A Appendix

A.1 Proofs of Controller Properties

Proof of Lemma 1 We proceed by induction. By assumption 3, $J^0_{0\to H}(x^j_0) < \infty$. By the definition of $V^{\pi^j}_{\mathcal{G}}$ and $\mathcal{F}^j_{\mathcal{G}}$, $J^j_{0\to H}(x^j_0) < \infty$. Let $J^j_{t\to t+H}(x^j_t) < \infty$ for some $t \in \mathbb{N}$. In the following expressions, we do not explicitly write the MPC problem constraints for clarity. Conditioning on the random variable x^j_t:

$$J^j_{t\to t+H}(x^j_t) = \mathop{\mathbb{E}}_{w^j_{t:t+H-1}}\left[\sum_{k=0}^{H-1} C(x^j_{t+k|t}, \pi^{*,j}_{t+k|t}(x^j_{t+k|t})) + V^{\pi^{j-1}}_{\mathcal{G}}(x^j_{t+H|t})\right] \tag{A.1.1}$$

$$= C(x^j_t, \pi^{*,j}_{t|t}(x^j_t)) + \mathbb{E}_{w^j_{t:t+H-1}}\left[\sum_{k=1}^{H-1} C(x^j_{t+k|t}, \pi^{*,j}_{t+k|t}(x^j_{t+k|t})) + V^{\pi^{j-1}}_{\mathcal{G}}(x^j_{t+H|t})\right] \tag{A.1.2}$$

$$\begin{aligned} &= C(x^j_t, \pi^{*,j}_{t|t}(x^j_t)) \\ &+ \mathop{\mathbb{E}}_{w^j_{t:t+H}}\Bigg[\sum_{k=1}^{H-1} C(x^j_{t+k|t}, \pi^{*,j}_{t+k|t}(x^j_{t+k|t})) + C(x^j_{t+H|t}, \pi^l(x^j_{t+H|t})) \\ &+ V^{\pi^{j-1}}_{\mathcal{G}}(x^j_{t+H+1|t})\Bigg],\ l \in [j-1] \end{aligned} \tag{A.1.3}$$

$$\begin{aligned} &\geq C(x^j_t, \pi^{*,j}_{t|t}(x^j_t)) \\ &+ \mathop{\mathbb{E}}_{w^j_t}\Bigg[\min_{\pi_{t+1:t+H|t+1}} \mathop{\mathbb{E}}_{w^j_{t+1:t+H}}\Bigg[\sum_{k=1}^{H-1} C(x^j_{t+k|t+1}, \pi_{t+k|t+1}(x^j_{t+k|t+1})) \\ &+ C(x^j_{t+H|t+1}, \pi_{t+H|t+1}(x^j_{t+H|t+1})) \\ &+ V^{\pi^{j-1}}_{\mathcal{G}}(x^j_{t+H+1|t+1})\Bigg]\Bigg] \end{aligned} \tag{A.1.4}$$

$$= C(x^j_t, \pi^j(x^j_t)) + \mathop{\mathbb{E}}_{w^j_t}\left[J^j_{t+1\to t+H+1}(x^j_{t+1})|x^j_t\right] \tag{A.1.5}$$

Equation A.1.1 follows from the definition in 5.1.1, equation A.1.3 follows from the definition of $V^{\pi_{j-1}}_{\mathcal{G}}$, which is defined as a point-wise minimum over $\left(L^{\pi^l}_{\mathcal{G}}\right)_{l=0}^{j-1}$. We take a function $L^{\pi^l}_{\mathcal{G}}$ that is active at $x^j_{t+H|t}$ and apply its definition to expand it and then replace $L^{\pi^l}_{\mathcal{G}}$ with $V^{\pi^{j-1}}_{\mathcal{G}}$ in the expansion. The inner expectation in equation A.1.4 conditions on the random variable x^j_{t+1}, and the outer expectation integrates it out. The inequality in A.1.4 follows from the fact that $[\pi^{*,j}_{t+1|t}, \ldots, \pi^{*,j}_{t+H-1|t}, \pi^{j-1}]$ is a possible solution to (A.1.4). Equation A.1.5 follows from the definition in equation 5.1.1.

