Diffusion-Reinforcement Learning Hierarchical Motion Planning in Adversarial Multi-agent Games

Zixuan Wu*, Sean Ye*, Manisha Natarajan* and Matthew C. Gombolay*

I. Environment Details

A. Terrain

The terrain has varying visibility levels, with dense forests in Prisoner Escape and wave height regions in Narco Interdiction that hinder the agents' detection abilities. The two domains are different in size, team composition and configuration. The Narco Traffic Interdiction domain is approximately three times as large as Prison Escape and with one more pursuit agent.

B. Pursuit Policy

If the evader is detected nearby and within reaching distance, the search agent will directly move to the detected location using a pointwise policy. Otherwise, the search agent will intercept by moving to the interception point perpendicular to the opponent agent's velocity and then towards the opponent's estimated location. If the evader is not detected, the search agent will spiral around the estimated evader location for a set time. If the evader is still not found, the policy is reset, and the search agent starts a spiral search at a random location.

II. Training Details

In this section, we will briefly introduce the diffusion and RL training details. It includes the hyperparameters and training pipelines.

A. Hyperparameters

We will introduce the hyperparameters used in diffusion and RL training at the two different domains respectively. We train our diffusion model from RRT* waypoints dataset with 10000 downsampled paths and validate with 1000 paths each of which includes 10 waypoints. We also set a small weight decay coefficient as 5×10^{-4} to prevent over-fitting.

TABLE I

Diffusion Hyperparameters (Prisoner Escape)

Parameter Name	Notation	Value
waypoint number	N_w	10
learning rate	l_d	2e-5
weight decay	wd	5e-4
epoch number	N_d	100

*All authors are associated with the Institute of Robotics and Intelligent Machines (IRIM), Georgia Institute of Technology, Atlanta, GA 30308, USA.

Correspondance Author: Zixuan Wu zwu380@gatech.edu

TABLE II

Diffusion Hyperparameters (Narco Interdiction)

Parameter Name	Notation	Value
waypoint number	N_w	10
learning rate	l_d	2e-6
weight decay	wd	5e-4
epoch number	N_d	150

We will list the RL hyperparameters. Both DDPG and SAC are off-policy actor-critic RL methods and we set the policy and critic learning rate as 0.003. They are all using target critic with soft updating tricks to stablize the training process and we set the coefficient as 0.01. Since SAC has a self-adjusted entropy regularization term, the entropy learning rate, entropy target and regularization coefficient are set as 0.003, 3, 0.2 respectively.

TABLE III

RL Hyperparameters (Prisoner Escape)

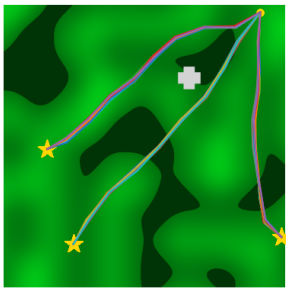
Parameter Name	Notation	Value
critic lr.	l_c	0.003
policy lr.	l_p	0.003
delay coef.	s_c	0.01
entropy lr.	l_e	0.003
entropy target	e_{trgt}	3
entropy regularization	α	0.2
waypoint reach threshold	d_w	10
discount factor	γ	0.97
detection penalty	r_{adv}	-1
distance penalty coeff.	c_d	$1 \sim 0.05$
distance penalty	r_d	$-2.5 * c_d * d$
waypoint reach reward	r_g	13

TABLE IV

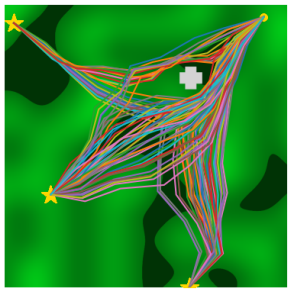
RL Hyperparameters (Narco Interdiction)

Parameter Name	Notation	Value
critic lr.	l_c	0.003
policy lr.	l_p	0.003
delay coef.	s_c	0.01
entropy lr.	l_e	0.003
entropy target	e_{trgt}	3
entropy regularization	α	0.2
waypoint reach threshold	d_w	30
discount factor	γ	0.97
detection penalty coeff.	r_{adv}	$1 \sim 0.25$
distance penalty	r_d	$-5 * c_d * d$
waypoint reach reward	r_g	20

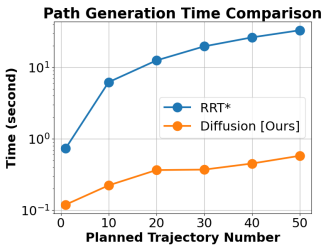
We summarize the hyperparameters on our two domains in Table I-IV. We save the diffusion model checkpoint every 2 epochs and select the one that has the best performance in our validation set to help our RL. The detection penalty



(a) Diffusion Trajectories (A*)



(b) Diffusion Trajectories (RRT*)



(c) Time Comparison

Traj.	RRT*	Diffusion
1	0.7±0.2	0.1±0.0
10	6.2±0.3	0.2±0.0
20	12.5±0.6	0.4±0.1
30	19.7±1.5	0.4±0.0
40	26.3±1.9	0.5±0.0
50	33.0±2.3	0.6±0.2

(d) Time Comparison (mean±std)

Fig. 1. The paths from the diffusion model trained on RRT* are more diverse than those trained on A* (1a-1b). Additionally, compared to the traditional RRT* planner, the diffusion model leverages the power of parallel computing to generate trajectories an order of magnitude faster (1c-1d).

coefficient c_d is linearly decreasing and d is the distance to the next waypoint.

III. Testing Details

A. Diffusion Paths Validation

The learned diffusion model should be able to generate diverse global paths that satisfies start and terminal constraints in a short time such that it can guide low-level RL to explore in valuable state space and accelerate training and inference. Figure 1a-1b compares the diffusion global paths trained on the datasets from RRT* and another commonly used path planner A*. We can see the diffusion model encodes a more spread distribution of the paths leading to the final hideouts when using RRT*. Figure 1c and Table 1d show the time needed to sample from diffusion model and get RRT* global plan. The diffusion model takes 85.7% less time generating one trajectory compared with RRT* which requires a map search and the gap is enlarged when generating more paths since we can draw samples from diffusion model in parallel. We generate the results using a machine with 11th Gen Intel Core i9-11900 processor and GeForce GTX 1660 Ti graphic card.