

Provably Safe Online Multi-Agent Navigation in Unknown Environments

Supplementary Material

Anonymous Author(s)

Affiliation

Address

email

1 A Exploration Mechanism

2 CLF-based approaches may cause agents to get stuck in local minima in complex navigation sce-
3 narios, such as environments with non-convex obstacle regions or where the destination is located
4 directly on the opposite side of an obstacle. To overcome this challenge, we propose an exploration
5 mechanism inspired by the Expansive Search Tree (EST) [1] and integrate it into our method to
6 encourage agents out of local minima.

7 Specifically, we consider the standard QP controller (6) as the convergence mode, and switch to
8 the exploration mode when the QP controller becomes infeasible, i.e., no feasible control satisfies
9 both convergence and safety constraints in (6). The exploration mode samples a state $\mathbf{x}_{\text{sample}} \sim$
10 $\mathcal{P}_{\text{state}, d_{\text{max}}}$ from its nearby space with the sampling radius determined by the maximum scanning
11 distance d_{max} of the LiDAR scanner, and generates an action by the QP controller to steer the agent
12 towards the sampled state $\mathbf{x}_{\text{sample}}$.

13 As we collect LiDAR readings at each time step and perform online SVM training, each agent
14 estimates the surrounding environment in a real-time manner. For any generated action that violates
15 safety constraints, the previously sampled state will be discarded and a new state will be sampled.
16 After repeating this sampling process for a fixed number of times N_{explore} , we switch back to the
17 convergence mode and navigates the agents towards their destinations. Algorithm 1 summarizes the
18 exploration mechanism integrated in our method.

19 B Methodology Details

20 We provide additional details about the proposed method in this section.

21 B.1 Graph Attention Networks

22 Graph attention networks (GATs) are a variant of graph neural networks (GNNs) that leverage the
23 attention mechanism to extract task-relevant features from graph data. Different from conventional
24 GNNs, which aggregate neighborhood information with (pre-defined) fixed weights, GATs allow
25 to differentiate neighbors w.r.t. their contributions when aggregating neighborhood information to
26 generate task-relevant features.

27 Specifically, consider a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with the node set $\mathcal{V} = \{1, \dots, n\}$ and the edge set \mathcal{E} .
28 The graph structure can be captured by a support matrix \mathbf{W} with (i, j) th entry $w_{ij} = [\mathbf{W}]_{ij} \neq 0$
29 if node i is connected to node j , i.e., $(i, j) \in \mathcal{E}$, or $i = j$, and $[\mathbf{W}]_{ij} = 0$ otherwise. The graph
30 data can be captured by a graph signal \mathbf{X} , which is a matrix whose i th row $\mathbf{x}_i = [\mathbf{X}]_i$ contains the
31 feature of node i . GATs are capable of assigning different attention coefficients $\{[\mathbf{W}]_{ij}\}_{j \in \mathcal{N}_i}$ to
32 the neighboring nodes $\{j\}_{j \in \mathcal{N}_i}$ of each node i , which represent the importance of the neighbors'

Algorithm 1 Exploration Mechanism (for agent A_i)

```
1: Input: State  $\mathbf{x}_i$ , target  $\mathbf{x}_i^d$ , safety constraints  $\mathcal{S}$ 
2: Output: Control action  $\mathbf{u}_i$ 
3: if QP controller infeasible then
4:   Switch to the exploration mode
5:   for  $i_{\text{explore}} = 1, \dots, N_{\text{explore}}$  do
6:     while exploration mode do
7:       Sample a state  $\mathbf{x}_{\text{sample}} \sim \mathcal{P}_{\text{state}, d_{\text{max}}}$ 
8:       Attempt to generate control action  $\mathbf{u}_i$  to steer towards  $\mathbf{x}_{\text{sample}}$ 
9:       if  $\mathbf{u}_i$  violates safety constraints  $\mathcal{S}$  then
10:        Discard  $\mathbf{u}_i$ 
11:       else
12:        break
13:   Apply  $\mathbf{u}_i$  to system (1)
14:   Scan the environment to collect LiDAR readings
15:   Train SVM with the latest LiDAR readings to update the environment CBF
16: Switch to the convergence mode
```

33 features for the feature update of the local node. These attention coefficients \mathbf{W} are learned with a
34 shared self-attention mechanism, enabling to quantify the relevance among graph nodes. Therefore,
35 GATs are *suitable candidates* to characterize the importance of the neighbors' LiDAR readings in
36 synthesizing the CBF of the unknown environment at the local agent.

