

## 868 A Proofs for Theorem 1 and Theorem 2

869 *Proof of Theorem 1* For any  $D \in \Delta\Pi$ , we have

$$\begin{aligned}
L(D, \bar{\lambda}) &= \frac{1}{T} \sum_{t=1}^T L(D, \lambda_t) && (L(D, \lambda) \text{ is linear in } \lambda) \\
&\leq \frac{1}{T} \sum_{t=1}^T (L(D_t, \lambda_t) + \epsilon_{\text{offline-RL}}(n)) && (\text{due to Equation (4)}) \\
&\leq \frac{1}{T} \sum_{t=1}^T L(D_t, \bar{\lambda}) + \epsilon_{\text{offline-RL}}(n) + \frac{R_T(\Lambda)}{T} && (\text{due to Equation (5)}) \\
&= L(\bar{D}, \bar{\lambda}) + \epsilon_{\text{offline-RL}}(n) + \frac{R_T(\Lambda)}{T} && (L(D, \lambda) \text{ is linear in } D).
\end{aligned}$$

870 For any  $\lambda \in \Lambda$ , we have

$$\begin{aligned}
L(\bar{D}, \lambda) &= \frac{1}{T} \sum_{t=1}^T L(D_t, \lambda) \\
&\geq \frac{1}{T} \sum_{t=1}^T L(D_t, \lambda_t) - \frac{R_T(\Lambda)}{T} && (\text{due to Equation (5)}) \\
&\geq \frac{1}{T} \sum_{t=1}^T L(\bar{D}, \lambda_t) - \epsilon_{\text{offline-RL}}(n) - \frac{R_T(\Lambda)}{T} && (\text{due to Equation (4)}) \\
&= L(\bar{D}, \bar{\lambda}) - \epsilon_{\text{offline-RL}}(n) - \frac{R_T(\Lambda)}{T}.
\end{aligned}$$

871

□

872 *Proof of Theorem 2* The result directly follows from the assumption of the stochastic oracle in  
873 Definition 4, the regret bound of EXP3 [Auer et al., 2002b], and the approximation error for the  
874 projection from  $[0, C]$  into the discrete domain  $\Lambda$  with resolution  $1/K$  is  $1/K$ .

875 In particular, by [Auer et al., 2002b], we have

$$\sum_{t=1}^T L(\tilde{D}_t, \lambda_t) - \min_{\lambda \in \Lambda} \sum_{t=1}^T L(\tilde{D}_t, \lambda) \leq H\sqrt{2\sqrt{TK} \ln K}.$$

876 Applying the result in Theorem 1, we have

$$\max_{D \in \Delta\Pi} L(D, \hat{\lambda}) + \epsilon_1 \leq L(\bar{D}, \hat{\lambda}) \leq \min_{\lambda \in \Lambda} L(\bar{D}, \lambda) + \epsilon_1,$$

877 where

$$\epsilon_1 = \epsilon_{\text{offline-RL}}(n) + H\sqrt{\frac{2K \log K}{T}}, \text{ and } \hat{\lambda} := \frac{1}{T} \sum_{t=1}^T \lambda_t.$$

878 Note that the running average  $\hat{\lambda}$  does not guarantee to be in  $\Lambda$ . Thus, with  $\bar{\lambda} = \text{Proj}_{\Lambda}(\hat{\lambda})$  – a projection  
879 of  $\hat{\lambda}$  onto  $\Lambda$ , we have that,

$$|L(D, \hat{\lambda}) - L(D, \bar{\lambda})| \leq \frac{H}{K}.$$

880 Overall, we have

$$\max_{D \in \Delta\Pi} L(D, \bar{\lambda}) + \epsilon_1 + \frac{H}{K} \leq L(\bar{D}, \bar{\lambda}) \leq \min_{\lambda \in \Lambda} L(\bar{D}, \lambda) + \epsilon_1 + \frac{H}{K},$$

881 Thus,  $(\bar{D}, \bar{\lambda})$  is  $\epsilon$ -approximate equilibrium, where

$$\epsilon = \epsilon_{\text{offline-RL}}(n) + H\sqrt{\frac{2K \log K}{T}} + \frac{H}{K}.$$

882

□

## B Ablation Results for O3SRL

We provide additional ablation results for O3SRL beyond those presented in the main paper.

