Signal Temporal Logic-Guided Apprenticeship Learning
Supplemental Document

Aniruddh G. Puranic, Jyotirmoy V. Deshmukh and Stefanos Nikolaidis

Abstract— **In this document, we cover the quantitative semantics of Signal Temporal Logic (STL) in Appendix I, theory related to affine transformations in rewards in Appendix II, and all details of experiments in Appendix III.**

Appendix I
Signal Temporal Logic

Definition 1.1 (Quantitative Semantics): Given an algebraic structure $(\oplus, \otimes, \top, \bot)$, we define the quantitative semantics for an arbitrary signal $\mathbf{x}$ against an STL formula φ at time t as in Table I.

TABLE I
Quantitative Semantics of STL

φ	$\rho(\varphi, \mathbf{x}, t)$
true/*false*	$\top/\bot$
μ	$f(\mathbf{x}(t))$
$\neg\varphi$	$-\rho(\varphi, \mathbf{x}, t)$
$\varphi_1 \wedge \varphi_2$	$\otimes(\rho(\varphi_1, \mathbf{x}, t), \rho(\varphi_2, \mathbf{x}, t))$
$\varphi_1 \vee \varphi_2$	$\oplus(\rho(\varphi_1, \mathbf{x}, t), \rho(\varphi_2, \mathbf{x}, t))$
$\mathbf{G}_I(\varphi)$	$\otimes_{\tau \in t+I}(\rho(\varphi, \mathbf{x}, \tau))$
$\mathbf{F}_I(\varphi)$	$\oplus_{\tau \in t+I}(\rho(\varphi, \mathbf{x}, \tau))$
$\varphi \mathbf{U}_I \psi$	$\oplus_{\tau_1 \in t+I}(\otimes(\rho(\psi, \mathbf{x}, \tau_1), \otimes_{\tau_2 \in [t, \tau_1)}(\rho(\varphi, \mathbf{x}, \tau_2)))$

A signal satisfies an STL formula φ if it is satisfied at time $t = 0$. Intuitively, the quantitative semantics of STL represent the numerical distance of "how far" a signal is away from the signal predicate. For a given requirement φ, a demonstration or policy d that satisfies it is represented as $d \models \varphi$ and one that does not, is represented as $d \not\models \varphi$. In addition to the Boolean satisfaction semantics for STL, various researchers have proposed quantitative semantics for STL, [1], [2] that compute the degree of satisfaction (or *robust satisfaction values*) of STL properties by traces generated by a system. In this work, we use the following interpretations of the STL quantitative semantics: $\top = +\infty$, $\bot = -\infty$, and $\oplus = \max$, and $\otimes = \min$, as per the original definitions of robust satisfaction proposed in [1], [3].

Appendix II
Derivations and Proofs

As mentioned in the main paper, we show that applying affine transformations to the reward function do not change the optimal policy. Particularly, we are concerned with scaling and shifting the rewards by a constant factor.

Lemma 2.1: The optimal policy is invariant to affine transformations in the reward function.

Proof: [Proof Sketch] From [4], we have the definition of the Q function as follows, for the untransformed reward function R:

$$Q(s,a) \doteq \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \cdot R(s,a)_{t+k+1} | S_t = s, A_t = a\right] \quad (1)$$

$$Q(s,a) \doteq R(s,a) + \gamma \sum_{s'} P(s,a,s') \max_{a'} Q(s',a') \quad (2)$$

We consider two cases of reward function affine transformations in our work: (a) scaling by a positive constant and (b) shifting by a constant. In both these cases, our objective is to express the new Q function in terms of the original. Note that we abbreviate $R(s,a)$ to just R for simplicity.

Case (a): Scaling R by a positive constant: Let the scaled reward function be defined as $R' = c \cdot R, c > 0$. The new Q function is then

$$Q'(s,a) \doteq \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \cdot R'_{t+k+1} | S_t = s, A_t = a\right]$$
$$Q'(s,a) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \cdot c \cdot R_{t+k+1} | S_t = s, A_t = a\right]$$
$$Q'(s,a) = c \cdot \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \cdot R_{t+k+1} | S_t = s, A_t = a\right]$$
$$Q'(s,a) = c \cdot Q(s,a)$$

Thus we see that the new Q function scales with the scaling constant.