We have shown that $J^j_{t\to t+H}(x^j_t) < \infty \implies \mathop{\mathbb{E}}_{w^j_t}\left[J^j_{t+1\to t+H+1}(x^j_{t+1|t})\right] < \infty$. So:

$$\mathop{\mathbb{E}}_{w^j_{0:t-1}}\left[J^j_{t\to t+H}(x^j_t)\right] < \infty \implies \mathop{\mathbb{E}}_{w^j_{0:t-1}}\left[\mathop{\mathbb{E}}_{w^j_t}\left[J^j_{t+1\to t+H+1}(x^j_{t+1|t})\right]\right] \tag{A.1.6}$$

$$= \mathop{\mathbb{E}}_{w^j_{0:t}}\left[J^j_{t+1\to t+H+1}(x^j_{t+1})\right] < \infty \tag{A.1.7}$$

By induction, $\mathop{\mathbb{E}}_{w^j_{0:t-1}}[J^j_{t\to t+H}(x^j_t)] < \infty \ \forall t \in \mathbb{N}$. Therefore, the controller is feasible at iteration j. □

Proof of Lemma 2 By Lemma 1 and Assumption 1, $\forall L \in \mathbb{N}$,

$$\mathop{\mathbb{E}}_{w^j_{1:L-1}}\left[\sum_{k=0}^{L-1} C(x^j_k, \pi^j(x^j_k)) + J^j_{L\to L+H}(x^j_L)\right] \le J^j_{0\to H}(x^j_0) \tag{A.1.8}$$

$$\implies \mathop{\mathbb{E}}_{w^j_{1:L-1}}\left[J^j_{L\to L+H}(x^j_L)\right] \le J^j_{0\to H}(x^j_0) - \mathop{\mathbb{E}}_{w^j_{1:L-1}}\left[\sum_{k=0}^{L-1} C(x^j_k, \pi^j(x^j_k))\right] \tag{A.1.9}$$

$$\le J^j_{0\to H}(x^j_0) - \varepsilon \sum_{k=0}^{L-1} P(x^j_k \notin \mathcal{G}) \tag{A.1.10}$$

Line A.1.10 follows from rearranging A.1.8 and applying assumption 1. Because $\mathcal{G}$ is robust control invariant by assumption 2, $x_t \in \mathcal{G} \implies x_{t+k} \in \mathcal{G}\ \forall k \ge 0$. Now, assume $\lim_{k\to\infty} P(x^j_k \notin \mathcal{G})$ does not exist or is nonzero. This implies that $P(x^j_k \notin \mathcal{G}) \ge \delta > 0$ infinitely many times. By the Archimedean principle, the RHS of A.1.10 can be driven arbitrarily negative, which is impossible. By contradiction, $\lim_{k\to\infty} P(x^j_k \notin \mathcal{G}) = 0$. □

Proof of Theorem 1 Let $j \in \mathbb{N}$

$$J^j_{0\to H}(x_0) \ge C(x_0, u_0) + \mathop{\mathbb{E}}_{w^j_0}\left[J^j_{1\to H+1}(x^j_1)\right] \tag{A.1.11}$$

$$\ge \mathop{\mathbb{E}}_{w^j}\left[\sum_{t=0}^{\infty} C(x^j_t, \pi^j(x^j_t))\right] + \lim_{t\to\infty} \mathop{\mathbb{E}}_{w^j_{0:t-1}}\left[J^j_{t\to t+H}(x^j_t)\right] \tag{A.1.12}$$

$$= \mathop{\mathbb{E}}_{w^j}\left[\sum_{t=0}^{\infty} C(x^j_t, \pi^j(x^j_t))\right] + \lim_{t\to\infty} \mathop{\mathbb{E}}_{\mathbb{1}\{x^j_t \notin \mathcal{G}\}}\left[\mathop{\mathbb{E}}_{w^j_{0:t-1}}\left[J^j_{t\to t+H}(x_t)|\mathbb{1}\{x^j_t \notin \mathcal{G}\}\right]\right] \tag{A.1.13}$$

$$= \mathop{\mathbb{E}}_{w^j}\left[\sum_{t=0}^{\infty} C(x^j_t, \pi^j(x^j_t))\right] + \lim_{t\to\infty} \mathop{\mathbb{E}}_{w^j_{0:t-1}}\left[J^j_{t\to t+H}(x^j_t)|x^j_t \notin \mathcal{G}\right] P(x^j_t \notin \mathcal{G}) \tag{A.1.14}$$

$$\geq \mathop{\mathbb{E}}_{w^j}\left[\sum_{t=0}^{\infty} C(x_t^j, \pi^j(x_t^j))\right] + \lim_{t\to\infty} \varepsilon P(x_t^j \notin \mathcal{G}) \tag{A.1.15}$$

$$= \mathop{\mathbb{E}}_{w^j}\left[\sum_{t=0}^{\infty} C(x_t^j, \pi^j(x_t^j))\right] = J^{\pi^j}(x_0) \tag{A.1.16}$$