### B.1 Number of update steps $M$

Recall that our approximate algorithm for empirical evaluation performs  $M$  stochastic gradient updates of a given offline RL algorithm in each iteration. Table 5 presents an ablation on the number of offline RL update steps  $M$ . This parameter controls how long the policy is trained before updating the  $\lambda$  value to balance reward and constraint satisfaction. The results demonstrate that  $M = 10$  provides a stable trade-off between learning progress and responsiveness to constraint violations. With  $M = 1$ , the frequent updates to  $\lambda$  overall increases costs adjusting the policy aggressively. Conversely, setting  $M = 100$  delays  $\lambda$  updates, which can result in policies drifting away from feasible regions, as seen in the sharp increase in costs for DroneRun and DroneCircle. Overall, the ablation results validate that safe and effective offline learning in O3SRL requires coordinated progress on both the policy and the constraint via well-timed  $\lambda$  updates. The choice of  $M = 10$  strikes this balance, supporting its use for all the results presented in the main paper.

Table 5: O3SRL results for different values of  $M$ : Offline RL update steps. Each value shows the average reward ( $\uparrow$ ) and cost ( $\downarrow$ ). Higher reward is better, and lower cost (up to threshold 1) is better.

Tasks		$M = 1$	$M = 10$	$M = 100$
BallRun	Reward $\uparrow$	<b>0.25<math>\pm</math>0.03</b>	<b>0.25<math>\pm</math>0.03</b>	<b>0.26<math>\pm</math>0.03</b>
	Cost $\downarrow$	<b>0.37<math>\pm</math>0.64</b>	<b>0.00<math>\pm</math>0.00</b>	<b>0.00<math>\pm</math>0.00</b>
CarRun	Reward $\uparrow$	<b>0.96<math>\pm</math>0.00</b>	<b>0.96<math>\pm</math>0.01</b>	<b>0.96<math>\pm</math>0.00</b>
	Cost $\downarrow$	<b>0.00<math>\pm</math>0.01</b>	<b>0.02<math>\pm</math>0.03</b>	<b>0.01<math>\pm</math>0.01</b>
DroneRun	Reward $\uparrow$	0.34 $\pm$ 0.06	<b>0.32<math>\pm</math>0.05</b>	0.27 $\pm$ 0.08
	Cost $\downarrow$	1.10 $\pm$ 0.90	<b>0.68<math>\pm</math>1.18</b>	3.72 $\pm$ 4.44
AntRun	Reward $\uparrow$	<b>0.34<math>\pm</math>0.05</b>	<b>0.33<math>\pm</math>0.13</b>	<b>0.39<math>\pm</math>0.16</b>
	Cost $\downarrow$	<b>0.20<math>\pm</math>0.21</b>	<b>0.14<math>\pm</math>0.10</b>	<b>0.97<math>\pm</math>1.36</b>
BallCircle	Reward $\uparrow$	<b>0.67<math>\pm</math>0.00</b>	<b>0.62<math>\pm</math>0.01</b>	<b>0.67<math>\pm</math>0.05</b>
	Cost $\downarrow$	<b>0.25<math>\pm</math>0.19</b>	<b>0.06<math>\pm</math>0.07</b>	<b>0.61<math>\pm</math>0.88</b>
CarCircle	Reward $\uparrow$	<b>0.66<math>\pm</math>0.01</b>	<b>0.66<math>\pm</math>0.03</b>	<b>0.68<math>\pm</math>0.01</b>
	Cost $\downarrow$	<b>0.10<math>\pm</math>0.17</b>	<b>0.11<math>\pm</math>0.16</b>	<b>0.17<math>\pm</math>0.29</b>
DroneCircle	Reward $\uparrow$	<b>0.44<math>\pm</math>0.06</b>	<b>0.49<math>\pm</math>0.07</b>	0.58 $\pm$ 0.20
	Cost $\downarrow$	<b>0.11<math>\pm</math>0.10</b>	<b>0.23<math>\pm</math>0.34</b>	4.87 $\pm$ 7.77
AntCircle	Reward $\uparrow$	<b>0.41<math>\pm</math>0.06</b>	<b>0.48<math>\pm</math>0.06</b>	<b>0.49<math>\pm</math>0.01</b>
	Cost $\downarrow$	<b>0.02<math>\pm</math>0.03</b>	<b>0.00<math>\pm</math>0.00</b>	<b>0.00<math>\pm</math>0.00</b>

### B.2 Search space of Lagrange variable $\lambda$

Recall that our search space for  $\lambda$  is  $[0, C]$ . Table 6 presents an ablation study on the impact of varying  $C$ , the maximum allowable value for the Lagrange variable  $\lambda$ , which determines how strongly constraint violations are penalized in O3SRL. Results are shown for  $C = 2, 5$ , and 10.