From Equation 2 and by later substituting for Q' from the above result, we have,

$$Q'(s,a) \doteq R'(s,a) + \gamma \sum_{s'} P(s,a,s') \max_{a'} Q'(s',a')$$
$$c \cdot Q(s,a) = c \cdot R(s,a) + \gamma \sum_{s'} P(s,a,s') \max_{a'}(c \cdot Q(s',a'))$$
$$c \cdot Q(s,a) = c \cdot R(s,a) + c\gamma \sum_{s'} P(s,a,s') \max_{a'} \cdot Q(s',a')$$
$$Q(s,a) = R(s,a) + \gamma \sum_{s'} P(s,a,s') \max_{a'} \cdot Q(s',a')$$

Thus the Bellman equation holds indicating that the policy is invariant to scaling by a positive constant.

Case (b): Shifting R by a constant: Let the shifted reward function be defined as $R' = R + c$. The new Q

The authors are with the Department of Computer Science, University of Southern California, USA. Email: {puranic, jdeshmuk, nikolaid}@usc.edu

function is then

$$
\begin{aligned}
Q'(s,a) &\doteq \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \cdot R'_{t+k+1} | S_t = s, A_t = a\right] \\
Q'(s,a) &= \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \cdot (R_{t+k+1} + c) | S_t = s, A_t = a\right] \\
Q'(s,a) &= \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \cdot R_{t+k+1} | S_t = s, A_t = a\right] + \sum_{k=0}^{\infty} \gamma^k c \\
Q'(s,a) &= Q(s,a) + \frac{c}{1-\gamma}
\end{aligned}
$$

Thus we see that the new Q values get shifted by the constant.

From Equation 2 and by later substituting for Q' from the above result, we have,

$$
\begin{aligned}
Q'(s,a) &\doteq R'(s,a) + \gamma \sum_{s'} P(s,a,s') \max_{a'} Q'(s',a') \\
Q(s,a) + \frac{c}{1-\gamma} &= R(s,a) + c \\
&\quad + \gamma \sum_{s'} P(s,a,s') \max_{a'} \left(Q(s',a') + \frac{c}{1-\gamma} \right) \\
Q(s,a) + \frac{c}{1-\gamma} &= R(s,a) + c \\
&\quad + \gamma \sum_{s'} P(s,a,s') \max_{a'} Q(s',a') \\
&\quad + \gamma \sum_{s'} P(s,a,s') \frac{c}{1-\gamma} \\
Q(s,a) + \frac{c}{1-\gamma} &= R(s,a) + \gamma \sum_{s'} P(s,a,s') \max_{a'} Q(s',a') \\
&\quad + c + \frac{c\gamma}{1-\gamma} \\
Q(s,a) &= R(s,a) + \gamma \sum_{s'} P(s,a,s') \max_{a'} \cdot Q(s',a')
\end{aligned}
$$

Thus the Bellman equation holds indicating that the policy is invariant to shifting by a constant.

■

Therefore, any combination of scaling or shifting does not affect the optimal policy in our work. Similarly, the optimal policy is shown to be invariant towards reward shaping with potential functions [5].

APPENDIX III
EXPERIMENT DETAILS

This section describes additional details about the experiments such as the STL task specifications, hyperparameters, training and evaluation results. The hyperparameters for all experiments, barring Frozenlake, are provided in Table II.