Equations A.1.11 and A.1.12 follow from repeated application of Lemma 1 (A.1.5). Equation A.1.13 follows from iterated expectation, equation A.1.14 follows from the cost function assumption 1. Equation A.1.15 follows again from assumption 1 (incur a cost of at least ε for not being at the goal at time t). Then, Equation A.1.16 follows from Lemma 2. Using the above inequality with the definition of $J^{\pi^j}(x_0)$,

$$J^j_{0\to H}(x_0) \geq J^{\pi^j}(x_0) = \mathop{\mathbb{E}}_{w^j_{0:H-1}}\left[\sum_{t=0}^{H-1} C(x_t^j, \pi^j(x_t)) + V_{\mathcal{G}}^{\pi^j}(x_H^j)\right] \tag{A.1.17}$$

$$\geq \mathop{\mathbb{E}}_{w^j_{0:H-1}}\left[\sum_{t=0}^{H-1} C(x_t^j, \pi^{*,j}_{t|0}(x_{t|0})) + V_{\mathcal{G}}^{\pi^j}(x_{H|0})\right] = J^{j+1}_{0\to H}(x_0) \tag{A.1.18}$$

$$\geq J^{\pi^{j+1}}(x_0) \tag{A.1.19}$$

Equation A.1.17 follows from equation A.1.16, equation A.1.18 follows from taking the minimum over all possible H-length sequences of policies in the policy class Π. Equation A.1.19 follows from equation A.1.16. By induction, this proves the theorem.

Note that this also implies convergence of $(J^{\pi^j}(x_0))_{j=0}^{\infty}$ by the Monotone Convergence Theorem. □

Proof of Lemma 3 The proof is identical to [5]. Because $\mathcal{SS}^j_{\mathcal{G}}$ is an increasing sequence of sets, $\mathcal{F}^j_{\mathcal{G}}$ is also an increasing sequence of sets by definition. □

A.2 Adjustable Boundary Condition LMPC Implementation Details

Solving the MPC Problem As in [1], we sample a fixed population size of action sequences at each iteration of CEM from a truncated Gaussian. These action sequences are simulated over a known model of the system dynamics and then the sampling distribution for the next iteration is updated based on the lowest cost sampled trajectories. For the cross entropy method we build off of the implementation in [2]. Precisely, at each timestep in a trajectory, a conditional Gaussian is initialized with the mean based on the final solution for the previous timestep and some fixed variance. Then, at each iteration of CEM, pop_size action sequences of plan_hor length are sampled from the conditional Gaussian, simulated over a model of the system dynamics, and then the num_elites samples with the lowest sum cost are used to refit the mean and variance of the conditional Gaussian distribution for the next iteration of CEM. This process is repeated num_iters times. The sum cost of an action sequence is computed by summing up the task cost function at each transition in the resulting simulated trajectory and then adding a large penalty for each constraint violating state in the simulated trajectory and

an additional penalty if the terminal state in the simulated trajectory does not have sufficient density under $\rho_{\mathcal{G}}$. For all experiments, we add a $1e6$ penalty for violating terminal state constraints and a $1e8$ penalty for violating task constraints. In practice to accelerate domain expansion to x^*, when selecting initial states x_S^j from $\bigcup_{k=0}^{j} \tilde{\mathcal{S}}\tilde{\mathcal{S}}^k$, we sort states in the safeset under $C_E^j(x)$ and use this to choose x_S^j close to x^* under $C_E^j(x)$. Note that this choice does not impact any of the theoretical guarantees.

Value Function We represent each member of the probabilistic ensemble of neural networks used to approximate $L^{\pi^j}(x)$ with a neural network with 3 hidden layers, each with 500 hidden units. We use swish activations, and update weights using the Adam Optimizer with learning rate 0.001. We use 10 epochs to learn the weights for $L^{\pi^j}(x)$.