37 The key component of the GAT lies in the attention mechanism, which computes attention coeffi-
38 cients between neighboring nodes. For the local node i and its neighboring node j , the unnormalized
39 attention coefficient is calculated as

$$e_{ij} = a(\mathbf{H}\mathbf{x}_i, \mathbf{H}\mathbf{x}_j), \quad (\text{S.1})$$

40 where \mathbf{H} is a learnable weight matrix (i.e., linear transformation) that maps the node features \mathbf{x}_i and
41 \mathbf{x}_j to some higher-order features, and $a(\cdot)$ is a shared attention function such as a feedforward neural
42 network. The unnormalized e_{ij} is then passed through a softmax function to get the normalized
43 attention coefficients as

$$w_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}_i} \exp(e_{ik})}, \quad (\text{S.2})$$

44 which makes the sum of neighbors' attention coefficients equal to one for ease of probabilistic
45 interpretation. With the normalized attention coefficients \mathbf{W} , each node i generates its task-relevant
46 feature as a weighted sum of neighbors' higher-order features

$$\mathbf{x}'_i = \sigma \left(\sum_{j \in \mathcal{N}_i} w_{ij} \mathbf{H}\mathbf{x}_j \right), \quad (\text{S.3})$$

47 where $\sigma(\cdot)$ is the non-linearity function such as ReLU and absolute value. Fig. 1 illustrates the
48 attention mechanism, where the task-relevant feature, in our case, is the predicted action $\mathbf{x}'_i = \mathbf{u}_i$
49 that mimics the expert action \mathbf{u}_i^* given in the training data.

50 **Discussion.** The attention coefficients learned by the GAT weigh the neighboring features when
51 generating the task-relevant feature, i.e., the predicted action. That is, they represent how much the
52 neighboring features are accounted for in the generation of the task-relevant feature. Therefore, it
53 provides justifications for considering the attention coefficients as indicators to quantify the impor-
54 tance of the neighboring agents to the local agent, and for leveraging the latter to remove LiDAR
55 readings of less important neighbors to reduce computation of online SVM training – see Section
56 4.2 of the main paper.

57 B.2 Methodology Framework

58 The proposed method contains two main components: SVM-based CBF-CLF-QP navigation and
59 attention-based computation reduction. The former generates safe actions for the agents to approach

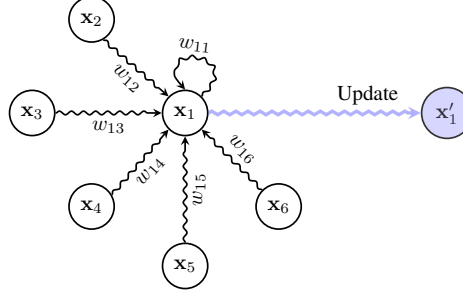


Figure 1. An illustration of the attention mechanism within the GAT [2]. The local node 1 aggregates the neighbors' features weighted by their corresponding attention coefficients, and leverages the latter to update its task-relevant feature \mathbf{x}'_1 .

their destinations (Section 4.1 of the main paper), while the latter identifies important neighbors with the attention mechanism of the GAT and uses only LiDAR readings of these neighbors for online SVM training to reduce real-time computation (Section 4.2 of the main paper). Fig. 2 provides an overview for the general framework of the proposed OE-CLBF controller to ease understanding.