A. Task - Discrete-Space Frozenlake

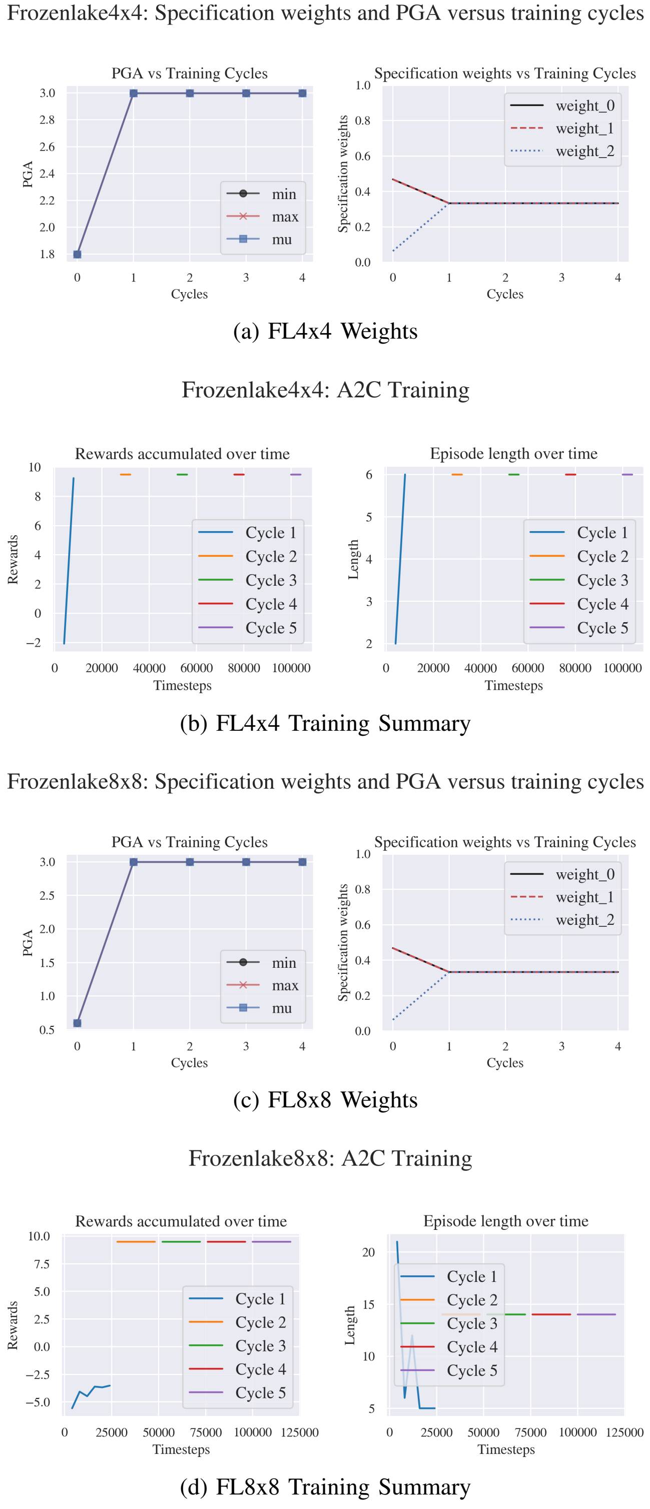
We make use of the $Frozenlake$ (FL) deterministic environments from OpenAI Gym [6] that consist of a grid-world of sizes 4x4 or 8x8 with a reach-avoid task. Informally, the task specifications are (i) eventually reaching the goal, (ii) always avoid unsafe regions and (iii) take as few steps as possible. In these small environments $m = 5$ demonstrations of varying optimality are manually generated. We use A2C as the RL agent and show the training results in Fig. 1. The left figures show the statistics of the rollout PGAs and the evolution of weights over time. The right figures show the rewards accumulated and episode lengths.

We see from the left figures, that initially, the non-uniform weights of specifications correspond to the sub-optimal demonstrations. And over time, the weights all converge to $1/3$ indicating that there are no edges in the final DAG, while the PGAs of rollouts from the final policy are maximum, as hypothesized. Since the environments are deterministic, the final policy achieve a 100% success rate. Since the task can be achieved even with IRL-based methods, we compare the amount of demonstrations required. Under identical conditions, the minimum number of demonstrations used by MCE-IRL are 50 for 4x4 grid and 300 for 8x8 grid. The algorithm in [7] uses over 1000 demonstrations in the 8x8 grid, even though they use temporal logic specifications similar to ours. *This clearly suggests that the choice of the reward inference algorithm plays a significant role in sample complexity*. This is due to the unsafe regions being scattered over the map, requiring the desirable *dense* features to appear very frequently.

