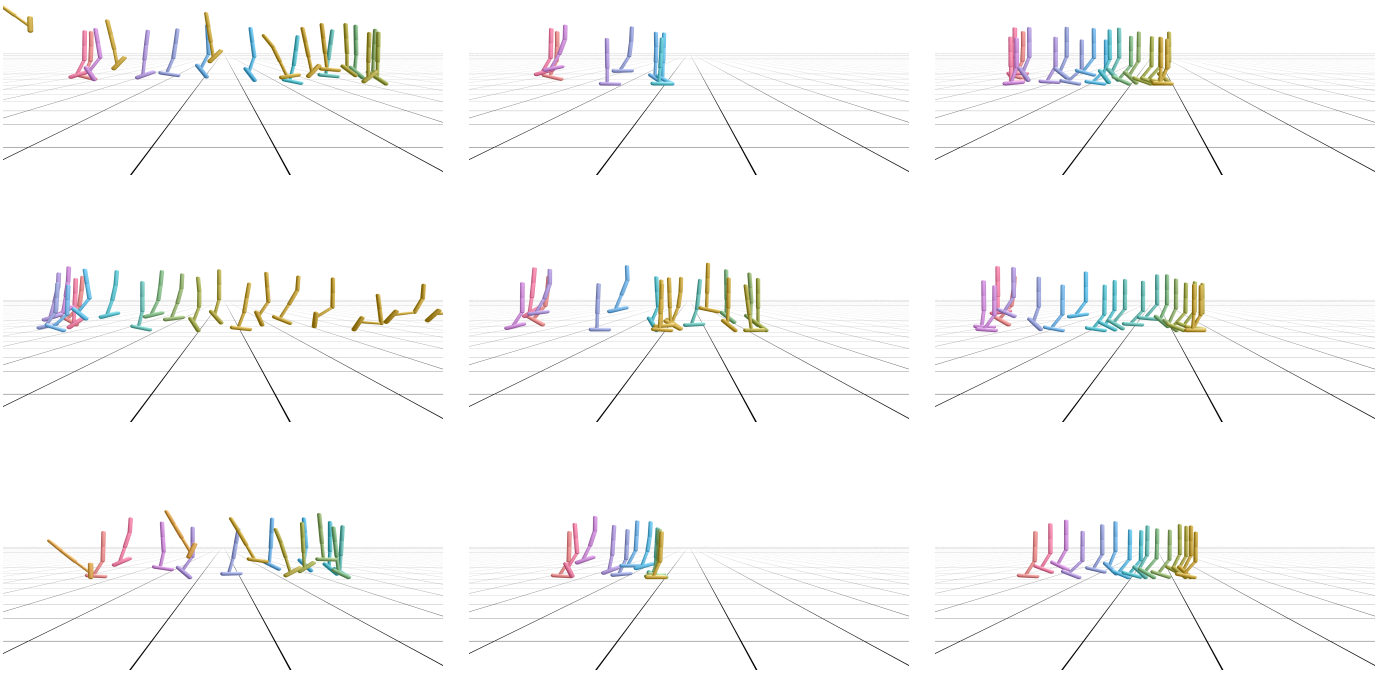


Fig. 1: Comparison of the policy $\pi(\cdot, z)$ for different values of z on the 1D double integrator.



(a) $z = 0.0$ (b) $z = 0.75$ (c) $z = 2.8$

Fig. 2: Comparison of the policy $\pi(\cdot, z)$ for $z = 0$, $z = 0.75$ and $z = 2.8$ respectively from left to right on the Hopper system. The direction of time follows the colors red, purple, blue, green, yellow. Each row represents a different initial condition. The policy for $z = 0.0$ is too aggressive and eventually topples, violating the safety constraints. In contrast, the policy for $z = 2.8$ prioritizes safety by keeping the torso vertical, but hops very slowly in doing so and also overshoots the goal region (compare with $z = 2.8$). Taking z to be a value between these two extremes (e.g., $z = 0.75$) stabilizes to the goal while maintaining safety. By training a policy π that is conditioned on z, we can maintain safety and obtain a stabilizing controller despite π being suboptimal by learning a proper value of z^*.

We next show the policy rollouts on the Hopper system for different values of z in Figure 2. Again, we can see that larger values of z correlates to higher emphasis on safety. In the case of Hopper, note that the optimal unconstrained optimizer should be able to maintain safety. However, despite the learned policy being suboptimal (and hence unsafe), we are still able to obtain a safe final policy by using $z > 0$.