The ablation results highlight that small values of  $C$  (e.g.,  $C = 2$ ) often lead to insufficient constraint enforcement, resulting in catastrophic safety violations in some environments. For example, in *CarCircle* and *DroneCircle*, the average cost exceeds the threshold by a factor of over 8 and 13 respectively, completely defeating the purpose of safe learning. These cases show that when  $\lambda$  is capped too low, the algorithm lacks the leverage to penalize unsafe behavior adequately.

In contrast, increasing  $C$  to 5 or 10 significantly improves safety across all tasks. For instance, in *DroneCircle*, the cost drops from 13.32 at  $C = 2$  to 0.23 at  $C = 5$ , and further to 0.00 at  $C = 10$ .

908 However, in tasks with already low costs (e.g., *AntCircle*), large  $C$  values can lead to reduced reward,  
 909 due to over-penalization.

910 Overall,  $C = 5$  emerges as a robust choice, offering strong safety guarantees while maintaining high  
 911 reward across environments. This ablation underscores the necessity of a sufficiently expressive  $\lambda$   
 912 range to enable O3SRL to satisfy constraints in practice.

Table 6: O3SRL results for different values of  $C$ :  $\lambda$  maximum value. Each value shows the average reward ( $\uparrow$ ) and cost ( $\downarrow$ ). Higher reward is better, and lower cost (up to threshold 1) is better.

Tasks		$C = 2$	$C = 5$	$C = 10$
BallRun	Reward $\uparrow$	<b>0.28<math>\pm</math>0.01</b>	<b>0.25<math>\pm</math>0.03</b>	<b>0.25<math>\pm</math>0.01</b>
	Cost $\downarrow$	<b>0.00<math>\pm</math>0.00</b>	<b>0.00<math>\pm</math>0.00</b>	<b>0.00<math>\pm</math>0.00</b>
CarRun	Reward $\uparrow$	<b>0.97<math>\pm</math>0.00</b>	<b>0.96<math>\pm</math>0.01</b>	<b>0.95<math>\pm</math>0.01</b>
	Cost $\downarrow$	<b>0.06<math>\pm</math>0.10</b>	<b>0.02<math>\pm</math>0.03</b>	<b>0.00<math>\pm</math>0.00</b>
DroneRun	Reward $\uparrow$	0.43 $\pm$ 0.10	<b>0.32<math>\pm</math>0.05</b>	<b>0.33<math>\pm</math>0.05</b>
	Cost $\downarrow$	1.93 $\pm$ 1.90	<b>0.68<math>\pm</math>1.18</b>	<b>0.38<math>\pm</math>0.66</b>
AntRun	Reward $\uparrow$	0.34 $\pm$ 0.09	<b>0.33<math>\pm</math>0.13</b>	<b>0.20<math>\pm</math>0.06</b>
	Cost $\downarrow$	1.16 $\pm$ 0.95	<b>0.14<math>\pm</math>0.10</b>	<b>0.27<math>\pm</math>0.47</b>
BallCircle	Reward $\uparrow$	0.82 $\pm$ 0.06	<b>0.62<math>\pm</math>0.01</b>	<b>0.58<math>\pm</math>0.03</b>
	Cost $\downarrow$	7.20 $\pm$ 2.99	<b>0.06<math>\pm</math>0.07</b>	<b>0.01<math>\pm</math>0.01</b>
CarCircle	Reward $\uparrow$	0.75 $\pm$ 0.11	<b>0.66<math>\pm</math>0.03</b>	<b>0.57<math>\pm</math>0.08</b>
	Cost $\downarrow$	8.07 $\pm$ 7.53	<b>0.11<math>\pm</math>0.16</b>	<b>0.00<math>\pm</math>0.00</b>
DroneCircle	Reward $\uparrow$	0.79 $\pm$ 0.06	<b>0.49<math>\pm</math>0.07</b>	<b>0.37<math>\pm</math>0.03</b>
	Cost $\downarrow$	13.32 $\pm$ 2.62	<b>0.23<math>\pm</math>0.34</b>	<b>0.00<math>\pm</math>0.00</b>
AntCircle	Reward $\uparrow$	<b>0.58<math>\pm</math>0.02</b>	<b>0.48<math>\pm</math>0.06</b>	<b>0.41<math>\pm</math>0.02</b>
	Cost $\downarrow$	<b>0.25<math>\pm</math>0.26</b>	<b>0.00<math>\pm</math>0.00</b>	<b>0.04<math>\pm</math>0.07</b>