Start State Expansion We again perform trajectory optimization using the cross entropy method and for each experiment use the same pop_size, num_elites, num_iters parameters as for solving the MPC problem. Costs for action sequences are computed by summing up $C_E^j(x)$ evaluated at each state x in the corresponding simulated trajectory, and the same mechanism is used for enforcing the terminal state constraint and task constraints as for solving the MPC problem.

A.3 Experiment Specific Parameters

Pointmass Navigation

Environment Details: We use $\psi = 0.2$ and $\sigma = 0.05$ in all experiments in this domain. Demonstration trajectories are generated by guiding the robot past the obstacle along a very suboptimal hand-tuned trajectory for the first half of the trajectory before running LQR with clipped actions on a quadratic approximation of the true cost. Gaussian noise is added to the demonstrator controller. The task horizon is set to $T = 50$.

Task Controller MPC Parameters: For the single start, single goal set case we use popsize = 400, num_elites = 40, cem_iters = 5, and plan_hor = 15. For all start state expansion experiments, we utilize the same popsize, num_elites, and cem_iters but utilize plan_hor = 20. For experiments we utilize $\alpha = 2$ for the kernel width parameter for density model $\rho_\alpha^{\mathcal{G}}$.

Start State Expansion Parameters: We utilize $H' = H - 5$ for all experiments (trajectory optimization horizon for exploration policy).

SAVED Baseline Experimental Parameters: We supply SAVED with 100 demonstrations generated by the same demonstration policy as for ABC-LMPC. We utilize $\alpha = 3$ and utilize the implementation from [1]. Both the value function and dynamics for SAVED are represented with a probabilistic ensemble of 5 neural networks with 3 hidden layers of 500 hidden units each. We use swish activations, and update weights using the Adam Optimizer with learning rate 0.001.

7-Link Reacher Arm

Environment Details: We use $\sigma = 0.03$ for all experiments. The state space consists of the 7 joint angles. Each link is of 1 unit in length and the goal is to control the end effector position to a 0.5 radius circle in $\mathbb{R}^2$ centered at $(3, -3)$. We do not model self-collisions but also include a circular obstacle of radius 1 in the environment which the kinematic chain must navigate around. Collisions with the obstacle are checked by computing the minimum distance between each link in the kinematic chain and the center of the circular obstacle and determining whether any link has a minimum distance from the center of the obstacle that is less than the radius of the obstacle. The task horizon is set to $T = 50$. We build on the implementation provided through [32].

Task Controller MPC Parameters: For the single start, single goal set case we use popsize = 400, num_elites = 40, cem_iters = 5, and plan_hor = 15. For all start state expansion experiments, we utilize the same popsize, num_elites, and cem_iters but utilize plan_hor = 20. For experiments we utilize $\alpha = 0.5$ for the kernel width parameter for density model $\rho_\alpha^{\mathcal{G}}$.

Start State Expansion Parameters: We utilize $H' = H - 5$ for all experiments (trajectory optimization horizon for exploration policy).

SAVED Baseline Experimental Parameters: We supply SAVED with 100 demonstrations generated by the same demonstration policy as for ABC-LMPC. We utilize $\alpha = 0.5$ and utilize the implementation from [1]. Both the value function and dynamics for SAVED are represented with a probabilistic ensemble of 5 neural networks with 3 hidden layers of 500 hidden units each. We use swish activations, and update weights using the Adam Optimizer with learning rate 0.001.

Inverted Pendulum

Environment Details: We use $\sigma = 0.5$ for all experiments. The robot consists of a single link and can exert a torque to rotate it. The state space consists of the angle and angular velocity of the pendulum. Note that there are only stable orientations, the upright orientation and downward orientation for this task, and thus for a goal set to be robust control invariant, it will likely need to be defined around the neighborhood of these orientations. The task horizon is set to $T = 40$. We define $\mathcal{G}_1$ as the goal set centered around the downward orientation and $\mathcal{G}_2$ as the goal set centered around the upright orientation. Precisely, inclusion in $\mathcal{G}_1$ is determined by determining whether the orientation of the pendulum is within 45 degrees of the downward orientation. Similarly, inclusion in $\mathcal{G}_2$ is determined by determining whether the orientation of the pendulum is within 45 degrees of the upward orientation.