APPENDIX C
DISCOUNTING IN EFPPO

As noted in the main text, we randomly sample z from $[z, z_{\max}]$ when solving the inner problem of EFCOCP, where $z_{\max}$ is an upper bound of the total cost under the optimal policy π^*, i.e.,

$$z_{\max} \geq \sum_{k=0}^{\infty} l(x_k), \tag{C.1}$$

for any trajectory $\{x_k\}_{k=0}^{\infty}$. However, if the system under the optimal policy does not stabilize to the zero-set of l fast enough (or not at all) due to lack of controllability, then this may be infinite. While this is not a problem for the solution of the optimization problem if the system is not controllable from x_0, it is problematic when we apply reinforcement learning to the problem and learn a neural network that approximates the policy value function V^π. Such a term will dominate the loss function when training V^π.

To alleviate this, we apply a small discount factor $\gamma \in (0, 1)$, taken to be 0.97 in all of our experiments. Consequently, we now consider the *discounted* EFCOCP inner problem, where the cost function $\tilde{J}^\pi$ now takes the form

$$\tilde{J}(x_0, z) \coloneqq \max\Big\{ \max_{k\geq 0} \gamma^k h(x_k), \sum_{k=0}^{\infty} \gamma^k l(x_k) - z \Big\}. \tag{C.2}$$

The dynamic programming equations are modified correspondingly, which we derive below.

$$\begin{aligned}
\tilde{V}(x_0, z_0) &= \min_{u_{0:\infty}} \max\Big\{ \max_{k\geq 0} \gamma^k h(x_k), \sum_{k=0}^{\infty} \gamma^k l(x_k) - z \Big\}, && \text{(C.3)}\\
&= \min_{u_{0:\infty}} \max\Big\{ h(x_0), \max_{k\geq 1} \gamma^k h(x_k), \sum_{k=1}^{\infty} \gamma^k l(x_k) - \big(z - l(x_0)\big) \Big\}, && \text{(C.4)}\\
&= \min_{u_0} \max\Big\{ h(x_0), \min_{u_{1:\infty}} \max\Big(\max_{k\geq 1} \gamma^k h(x_k), \sum_{k=1}^{\infty} \gamma^k l(x_k) - \big(z - l(x_0)\big) \Big) \Big\}, && \text{(C.5)}\\
&= \min_{u_{0:\infty}} \max\left\{ h(x_0), \gamma \min_{u_{1:\infty}} \max\left(\max_{k\geq 0} \gamma^k h(x_{k+1}), \sum_{k=0}^{\infty} \gamma^k l(x_{k+1}) - \frac{z - l(x_0)}{\gamma} \right) \right\}, && \text{(C.6)}\\
&= \min_{u_{0:\infty}} \max\left\{ h(x_0), \gamma V\left(x_1, \frac{z - l(x_0)}{\gamma} \right) \right\}, && \text{(C.7)}\\
&= \min_{u_{0:\infty}} \max\left\{ h(x_0), \gamma V\left(x_1, z_1 \right) \right\}, && \text{(C.8)}
\end{aligned}$$

where the "dynnamics" for z now read

$$z_{k+1} = \frac{z_k - l(x_k)}{\gamma}. \tag{C.9}$$

Following this, the policy value function V^π and policy action-value function Q^π used for EFPPO are modified accordingly. With the discounted formulation, V^π is now finite assuming $h(x_k)$ does not explode and $l(x_k)$ does not grow faster than γ^k, which is satisfied in most practical problems where the system has enough control authority. While we can find $z_{\max}$ analytically, in practice $z_{\max}$ is found empirically by running the inner loop of EFPPO for several iterations and then taking $z_{\max}$ to be a constant multiple (e.g., 1.5) of the largest value of $\sum_{k=0}^{\infty} \gamma^k l(x_k)$ seen so far. Since the initial policy is generally worse than π^* (i.e., has larger cost), this procedure yields a conservative over-estimate of $z_{\max}$ that we have found to be robust. In our experiments, this procedure only needs to be performed once for every new task to set $z_{\max}$ and does not require any tuning afterwards.

APPENDIX D
SIMULATION DETAILS

Details for the simulation environments used are provided below.

D1 1D Double-Integrator

States			Controls		
Index	Symbol	Description	Index	Symbol	Description
0	p	Position	0	a	Acceleration
1	v	Velocity			

TABLE I: States and Controls for the 1D Double-Integrator

The 1D Double-Integrator is a system with 2 state and 1 control dimensions (see Table I). The dynamics are linear and take the form

$$\begin{bmatrix} p_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_k \\ v_k \end{bmatrix} + \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \end{bmatrix} \begin{bmatrix} a_k \end{bmatrix} \tag{D.1}$$

for timestep Δt. We use $\Delta t = 0.025$.

