Robotica (2024), **42**, pp. 2781–2800
doi:10.1017/S0263574724001115

RESEARCH ARTICLE

Robust motion planning for mobile robots under attacks against obstacle localization

Fenghua Wu[1], Wenbing Tang[2], Yuan Zhou[1], Shang-Wei Lin[1], Zuohua Ding[3] and Yang Liu[1]

[1]School of Computer Science and Engineering, Nanyang Technological University, Singapore, Singapore
[2]Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, Shanghai, PR China
[3]School of Information Science and Technology, Zhejiang Sci-Tech University, Hangzhou, PR China
Corresponding author: Wenbing Tang; Email: wenbingtang@hotmail.com

Received: 6 June 2023; **Revised:** 14 June 2024; **Accepted:** 20 June 2024; **First published online:** 18 September 2024

Keywords: motion planning; deep reinforcement learning; generative adversarial network; obstacle localization attacks

Abstract

Thanks to its real-time computation efficiency, deep reinforcement learning (DRL) has been widely applied in motion planning for mobile robots. In DRL-based methods, a DRL model computes an action for a robot based on the states of its surrounding obstacles, including other robots that may communicate with it. These methods always assume that the environment is attack-free and the obtained obstacles' states are reliable. However, in the real world, a robot may suffer from obstacle localization attacks (OLAs), such as sensor attacks, communication attacks, and remote-control attacks, which cause the robot to retrieve inaccurate positions of the surrounding obstacles. In this paper, we propose a robust motion planning method `ObsGAN-DRL`, integrating a generative adversarial network (GAN) into DRL models to mitigate OLAs in the environment. First, `ObsGAN-DRL` learns a generator based on the GAN model to compute the approximation of obstacles' accurate positions in benign and attack scenarios. Therefore, no detectors are required for `ObsGAN-DRL`. Second, by using the approximation positions of the surrounding obstacles, `ObsGAN-DRL` can leverage the state-of-the-art DRL methods to compute collision-free motion commands (e.g., velocity) efficiently. Comprehensive experiments show that `ObsGAN-DRL` can mitigate OLAs effectively and guarantee safety. We also demonstrate the generalization of `ObsGAN-DRL`.

1. Introduction

Motion planning is one of the essential tasks for mobile robots. It aims to generate a collision-free trajectory for each robot moving around in an environment with obstacles. Various motion planning methods, classified into traditional methods and heuristic approaches [1], have been proposed in the literature, such as state lattice [2], cell decomposition [3], roadmap [4], sampling [5], potential fields [6], velocity obstacles [7], mathematical programing [8, 9], fuzzy logic [10], and evolutionary algorithm [11]. Due to the crowded, dynamic, and unpredictable obstacles in the open environments, real-time and robust motion planning is necessary but challenging.

With the rapid development of deep learning technologies, deep reinforcement learning (DRL) has become a promising method for motion planning since it leverages offline training to improve online computation efficiency. Several agent-based DRL methods have been proposed, which take the states of the robot and the obstacles, either raw data (e.g., image) or post-processing data, as inputs and compute the corresponding motion commands [12–19]. For example, the authors in ref. [14] proposed a model combining Long Short-Term Memory (LSTM) and DRL to deal with a varying number of obstacles, where the obstacles are sorted according to their distances to the robot. Later, the work in ref. [16] improves this method by sorting the obstacles based on their collision criticality to the robot, as the relative distance cannot reflect the importance of an obstacle to the robot's collision avoidance.

Current DRL methods usually assume that the robot can retrieve accurate positions of its surrounding obstacles, including other robots. However, the robot is vulnerable to various physical and cyberattacks in the real world, resulting in receiving wrong information about the obstacles [20–24]. Obstacle localization attacks (OLAs) are one of the widely existing attacks in robotic systems, where an attacker can tamper with the positions of environmental obstacles and send malicious data to the robot [25–27]. For example, an attacker may obtain access to other robots unauthorizedly and send wrong-position messages to a robot [26]. An attacker may also perform cyberattacks against the communication network among robots to modify the transmitted position signals [27]. Consequently, the robot will make wrong decisions and cause severe accidents, for example, collisions, after receiving the wrong information. Although attacks may be relatively rare in the real world, they still occur frequently and lead to serious consequences. Therefore, it is crucial to develop effective and advanced mitigation methods before mobile robots are widely deployed.