C Proof of Theorem 1

We start the proof by defining the generalization error of the learned SVM classifier f_{SVM}^* as the probability risk in classifying any unseen data (\mathbf{z}, y) , i.e.,

$$R(f_{\text{SVM}}^*) = \Pr(f_{\text{SVM}}^*(\mathbf{z}) \neq y), \quad (\text{S.4})$$

where $\Pr(\cdot)$ is the probability measure and (\mathbf{z}, y) is drawn randomly from the space of interest. With the conditions provided in the theorem and the LiDAR readings $\mathcal{Z} = \{(\mathbf{z}_l, y_l)\}_{l=1}^L$, we can follow Theorem 1 of [3] to bound the probability risk as

$$R(f_{\text{SVM}}^*) \leq \frac{1}{L} \left(\kappa(f_{\text{SVM}}^*) \log_2 \left(\frac{8eL}{\kappa(f_{\text{SVM}}^*)} \right) \log_2(32L) + \ln \left(\frac{L^2}{\alpha} \right) \right) \text{ with } \kappa(f_{\text{SVM}}^*) = \left\lceil \left(\frac{8C}{\epsilon(f_{\text{SVM}}^*)} \right)^2 \right\rceil. \quad (\text{S.5})$$

For the learned safety function h_{SVM}^* from f_{SVM}^* , (S.5) is equivalent to that for any position $\mathbf{z} \notin \mathcal{Z}$, if $h_{\text{SVM}}^*(\mathbf{z}) > 0$, it holds that

$$\Pr(\mathbf{z} \in \mathcal{C}_{\text{safe}}) \geq 1 - \delta \quad (\text{S.6})$$

with

$$\delta \leq \frac{1}{L} \left(\kappa(f_{\text{SVM}}^*) \log_2 \left(\frac{8eL}{\kappa(f_{\text{SVM}}^*)} \right) \log_2(32L) + \ln \left(\frac{L^2}{\alpha} \right) \right) \text{ with } \kappa(f_{\text{SVM}}^*) = \left\lceil \left(\frac{8C}{\epsilon(f_{\text{SVM}}^*)} \right)^2 \right\rceil, \quad (\text{S.7})$$

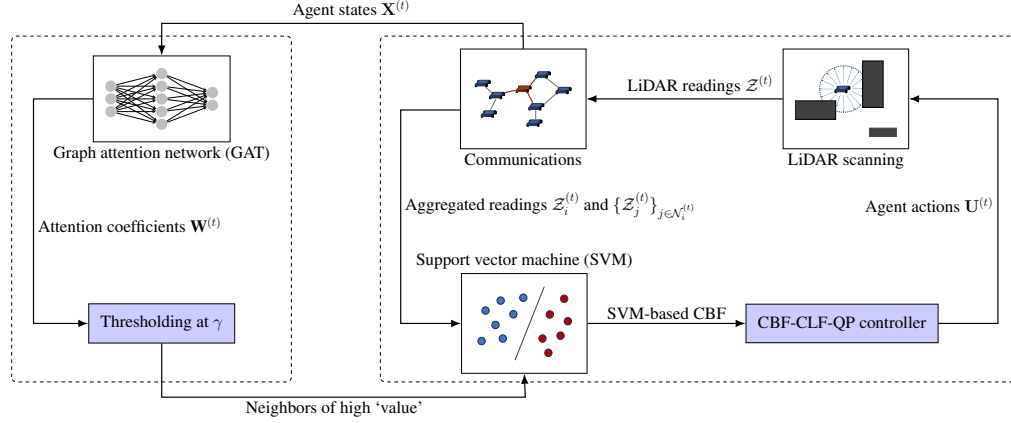
where $\mathcal{C}_{\text{safe}}$ is the safety set of the environment. Define the safety set $\mathcal{C}_{\text{SVM}, \text{safe}}$ w.r.t. the learned safety function h_{SVM}^* as

$$\mathcal{C}_{\text{SVM}, \text{safe}} = \{\mathbf{z} | h_{\text{SVM}}^*(\mathbf{z}) > 0 \text{ for all } \mathbf{z} \in \mathcal{E}\}. \quad (\text{S.8})$$

By substituting this safety function h_{SVM}^* into the CBF constraint (3) and the latter into the CBF-CLF-QP controller (6), we can follow Theorem 3 of [4] to prove that there exists a class \mathcal{K} function $\alpha(h_{\text{SVM}}^*)$ such that the h_{SVM}^* -based safety set $\mathcal{C}_{\text{SVM}, \text{safe}}$ [cf. (S.8)] is *forward invariant*, i.e., the agent will stay in $\mathcal{C}_{\text{SVM}, \text{safe}}$ with generated actions.