### 913 B.3 Discretization of $\lambda$

914 Table 7 presents an ablation comparing two  $\lambda$  set choices used in O3SRL: a uniformly spaced set  
 915 (e.g., [0, 1.25, 2.5, 3.75, 5]) and an adaptive set that retains the extreme values 0 and 5 but distributes  
 916 intermediate values more densely near lower penalty regions (e.g., [0, 0.5, 1.12, 2.5, 5]). The purpose  
 917 of the adaptive grid is to modulate constraint penalization based on the cost threshold  $\kappa$ , allowing the  
 918 algorithm to apply smaller penalties in more permissive cost budget scenarios. The results show that  
 919 while both grids perform similarly under strict cost threshold constraints ( $\kappa = 5$ ), the adaptive grid  
 920 provides clear advantages as the constraint becomes more relaxed. For instance, under  $\kappa = 40$ , the  
 921 adaptive grid allows significantly higher reward in tasks such as *DroneCircle* (reward improves from  
 922 0.45 to 0.65), by leveraging a smoother trade-off between reward maximization and cost constraint  
 923 satisfaction. These findings suggest that finer resolution at low penalty levels enables better policy  
 924 selection for higher cost constraint thresholds, making the adaptive grid more effective for balancing  
 925 performance and safety.

## 926 C Experimental Details

927 This section provides additional details about the experimental setup and hyper-parameters.

### 928 C.1 Description of environments

929 We evaluate O3SRL performance in a set of environments from the **Bullet-Safety-Gym** suite. Each  
 930 environment is defined by a combination of *robot type* and *task*, resulting in scenarios such as  
 931 *BallRun*, *CarCircle*, *DroneRun*, and so on. The agent types include **Ball**, **Car**, **Drone**, and **Ant**, and  
 932 each is evaluated under either the **Run** or **Circle** task. Figure 1 illustrates the Bullet-Safety-Gym

Table 7: O3SRL results for different values of the set of  $\lambda$ s and cost limits  $\kappa$ . Each value shows the average reward ( $\uparrow$ ) and cost ( $\downarrow$ ). Higher reward is better, and lower cost (up to threshold 1) is better.

Tasks		$\kappa = 5$		$\kappa = 20$		$\kappa = 40$	
		uniform	adaptive	uniform	adaptive	uniform	adaptive
BallRun	Reward $\uparrow$	<b>0.25</b> $\pm$ 0.01	<b>0.25</b> $\pm$ 0.03	<b>0.25</b> $\pm$ 0.01	<b>0.26</b> $\pm$ 0.01	<b>0.25</b> $\pm$ 0.01	<b>0.53</b> $\pm$ 0.48
	Cost $\downarrow$	<b>0.00</b> $\pm$ 0.00	<b>0.00</b> $\pm$ 0.00	<b>0.00</b> $\pm$ 0.00	<b>0.06</b> $\pm$ 0.11	<b>0.00</b> $\pm$ 0.00	<b>0.71</b> $\pm$ 1.24
DroneRun	Reward $\uparrow$	0.39 $\pm$ 0.04	<b>0.32</b> $\pm$ 0.05	<b>0.31</b> $\pm$ 0.11	<b>0.38</b> $\pm$ 0.05	<b>0.30</b> $\pm$ 0.05	<b>0.38</b> $\pm$ 0.10
	Cost $\downarrow$	1.95 $\pm$ 1.97	<b>0.68</b> $\pm$ 1.18	<b>0.22</b> $\pm$ 0.33	<b>0.41</b> $\pm$ 0.41	<b>0.60</b> $\pm$ 0.72	<b>0.44</b> $\pm$ 0.44
BallCircle	Reward $\uparrow$	<b>0.63</b> $\pm$ 0.04	<b>0.62</b> $\pm$ 0.01	<b>0.63</b> $\pm$ 0.08	<b>0.69</b> $\pm$ 0.09	<b>0.62</b> $\pm$ 0.06	<b>0.76</b> $\pm$ 0.04
	Cost $\downarrow$	<b>0.07</b> $\pm$ 0.12	<b>0.06</b> $\pm$ 0.07	<b>0.20</b> $\pm$ 0.30	<b>0.58</b> $\pm$ 0.66	<b>0.10</b> $\pm$ 0.10	<b>0.64</b> $\pm$ 0.18
DroneCircle	Reward $\uparrow$	<b>0.46</b> $\pm$ 0.02	<b>0.49</b> $\pm$ 0.07	<b>0.46</b> $\pm$ 0.01	<b>0.53</b> $\pm$ 0.01	<b>0.45</b> $\pm$ 0.03	<b>0.65</b> $\pm$ 0.11
	Cost $\downarrow$	<b>0.00</b> $\pm$ 0.00	<b>0.23</b> $\pm$ 0.34	<b>0.00</b> $\pm$ 0.00	<b>0.36</b> $\pm$ 0.27	<b>0.00</b> $\pm$ 0.00	<b>0.85</b> $\pm$ 0.55