B. Task - Reaching Pose

The end-effector of a Franka Panda robot [8] is required to reach the target pose as quickly as possible, the specifications for which are given as: $\varphi_1 := \mathbf{F}(d < \delta)$ and $\varphi_2 := \mathbf{G}(t < T)$, where d is the l^2-norm of the difference between the end-effector and target poses, δ is a small threshold to determine success, and T is the desired time in which the target must be achieved. For evaluation on a more precise environment, we use a surgical robot environment - SURROL [9] that is built on the da Vinci Surgical Robot Kit [10]. In this common surgical task, a needle is placed on a surface and the goal is to move the end-effector towards the needle center. The specifications for this task follow the same template above, however, the threshold is very small, i.e., $\delta = 0.025$, requiring highly precise movements. The reward function was modeled neural network and the RL agent used SAC [11] with hindsight experience replay (HER) [12]. To validate reproducibility, the training and evaluation was performed over 5 random seeds using the same 5 demonstrations. The results for both these environments are shown in Fig. 2. The first column shows the PGA over time or cycles (note the scale of y-axis). The learned policies in both environments achieve have PGA $\approx$ 2 since there are 2 specifications. The second column represents the specification weights. In the surgical task, the final weights are uniform as desired due to the small room for error, while the Panda task has a larger threshold for completion which affects the resolution of the smooth STL semantics, though all tasks are completed successfully. The hyperparameters for both tasks: Panda-Reach and Needle-Reach, were nearly identical. The specifications for both these tasks are:

1) Reaching the target pose: φ_1 :=



(a) FL4x4 Weights

(b) FL4x4 Training Summary

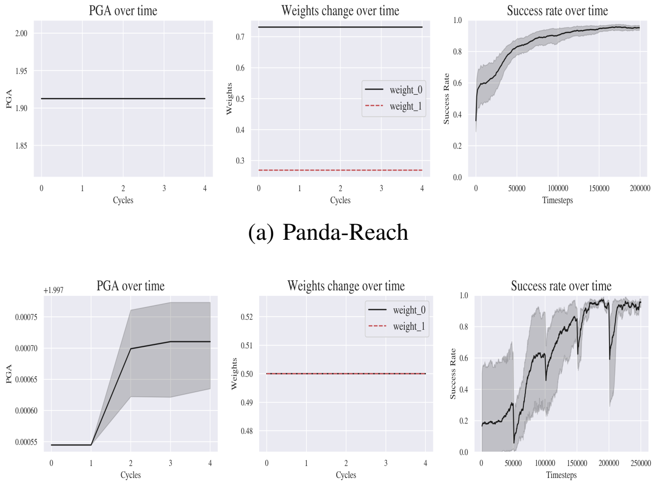
(c) FL8x8 Weights

(d) FL8x8 Training Summary

Fig. 1. Results for the 4x4 and 8x8 Frozenlake environments.

$\mathbf{F}(\|ee_{pose} - target_{pose}\| \leq \delta)$, where ee indicates the end-effector and δ is the threshold used to determine success. For Panda-Reach, $\delta = 0.2$ and for Needle-Reach, $\delta = 0.025$.

2) Reaching the target as quickly as possible: $\varphi_2 := \mathbf{G}(t <= 50)$, where t is the time when the end-effector reaches the target.



(a) Panda-Reach

(b) Needle-Reach

Fig. 2. Summary of training and evaluations for the pose-reaching tasks.

In both tasks, using just 5 demonstrations, AL-STL achieved over 99% mean success rate in both, training (right figures) and evaluations; 5 random seeds were used for evaluations. For Needle-Reach, the baselines [9], [13] that used BC and IRL, required 100 expert demonstrations. It is shown in [13] that, when the number of demonstrations is reduced to just 10, which is still **2x** larger than ours, the success rate drops drastically. For Panda-Reach, the authors of [14] show that imitation learning outperforms adversarial IRL techniques when each method uses 50 demonstrations, though both eventually learn to succeed in the task. This however is still **10x** more than the amount of samples required by our work.

C. Task - Placing Cube

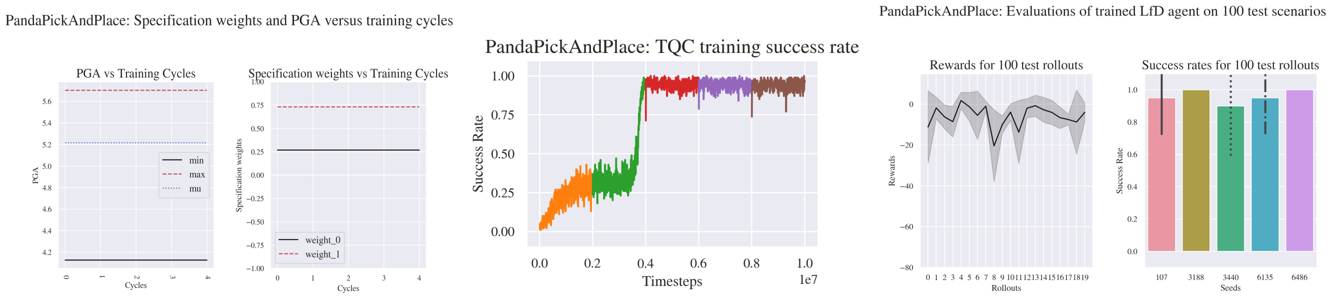
The specifications for both these tasks are:

1) Placing the cube at the target pose: $\varphi_1 := \mathbf{F}(\|cube_{pose} - target_{pose}\| \leq 0.05)$.
2) Reaching the target as quickly as possible: $\varphi_2 := \mathbf{G}(t <= 50)$, where t is the time when the end-effector reaches the target.

The statistics of the PGA shows that is maximum value is ≈ 6 since there are 2 specifications, each scaled by a factor of 3.

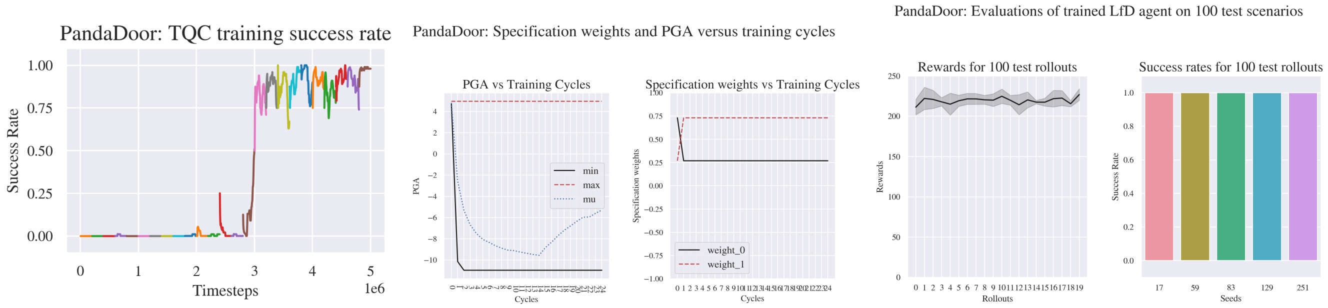
D. Task - Opening Door

The Panda robot uses operational space control to control the pose of the end-effector. The horizon for this task is 500 and the control frequency is 20 Hz. The specifications for both these tasks are:



(a) PGA and Weights (b) RL training summary (c) Success rates on test trials

Fig. 3. Summary of training and evaluations for the Cube-Placing task.



(a) RL training summary (b) PGA and Weights (c) Success rates on test trials

Fig. 4. Summary of training and evaluations for the Door-Opening task.

1) Opening the door: $\varphi_1 := \mathbf{F}(\angle door\_hinge \geq 0.3)$. Angle is measured in radians.
2) Reaching the door handle: $\varphi_2 := \mathbf{F}(\|ee - door\_handle\| < 0.2)$; end-effector should be within $2cm$ of the door handle.

E. Task - Safe Mobile Navigation

The mobile robot consists of two independently driven parallel wheels and one free-rolling rear wheel, having similar dynamics as a TurtleBot. The environment contains 8 hazard markers scattered around the map and a single goal location. The location of the hazards, goal and robot are randomized for each episode. Traversing any of the hazards incurs a cost of 1. Due to the map randomization, the optimal policy may not always be able to ensure complete hazard-avoidance, and must rather minimize this cost. The robot is equipped with Lidar that provides 16 measurements each for the distances between: (i) robot and goal, and (ii) robot and nearest hazard. The task specifications are:

1) Reaching the goal: $\varphi_g := \mathbf{F}(\bigvee_{i=1}^{16}(d_g^i < 0.1))$, where d_g^i is the Lidar's i-th distance measurement to the goal.
2) Maintaining safety: $\varphi_s := \mathbf{G}(cost < 1)$, where $cost$ is the value incurred when the risk-area Lidar detects that the robot is too close to a hazard. The $cost$ is given by the formula $\bigwedge_{i=1}^{16}(d_l^i > 0)$, where d_l^i is the risk-Lidar's i-th distance measurement to the nearest hazard.
3) Completing the task within a specific time: $\varphi_t := \mathbf{G}(t < T)$, where $T = 1000$ is the maximum episode time.