Since the h_{SVM}^* -based safety set $\mathcal{C}_{\text{SVM}, \text{safe}}$ depends on the learned safety function h_{SVM}^* , which, in turn, depends on the training data of L LiDAR readings \mathcal{Z} , and each agent only re-scans these LiDAR readings every Δt , we assume $\mathcal{C}_{\text{SVM}, \text{safe}}$ keeps the same in the time interval Δt . Therefore, we have

$$\mathbf{z}^{(t)} \in \mathcal{C}_{\text{SVM}, \text{safe}} \quad (\text{S.9})$$



Attention-based Computation Reduction

SVM-based CBF-CLF-QP Navigation

Figure 2. General framework of the proposed OE-CLBF controller, which contains SVM-based CBF-CLF-QP navigation and attention-based computation reduction. The former senses the surrounding unknown environment with LiDAR scanner, synthesizes the corresponding CBF with online SVM, and incorporates it into the CBF-CLF-QP controller to generate agent actions. The latter quantifies the importance of the neighboring agents to the local agent with GAT, thresholds useless LiDAR readings from less important neighbors, and uses only useful and relevant LiDAR readings for online SVM training to reduce computation.

83 for any time t until the next time interval Δt .

84 By combining (S.6) and (S.9), we complete the proof that

$$\Pr(\mathbf{z}^{(t)} \in \mathcal{C}_{\text{safe}}) \geq 1 - \delta \quad (\text{S.10})$$

85 with

$$\delta \leq \frac{1}{L} \left(\kappa(f_{\text{SVM}}^*) \log_2 \left(\frac{8eL}{\kappa(f_{\text{SVM}}^*)} \right) \log_2(32L) + \ln \left(\frac{L^2}{\alpha} \right) \right) \text{ with } \kappa(f_{\text{SVM}}^*) = \left\lceil \left(\frac{8C}{\epsilon(f_{\text{SVM}}^*)} \right)^2 \right\rceil \quad (\text{S.11})$$

86 for any time t in the next interval Δt .

87 D Additional Experiments

88 **Graph attention learning.** The goal of this experiment is to show the training procedure of the
 89 graph attention network (GAT), which mimics the expert action, i.e., the one generated with all
 90 neighborhood information, following an imitation learning framework. The initial and goal positions
 91 \mathbf{S} and \mathbf{D} of the agents are initialized randomly in the environment and the obstacles are distributed
 92 between \mathbf{S} and \mathbf{D} . We generate 200 multi-agent trajectories of 200 time steps with 4×10^4 samples
 93 to construct the dataset, and leverage the ADAM optimizer with a learning rate 10^{-4} for training.
 94 The mean square error (MSE) is used as the loss function to measure the difference between the
 95 predicted action and the expert action.

96 Fig. 3 plots the training procedure of the GAT over 10^4 epochs. We see that the training loss de-
 97 creases with the number of epochs and the decreasing rate reduces; ultimately, reaching a stationary
 98 solution. The convergent loss is small, which indicates a good performance of the trained GAT for
 99 action generation and a good prediction of attention coefficients for neighbors' importance.

100 **Exploration mechanism.** The goal of this experiment is to show the necessity of incorporating the
 101 exploration mechanism [Appendix A] into our method for resolving challenging navigation scenar-
 102 ios. Fig. 4 shows one example of these scenarios, where the obstacles form a non-convex safety
 103 space and the agent easily gets stuck in a local region due to its limited sensing and communication
 104 ranges. We see that our method navigates all agents towards destinations successfully in Fig. 4a,
 105 where the exploration mechanism pulls agent A_1 out of the local dilemma by searching over its
 106 nearby space. The method without exploration mechanism fails the navigation task of agent A_1 in
 107 Fig. 4b, because it can only generate actions that stay in the local safety set.