The existing attack mitigation approaches predominantly focus on strategies that rely on attack detectors to identify attacks or compromised data [28–31]. For example, a cumulative sum detector is proposed in ref. [28] to detect attacks before a mitigation operation; a Bayesian inference-based detector is designed to activate the mitigation module [29]; in ref. [30], the mitigation module is triggered when a time-window-based detector detects an attack; an observer-based anomaly detection method is proposed in ref. [31] as a prerequisite for conducting the mitigation operation. However, they encounter two primary challenges: First, the design of attack detectors necessitates tailoring to distinct attack types, imposing a considerable burden in terms of development and maintenance. Second, the accuracy of attack detectors is paramount, as inaccuracies can trigger unnecessary mitigation actions and erroneous identification of normal data as compromised.

In response to these two challenges in the existing attack mitigation methods, we introduce a pioneering method that circumvents the dependence on attack detectors, a significant advancement not adequately addressed in the literature. Nevertheless, mitigation without detectors must overcome a critical challenge: devising a unified method capable of addressing normal and attack scenarios. Intuitively, some robust position estimation methods for a single robot could be repurposed to predict the obstacles' positions, such as multi-sensor fusion techniques [32–35] and complex filters [36–39]. For example, in ref. [32], the measurements from global positioning system (GPS), light detection and ranging (LiDAR), and inertial measurement unit (IMU) are fused to provide more reliable state estimation; the information from GPS, inertial navigation system (INS), and odometer sensors is fused to improve the accuracy of localization [33]. In addition, Kalman filter (KF) [36, 37] and particle filter (PF) [38, 39] are widely applied filters to generate robust state estimation. However, their performance will degrade significantly when they are applied to mitigate OLAs (Experimental results will be given in Section 4.3). Hence, it is of great necessity and significance to design detector-agnostic and unified mitigation strategies that can generate approximately correct data for both normal and attacked data.

Motivated by generative adversarial network (GAN) models, which can generate synthetic realistic data by learning the underlying data distribution, we introduce the application of GANs to design our detector-agnostic mitigation strategy. Benefiting from the adversarial training mechanism of GANs, one can train a generator to generate realistic positions for robots using the guidance provided by an auxiliary discriminator model. Consequently, a well-trained generator can effectively preserve normal positions while correcting compromised positions within a unified framework. Therefore, our method simplifies traditional attack mitigation methods' detection and mitigation steps into a single step.

In detail, focusing on the mitigation of OLAs against a robot, we propose a detector-agnostic motion planning method `ObsGAN-DRL` in this paper. `ObsGAN-DRL` consists of two modules: the security module and the functionality module. The security module leverages a generator model from a well-trained GAN to learn the distribution of the real data and approximate the accurate value of faked positions. Therefore, the security module can maintain benign positions while correcting attacked ones without any attack detector. The functionality module takes the robot's state and the mitigated states of obstacles as input and computes collision-free motion commands. Therefore, any motion planning algorithm can be used as long as it is appropriate for the robot's operating environment. For example, a dynamic environment requires the motion planning method to handle dynamic obstacles. In this paper, we apply the DRL model proposed in ref. [16] to deal with a varying number of obstacles in the environment and guarantee real-time computation efficiency. In detail, the states, including the corrected positions, of the obstacles and the robot are fed to the DRL model, which will infer a motion command to maximize the cumulative rewards.

We conduct comprehensive experiments to evaluate the effectiveness and efficiency of `ObsGAN-DRL` on two kinds of OLAs: disturbance attacks, which add random values to the original data, and replacement attacks, which use random values to replace the original positions. The results show that (1) the GAN-based mitigation strategy does not reduce the performance of benign scenarios significantly; (2) `ObsGAN-DRL` can effectively mitigate different OLAs, improving the success rates from 68.3% to 95.5% under disturbance attacks and from 52.2% to 96.3% under replacement attacks; (3) `ObsGAN-DRL` outperforms the KF and PF methods (95.5% vs 64.5% for disturbance attacks and 96.3% vs 64.65% for replacement attacks); and (4) the GAN-based mitigation strategy is compatible with other motion planning methods. Therefore, our method exhibits the following advantages: (1) it is detector-agnostic and planner-agnostic, meaning it can integrate into different motion planners without requiring any attack detectors; (2) our method can handle varying numbers of obstacles in different environments; and (3) our method can mitigate various attacks against obstacle locations.

The main contributions of this paper are threefold:

- We propose the first GAN-based strategy to mitigate OLAs against a robot without any attack detector. Moreover, it is planner-agnostic and generalizes to different attacks.
- We propose a two-module robust motion planning framework, `ObsGAN-DRL`, for a robot moving around in adversarial environments with a varying number of robots.
- We conduct extensive experiments to demonstrate the effectiveness and efficiency of `ObsGAN-DRL`.

The rest of this paper is organized as follows. Section 2 states the theoretical basis and the problem statement. Section 3 provides the detailed design and algorithm of `ObsGAN-DRL`. The experimental results are described in Section 4. Conclusion and future work are finally provided in Section 5.