The training and evaluation results are shown in Fig. 5.

F. Task - Safe FreightFranka Cabinet Drawer

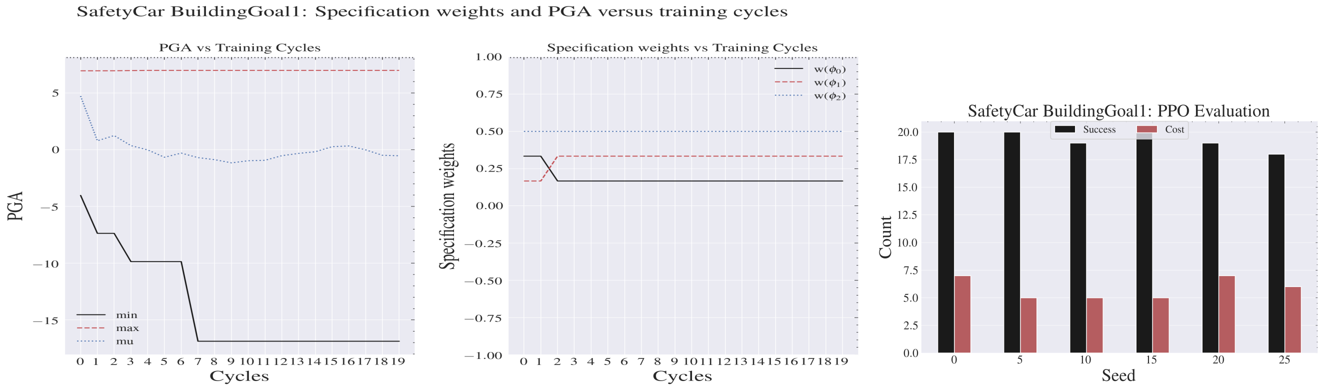
This setup consist of a *Franka Emika Panda* manipulator arm mounted on a *Fetch Robotics Freight* mobile robot platform. The environment consists of a cabinet with an open drawer and a rectangular risk zone. The task for the mobile-manipulator is to close the drawer while minimizing entry into the risk zone. The robot is controlled via its joint-space. The specifications for both these tasks are:

1) Closing the drawer: $\varphi_g := \mathbf{F}(drawer_y < 0.2)$. The drawer must be closed (as measured by its y-axis) within a 0.2 units tolerance.
2) Maintaining safety: $\varphi_s := \mathbf{G}(cost < 1)$, where $cost$ is the value incurred when the risk-area Lidar detects that the robot is too close to a hazard. The $cost$ is given by the formula $\bigwedge_{i=1}^{16}(d_l^i > 0)$, where d_l^i is the risk-Lidar's i-th distance measurement to the nearest hazard.
3) Completing the task within a specific time: $\varphi_t := \mathbf{G}(t < T)$, where $T = 192$ is the episode horizon.

The training and evaluation results are shown in Fig. 6.

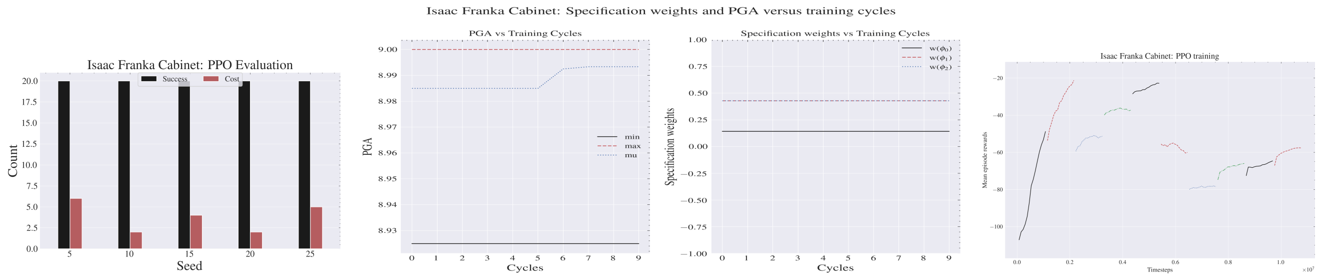
References

[1] G. E. Fainekos and G. J. Pappas, "Robustness of temporal logic specifications for continuous-time signals," *Theoretical Computer Science*, 2009.
[2] S. Jaksic, E. Bartocci, R. Grosu, T. Nguyen, and D. Nickovic, "Quantitative monitoring of STL with edit distance," *Formal Methods in System Design*, 2018.
[3] A. Donzé and O. Maler, "Robust satisfaction of temporal logic over real-valued signals," in *FORMATS*, 2010.