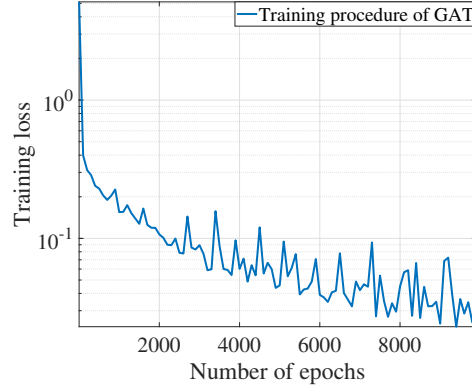


Figure 3. The training procedure of graph attention network (GAT) in the framework of imitation learning.

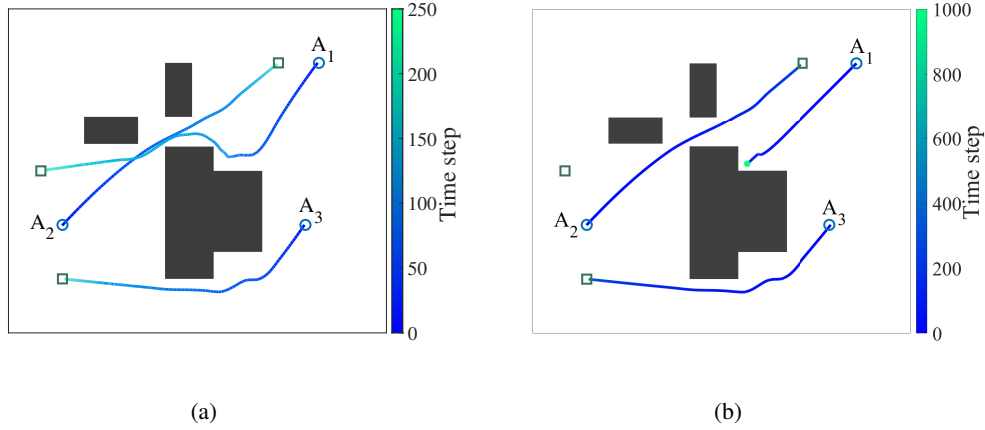


Figure 4. (a) Navigation procedure of our method with exploration mechanism, where all agents complete their tasks successfully. (b) Navigation procedure of our method without exploration mechanism, where agent A_1 gets stuck in local minima because of limited sensing and communication ranges. Blue circles are initial positions and green squares are destinations. Black rectangles are obstacles and blue-to-green lines are agent trajectories. Color bar represents time scale, showing that no agent-agent nor agent-obstacle collision occurs.

More Examples. Figs. 5a-5f show more examples of agents' trajectories generated by our method, for different navigation tasks in unknown environments with different obstacle positions, sizes and shapes. We see that in all these scenarios, all agents complete their navigation tasks successfully, where the success represents the agents reaching their destinations without collision within the maximal time step, and do not require any prior knowledge about environments, even in some challenging scenarios with cluttered obstacles. This highlights the effectiveness of the proposed OE-CLBF controller, which is applicable in unknown environments with different obstacle layouts.

References

- [1] J. M. Phillips, N. Bedrossian, and L. E. Kavraki. Guided expansive spaces trees: A search strategy for motion-and cost-constrained state spaces. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3968–3973, 2004.
- [2] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *International Conference on Learning Representations (ICLR)*, 2018.
- [3] R. Herbrich and T. Graepel. A pac-bayesian margin bound for linear classifiers: Why svms work. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2000.
- [4] W. Xiao and C. Belta. High-order control barrier functions. *IEEE Transactions on Automatic Control*, 67(7):3655–3662, 2022.