The control constraints are box constraints within $[-1, 1]$

$$|a| \leq 1. \tag{D.2}$$

The state constraints (which define the avoid set $\mathcal{A}$) are

$$|p| \leq 1, \quad |v| \leq 1 \tag{D.3}$$

To represent $\mathcal{A}$, we define $h(\mathbf{x}) = \max h_1(\mathbf{x}), h_2(\mathbf{x})$, where

$$h_1(\mathbf{x}) \coloneqq |p| - 1, \quad h_2(\mathbf{x}) \coloneqq |v|^3 - 1. \tag{D.4}$$

The goal set $\mathcal{G}$ is defined as the region

$$\mathcal{G} \coloneqq \{ \mathbf{x} \mid p \in [0.65, 0.85] \}, \tag{D.5}$$

which we represent via the cost function l as

$$l(x) \coloneqq \big[|p - 0.75| - 0.1\big]^+. \tag{D.6}$$

D2 2D Single-Integrator with Sector Obstacle

States			Controls		
Index	Symbol	Description	Index	Symbol	Description
0	p_x	Position along x	0	v_n	Velocity along the normal to the origin
1	p_y	Position along y	1	v_t	Velocity along the tangent to the origin

TABLE II: States and Controls for the 2D Single-Integrator

The 2D Single-Integrator is a system with 2 state and 2 control dimensions (see Table II). The continuous-time dynamics are as

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} p_x \\ p_y \end{bmatrix} = \frac{1}{\sqrt{p_x^2 + p_y^2}}\begin{bmatrix} -p_x & -p_y \\ p_y & -p_x \end{bmatrix}\begin{bmatrix} v_n \\ v_t \end{bmatrix}, \tag{D.7}$$

where the denominator is clipped to prevent division by 0. The discrete-time dynamics are obtained by discretizing the above using Euler integration with timestep $\Delta t = 0.05$.

The control constraints are box constraints within $[-1, 1]^2$

$$|v_n| \leq 1, \quad |v_t| \leq 1. \tag{D.8}$$

The state constraints are represented as the set $h(x) = \max(h_0(x), h_1(x)) \leq 0$, where

$$h_0(\mathbf{x}) \coloneqq r - 1, \quad h_1(\mathbf{x}) \coloneqq 0.2(1 - \sqrt{2}) + \sqrt{2}p_x - r. \tag{D.9}$$

where $r \coloneqq \sqrt{p_x^2 + p_y^2}$ denotes the distance to the origin. h_0 defines a circle with radius 1, while h_1 defines the sector obstacle. The goal set $\mathcal{G}$ is defined as a circle at the origin with radius $R = 0.05$, which we represent via the cost function

$$l(\mathbf{x}) \coloneqq [r - 0.05]^+. \tag{D.10}$$

D3 Hopper Stabilization

States			Controls		
Index	Symbol	Description	Index	Symbol	Description
0	p_x	x-coordinate of the torso	0	τ_t	Torque applied to the thigh motor
1	p_z	z-coordinate of the torso	1	τ_l	Torque applied to the leg motor
2	θ	Angle of the torso	2	τ_f	Torque applied to the foot motor
3	θ_t	Joint Angle of the thigh			
4	θ_l	Joint Angle of the leg			
5	θ_f	Joint Angle of the foot			
6	v_x	Velocity of x-coordinate of the torso			
7	v_z	Velocity of z-coordinate of the torso			
8	ω	Angular velocity of the torso			
9	ω_t	Joint Velocity of the thigh			
10	ω_l	Joint Velocity of the leg			
11	ω_f	Joint Velocity of the foot			

TABLE III: States and Controls for the Hopper

The Hopper is a system with 12 state and 3 control dimensions (see Table III) implemented using the Brax [2] simulator. Note that the version of Hopper we use includes p_x. The original Hopper environment from Brax and other simulators such as Mujoco [7] exclude p_x, since the goal is to learn a limit-cycle that is independent of p_x. In Brax, the dynamics are defined using the rigid body equations, which are then discretized using the default integrator settings (Euler integration, $\Delta t = 0.008$). The control constraints are box constraints within $[-1, 1]^3$ taken from the default settings. The state constraints are represented as the set $h(\mathbf{x}) = \max(h_0(\mathbf{x}), h_1(\mathbf{x})) \leq 0$, where

$$h_0(\mathbf{x}) \coloneqq 0.7 - p_z, \quad h_1(\mathbf{x}) \coloneqq |\theta| - 0.2, \tag{D.11}$$

which represent maintaining a minimum height and preventing the torso from tipping over too much, and are taken from the default settings. The goal set $\mathcal{G}$ is defined as the set of states where p_x is within the set $[2.8, 3.0]$. This is represented via the cost function

$$l(\mathbf{x}) \coloneqq [|p_x - 2.9| - 0.1]^+. \tag{D.12}$$

States			Controls		
Index	Symbol	Description	Index	Symbol	Description
0	v_T	Air speed	0	Nz_d	Setpoint for accleration
1	α	Angle of attack	1	Ps_d	Setpoint for stability roll rate
2	β	Angle of sideslip	2	$NypR_d$	Setpoint for side acceleration and yaw rate
3	ϕ	Roll	3	δ_t	Throttle
4	θ	Pitch			
5	ψ	Yaw			
6	P	Roll rate			
7	Q	Pitch rate			
8	R	Yaw rate			
9	p_N	Northward displacement			
10	p_E	Eastward displacement			
11	p_U	Altitude			
12	pow	Engine power lag			
13	Nz	Upward acceleration			
14	Ps	Stability roll rate			
15	$NypR$	Side acceleration and yaw rate			
15	$\mathcal{V}$	Valid mask			