933 environments, demonstrating various agent types (Ball, Car, Drone, Ant) performing the Run and  
 934 Circle tasks.

935 **Run Environments:** In the Run task, agents are required to move quickly along a straight corridor.  
 936 They receive rewards based on forward progress, encouraging high-speed locomotion. However,  
 937 they are penalized for crossing lateral safety boundaries or exceeding a predefined velocity threshold.  
 938 These safety constraints are not enforced through physical barriers but through a cost signal that  
 939 tracks violations.

940 **Circle Environments:** In the Circle task, agents are trained to move in a clockwise direction around  
 941 a circular track. Rewards are higher when agents maintain fast, smooth motion near the boundary of  
 942 the circle. Deviating from the intended path or exiting the safety zone results in penalties.

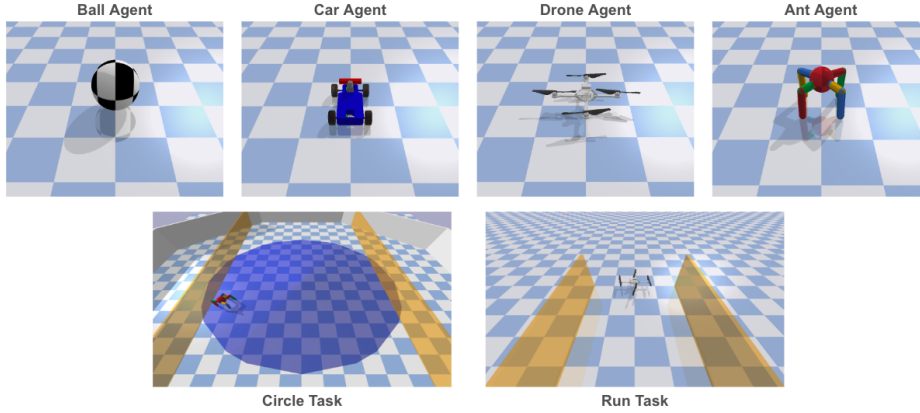


Figure 1: Visualization of the Bullet-Safety-Gym environments featuring different agents and tasks.

## 943 C.2 Training details

944 Our implementation is available at <https://anonymous.4open.science/r/O3SRL-E55F/>

945 **Adaptive  $\lambda$  set:** We construct the  $\lambda$  set using a geometric spacing scheme that adapts to the cost  
 946 limit. The grid always includes  $\lambda = 0$  and a maximum value  $\lambda_{\max} = C$ , and places  $k - 2$  intermediate  
 947 points between them. These intermediate values are spaced geometrically and scaled by a factor  
 948  $\text{shrink} = (\text{reference\_limit}/\text{cost\_limit})^{\alpha_{\text{shrink}}}$ , which compresses the grid as the cost limit increases,  
 949 where  $\text{reference\_limit} = 5$  and  $\alpha_{\text{shrink}} = 0.3$ . This design ensures finer resolution among low-penalty  
 950 values when constraints are loose, and broader coverage when strict cost constraints require stronger  
 951 penalization.

952 **Rewards scale:** We observe that reward distributions in offline datasets can be heavy-tailed, with  
occasional extreme values that disproportionately affect learning stability, as illustrated in Figure 2

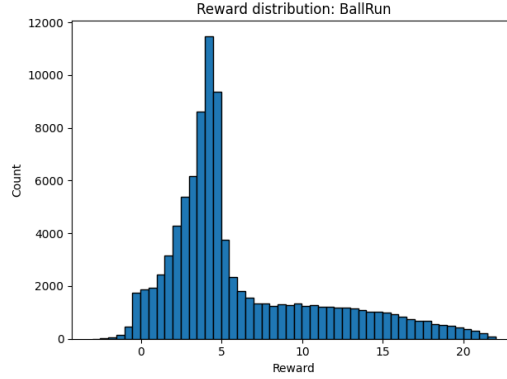


Figure 2: An example of a heavy-tailed reward distribution from the *BallRun* dataset.