(a) PGA and Weights (b) Evaluation summary

Fig. 5. Summary of training and evaluations for the Safe Mobile Navigation task.



(a) Evaluation summary (b) PGA and Weights (c) RL training summary

Fig. 6. Summary of training and evaluations for the Safe FreightFranka Cabinet Drawer task.

[4] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. The MIT Press, 2018.

[5] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *ICML*. Morgan Kaufmann, 1999, pp. 278–287.

[6] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *CoRR*, vol. abs/1606.01540, 2016.

[7] W. Zhou and W. Li, "Safety-aware apprenticeship learning," in *CAV*. Springer, 2018.

[8] Q. Gallouédec, N. Cazin, E. Dellandréa, and L. Chen, "panda-gym: Open-Source Goal-Conditioned Environments for Robotic Learning," *NeurIPS Workshop*, 2021.

[9] J. Xu, B. Li, B. Lu, Y.-H. Liu, Q. Dou, and P.-A. Heng, "Surrol: An open-source reinforcement learning centered and dvrk compatible platform for surgical robot learning," in *IROS*, 2021.

[10] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, "An open-source research kit for the da vinci surgical system," in *ICRA*, 2014.

[11] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *ICML*, 2018.

[12] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, "Hindsight experience replay," in *NeurIPS*, vol. 30, 2017.

[13] T. Huang, K. Chen, B. Li, Y.-H. Liu, and Q. Dou, "Demonstration-guided reinforcement learning with efficient exploration for task automation of surgical robot," in *ICRA*, 2023.

[14] S. Haldar, V. Mathur, D. Yarats, and L. Pinto, "Watch and match: Supercharging imitation with regularized optimal transport," *CoRL*, 2022.

TABLE II

ALL TASK HYPERPARAMETERS.

PARAMETERS	VALUES					
	Panda-Reach	Needle-Reach	Panda Pick-Place	Panda Door	Safe Navigation	Safe FreightFranka Drawer
# Demos	5					
Reward Model[a]	FCN $[200 \rightarrow 200]$		GP (Scale+RBF)	FCN $[16 \rightarrow 16 \rightarrow 16]$	FCN $[64 \rightarrow 64]$	
RL						
Model	SAC+HER		TQC+HER	TQC	PPO	
Training Timesteps	$2 \cdot 10^5$	$2.5 \cdot 10^5$	10^7	$5 \cdot 10^6$		10^7
# AL-STL Cycles	5			25	10	
Policy Network[b]	Shared $[64 \rightarrow 64]$		Shared $[512 \rightarrow 512 \rightarrow 512]$	Shared $[256 \rightarrow 256]$	Exclusive $[256 \rightarrow 256]$	Exclusive $[512 \rightarrow 512]$
Learning Rate	$3 \cdot 10^{-4}$		10^{-3}		$3 \cdot 10^{-4}$	$5 \cdot 10^{-4}$
Discount Factor γ	0.95			0.97	0.95	0.99
Learning Starts	100		1000	100	-	-
Batch Size	256		2048	256	2500	1024
Polyak Update τ	0.005		0.05	0.5	-	-
# Epochs	-				15	10
# Rollout Buffer	-				5000	76,800
# Envs	-				5	400
PGA λ	0.9			0.3	0.5	
Training Success Rate	100%		98%	98%	-	-
Test Success Rate	100%		96%	100%	$(98\%, 29\%)$	$(100\%, 19\%)$
Training Time (hours)	-		10.75 (2.15/cycle)	6.5 (0.26/cycle)	20 (0.8/cycle)	1.5 (0.15/cycle)

[a] FCN: Fully Connected Neural Network; GP: Gaussian Process.

[b] Shared: Both policy and value functions share parameters; Exclusive: Policy and value functions have independent parameters (neural networks).