TABLE IV: States and Controls for the F16

The F16 is a system with 17 state and 4 control dimensions (see Table IV) based on [5]. Note that the original system includes only 16 states. We have added the final state $\mathcal{V}$ to prevent the state from exiting the region where the F16 model is numerically accurate. The continuous-time dynamics are defined in [5] is a standard model used in aerospace engineering and described extensively in the textbook by Stevens and Lewis [6]. Notably, the dynamics makes use of look-up tables for aspects such as the engine model and aerodynamic coefficients. The discrete-time dynamics are defined by integrating the continuous-time dynamics using the RK4 integrator with a step size of $\Delta t = 0.05$. Moreover, when the system exits the region where the model is valid, defined as the set

$$\left\{ \mathbf{x} \;\middle|\; \alpha \in [\underline{\alpha}, \overline{\alpha}], \beta \in [\underline{\beta}, \overline{\beta}], \theta \in [-\frac{\pi}{2}, \frac{\pi}{2}] \right\}, \tag{D.13}$$

we set $\mathcal{V}$ to 0 and stop integrating the dynamics to prevent the model from misbehaving. The bounds for α and β are taken as the limits of the aerodynamic data tables, while the bounds for the pitch θ are used to avoid the singularity due to the use of Euler angles.

The control constraints are box constraints defined as

$$Nz_d \in [-10, 15], \quad Ps_d \in [-10, 10], \quad NypR_d \in [-10, 10], \quad \delta_t \in [0, 1] \tag{D.14}$$

The state constraints are represented as the set $h(\mathbf{x}) = \max_i h_i(\mathbf{x}) \leq 0$, where the constraint functions h_i are defined as

$$\text{(Avoid ground and stay below ceiling)} \qquad h_0(\mathbf{x}) \coloneqq \max(0 - p_U, p_U - 1000) \,/\, 200, \tag{D.15a}$$

$$\text{(Keep } \alpha \text{ valid)} \qquad h_1(\mathbf{x}) \coloneqq \max(\underline{\alpha} - \alpha, \alpha - \overline{\alpha}) \,/\, 0.2, \tag{D.15b}$$

$$\text{(Keep } \beta \text{ valid)} \qquad h_2(\mathbf{x}) \coloneqq \max(\underline{\beta} - \beta, \beta - \overline{\beta}) \,/\, 0.2, \tag{D.15c}$$

$$\text{(Keep } \theta \text{ valid)} \qquad h_3(\mathbf{x}) \coloneqq \max(\underline{\theta} - \theta, \theta - \overline{\theta}) \,/\, 0.2, \tag{D.15d}$$

$$\text{(Stay within the flight corridor)} \qquad h_4(\mathbf{x}) \coloneqq \max(-200 - p_E, p_E - 200) \,/\, 50. \tag{D.15e}$$

The goal region $\mathcal{G}$ is defined as the set of states where the altitude p_U is within the set $[50, 150]$. Note that this set is very close to the ground and thus the reason why we call this task "low-altitude flight corridor". We implement this via the cost function

$$l(\mathbf{x}) \coloneqq \max\left(50 - p_U, p_U - 150\right) / 250. \tag{D.16}$$

REFERENCES

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

[2] C. Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax - a differentiable physics engine for large scale rigid body simulation, 2021. URL http://github.com/google/brax.

[3] Gene Grimm, Michael J Messina, Sezai Emre Tuna, and Andrew R Teel. Model predictive control: for want of a local control lyapunov function, all is not lost. *IEEE Transactions on Automatic Control*, 50(5):546–558, 2005.

[4] Lars Grüne, Jürgen Pannek, Lars Grüne, and Jürgen Pannek. *Nonlinear model predictive control*. Springer, 2017.

[5] Peter Heidlauf, Alexander Collins, Michael Bolender, and Stanley Bak. Verification challenges in f-16 ground collision avoidance and other automated maneuvers. In *ARCH@ ADHS*, pages 208–217, 2018.

[6] Brian L. Stevens and Frank L. Lewis. Aircraft Control and Simulation. *Aircraft Engineering and Aerospace Technology*, 76(5), January 2004. ISSN 0002-2667. doi: 10.1108/aeat.2004.12776eae.001. URL https://doi.org/10.1108/aeat.2004.12776eae.001. Publisher: Emerald Group Publishing Limited.

[7] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.