Task Controller MPC Parameters: We utilize popsize = 600, num_elites = 40, cem_iters = 5, and plan_hor = 15. For experiments we utilize $\alpha = 2$ for the kernel width parameter for density model $\rho_\alpha^{\mathcal{G}}$.

Start State Expansion Parameters: We utilize $H' = H$ for all experiments (trajectory optimization horizon for exploration policy).

A.4 Controller Domain Expansion Strategy

Here we discuss how the controller domain can be expanded when the safe set and value function are updated based on samples from the exploration policy. To approximately expand $\bigcup_{k=0}^{j} \mathcal{SS}_{\mathcal{G}}^{k}$, we can again solve the following 1-step trajectory optimization problem:

$$\begin{aligned} \pi_{E,0:H'-1}^{j} = \underset{\pi_{0:H'-1} \in \Pi^{H'}}{\operatorname{argmin}} \quad & \underset{w_{0:H'-2}^{j}}{\mathbb{E}} \left[\sum_{i=0}^{H'-1} C_E^j(x_i^j, \pi_i(x_i^j)) \right] \\ \text{s.t.} \quad & x_{i+1}^j = f(x_i^j, \pi_i(x_i^j), w_i) \ \forall i \in \{0, \dots, H'-1\} \\ & x_{H'}^j \in \bigcup_{k=0}^{j-1} \mathcal{SS}_{\mathcal{G}}^{k}, \ \forall w_{0:H'-2} \in \mathcal{W}^{H'-1} \\ & x_{0:H'}^j \in \mathcal{X}^{H'+1}, \ \forall w_{0:H'-2} \in \mathcal{W}^{H'-1} \end{aligned} \tag{A.4.20}$$

For all $x_S^j \in \mathcal{SS}_{\mathcal{G}}^{j-1}$, the states $\bigcup_{k=0}^{H'} \mathcal{R}_k^{\pi_{E,0:H'-1}^{j}}(x_S^j) \cup \bigcup_{k=1}^{\infty} \mathcal{R}_k^{\pi^j}(\mathcal{R}_{H'}^{\pi_{E,0:H'-1}^{j}}(x_S^j))$ are added to $\mathcal{SS}_{\mathcal{G}}^{j}$. The second union is included to define the value function for the composition of π^j and $\pi_{E,0:H'-1}^{j}$. This is analogous to running the exploration policy followed by running the task-directed policy π^j. Denoting the safe set where π^j is executed as $\mathcal{SS}_{\mathcal{G}}^{\pi^j} = \bigcup_{k=1}^{\infty} \mathcal{R}_k^{\pi^j}(\mathcal{R}_{H'}^{\pi_{E,0:H'-1}^{j}}(\mathcal{SS}_{\mathcal{G}}^{j-1})) \cup \bigcup_{k=1}^{\infty} \mathcal{R}_k^{\pi^j}(\mathcal{SS}_{\mathcal{G}}^{j-1})$, we redefine $L_{\mathcal{G}}^{\pi^j}$ as:

$$L_{\mathcal{G}}^{\pi^j}(x) = \begin{cases} \underset{w}{\mathbb{E}} \left[C(x, \pi^j(x)) + L_{\mathcal{G}}^{\pi^j}(f(x, \pi^j(x), w)) \right] & x \in \mathcal{SS}_{\mathcal{G}}^{\pi^j} \setminus \mathcal{G} \\ \underset{w}{\mathbb{E}} \left[C(x, \pi_{E,0:H'-1}^{j}(x)) + L_{\mathcal{G}}^{\pi^j}(f(x, \pi_{E,0:H'-1}^{j}(x), w)) \right] & x \in \mathcal{SS}_{\mathcal{G}}^{j} \setminus \mathcal{SS}_{\mathcal{G}}^{\pi^j} \\ 0 & x \in \mathcal{G} \\ +\infty & x \notin \mathcal{SS}_{\mathcal{G}}^{j} \end{cases} \tag{A.4.21}$$

This means that trajectories from the exploration policy can spend more time outside of the safe set. In either case, the safe set remains robust control invariant.

Thus, each iteration j is split into two phases. In the first phase, π^j is executed and in the second phase, $\pi_{E,0:H'-1}^{j}$ is executed. This procedure provides a simple algorithm to expand the policy's domain $\mathcal{F}_{\mathcal{G}}^{j}$ while still maintaining its theoretical properties.