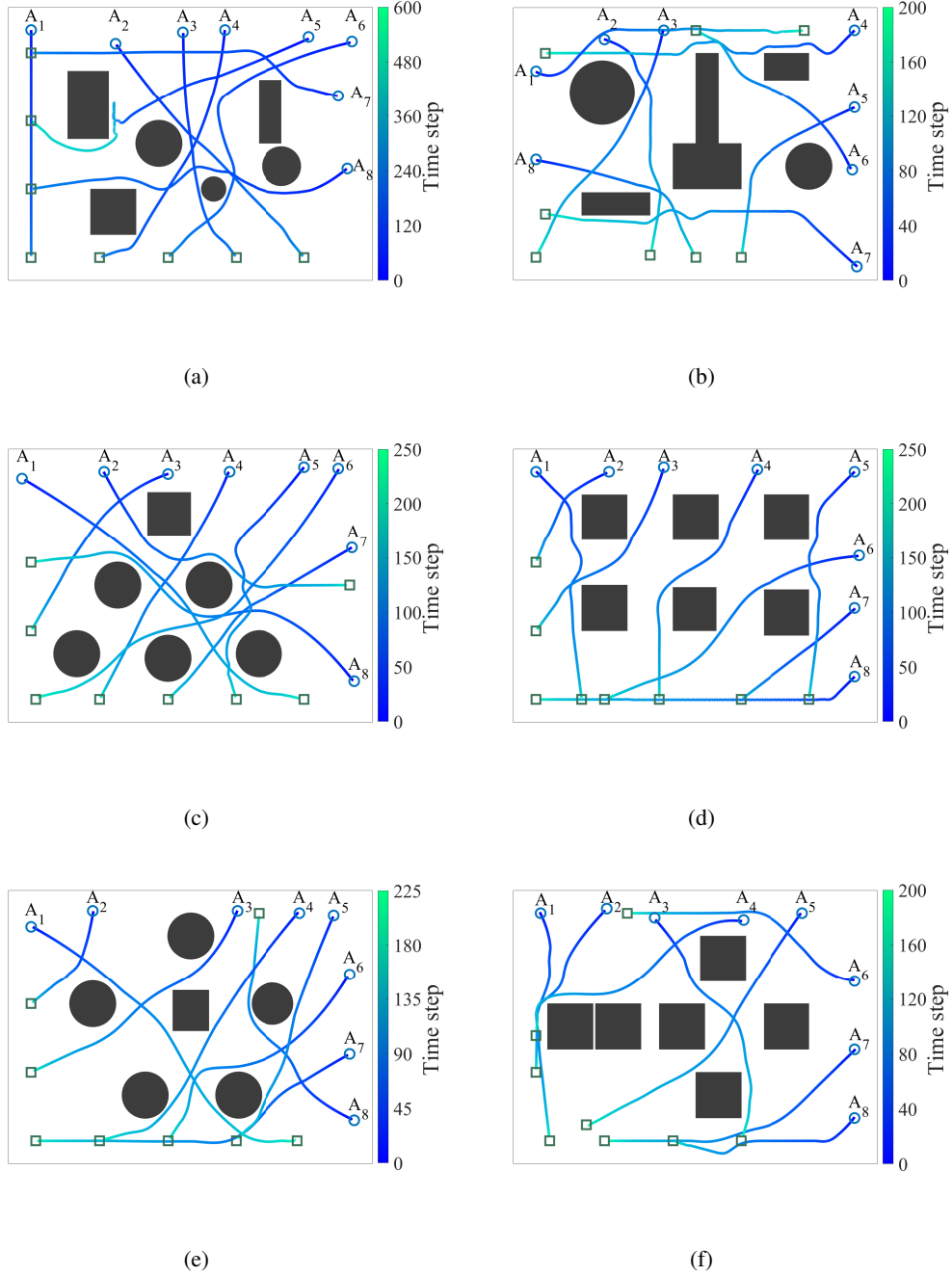


Figure 5. Navigation procedures of our method for different navigation tasks in different unknown environments with different obstacle layouts (i.e., different obstacle positions, sizes and shapes), and our method is capable of completing all navigation tasks successfully without collision either to obstacles or among agents.