953

954 To mitigate this, we clip rewards symmetrically at the  $\tau$ -th percentile of their absolute values:

$$r_{\text{clip}} = \text{percentile}(|r|, \tau), \quad r \leftarrow \text{clip}(r, -r_{\text{clip}}, r_{\text{clip}})$$

955 where  $\tau = 99$ . We then scale down the clipped rewards multiplying by  $0.9/r_{\text{clip}}$ .

956 **Hyperparameters:** We employ a batch size of 1024 for the tasks *CarCircle*, *DroneRun*, *AntCircle*,  
957 and *BallCircle*, while a batch size of 512 is used for the remaining tasks: *CarRun*, *DroneCircle*,  
958 *AntRun*, and *BallRun*. The other hyperparameters used in our experiments are listed in Table 8. For  
959 TD3+BC, we adopt the same hyperparameter settings as in the original paper.

Table 8: O3SRL Hyperparameters

<b>EXP3 parameters</b>	
Number of arms $k$	5
Maximum $C$	5
$\lambda$ update frequency $M$	10
$\lambda$ learning rate $\eta$	$2e-3$
<b>TD3BC parameters</b>	
Discount $\gamma$	0.99
Policy noise	0.2
Policy noise clip	(0.5, 0.5)
Policy update frequency	2
$\alpha$	2.5
Optimizer	Adam
Actor, Critic learning rate	$3e-4$
Actor, Critic hidden size	256
Training steps	100000
Seed	[10, 20, 30]

960 **Training Curves of O3SRL :** Figure 3 shows the training curves of O3SRL across the different  
961 tasks, plotting both the average reward and constraint cost over training iterations. The curves  
962 demonstrate that O3SRL is able to quickly stabilize and achieve a favorable trade-off between reward  
963 maximization and cost minimization. In most tasks, the cost decreases steadily and remains below  
964 the constraint threshold, while the reward improves progressively.

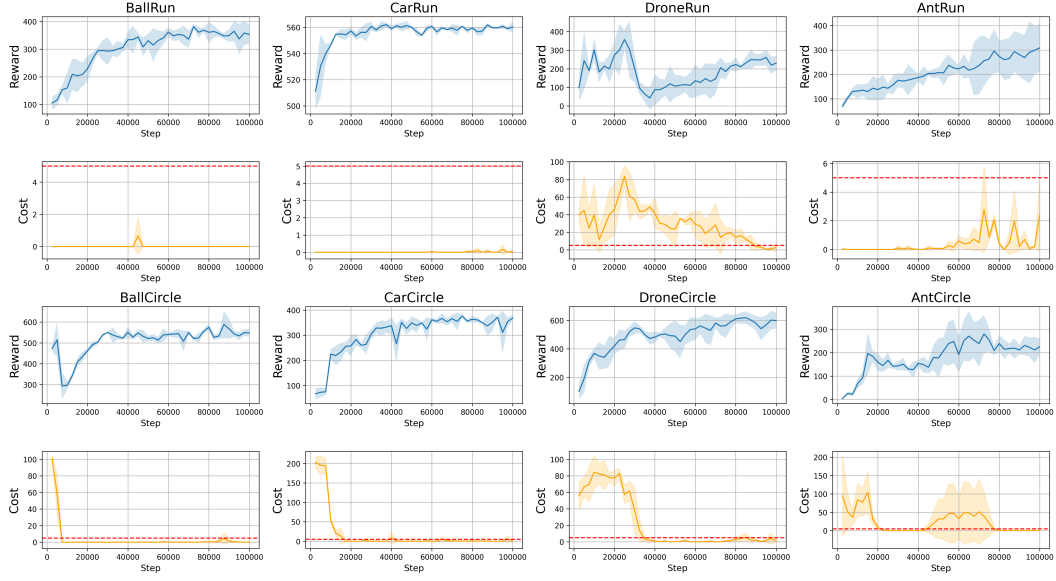


Figure 3: Training curves of O3SRL across different tasks, showing average reward (top) and cost (bottom) over training iterations.

965 **Computational Resources and Training Time:** All experiments were conducted using an NVIDIA  
966 A40 GPU with 48GB of memory. Training a single task for one random seed takes approximately 20  
967 minutes when the server is not under load.