2. Background and problem statement

2.1. Robot motion planning

For environment perception, a robot is equipped with different sensors, such as GPS and LiDAR, to identify its surrounding obstacles' states, that is, positions and velocities. The motion task for the robot is to move from the initial position $\mathbf{p}_I = (x_0, y_0)$ to the goal position $\mathbf{p}_G = (x_g, y_g)$ with a prefer speed v_f. Suppose the robot's safe radius is ρ, and the time is discretized into a set of time instants with an equal time step Δt. At any time instant t, $t \in \{0, 1, 2, \ldots\}$, the state of the robot is described as $s_t = (\mathbf{p}_t, \mathbf{v}_{t-1}, \mathbf{p}_g, v_f, \rho) \in \mathbb{R}^8$, where $\mathbf{p}_t = (x_t, y_t)$ is the robot's position at t, and $\mathbf{v}_{t-1} = (vx_{t-1}, vy_{t-1})$ is the velocity in the time duration $[(t-1)\Delta t, t\Delta t)$. The motion command at t is $\mathbf{v}_t$, that is, the velocity in the duration $[t\Delta t, (t+1)\Delta t)$. Note that for unicycle kinematics, $\mathbf{v}_t$ can be represented by (v, θ), that is, the speed and the orientation, resulting in $vx_t = v\cos\theta$ and $vy_t = v\sin\theta$. The set of detected obstacles at t is denoted as $\mathcal{O}_t = \{o_1, \ldots, o_{m_t}\}$, and the state of each obstacle o_i is denoted as $s_t^i = (\mathbf{p}_t^i, \mathbf{v}_t^i, \rho_i)$, where $\mathbf{p}_t^i = (x_t^i, y_t^i)$, $\mathbf{v}_t^i = (vx_t^i, vy_t^i)$, and ρ_i are the position, velocity, and safe radius of o_i at t, respectively. The state sequence of the obstacles at t is denoted as $s(\mathcal{O}_t)$. Hence, the motion problem for the robot can be described as:

$$\arg\min_{\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{T-1}} T \tag{1}$$

$$s.t. \quad \mathbf{p}_{t+1} = \mathbf{p}_t + \mathbf{v}_t \Delta t, \forall t \in \{0, 1, \ldots, T-1\}; \tag{2}$$

$$\|\mathbf{p}_t - \mathbf{p}_t^i\| \geq \rho + \rho_i, \forall o_i \in \mathcal{O}_t, t \in \{0, 1, \ldots, T\}; \tag{3}$$

$$\mathbf{p}_0 = \mathbf{p}_I, \ \mathbf{p}_T = \mathbf{p}_G. \tag{4}$$

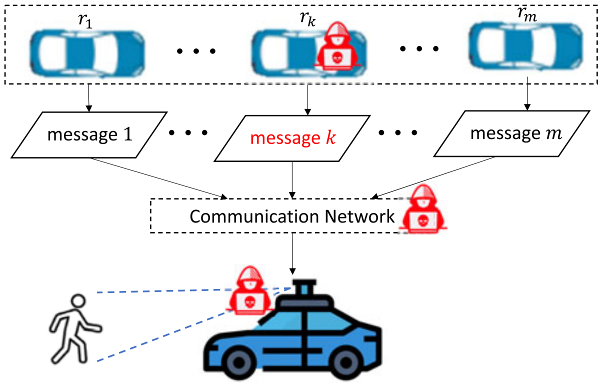


Figure 1. *The adversary performs obstacle localization attacks in a robot system via different attack ways.*

According to [12, 14], such a problem can be resolved efficiently with the DRL framework by maximizing the value function:

$$\mathbf{v}_t^* = \arg\max_{\mathbf{v}\in\mathcal{A}} R(s_t, s(O_t), \mathbf{v}) + \gamma^{\Delta t v_f} V^*(s_{t+1,\mathbf{v}}, s(O_{t+1})),$$

$$V^*(s_t, s(O_t)) = \sum_{t'=t}^{T-1} \gamma^{(t'-t)\Delta t v_f} R((s_{t'}, s(O_{t'})), \mathbf{v}_{t'}^*).$$

where $\mathcal{A}$ is the set of predefined actions, $\gamma \in [0, 1]$ is a discount factor, $R(s_t, s(O_t), \mathbf{v}_t)$ is one-step reward at s_t by taking $\mathbf{v}_t$, and $s_{t+1,\mathbf{v}}$ is the robot's next state under the action $\mathbf{v}$. Following [14], the reward function is defined as follows:

$$R(s_t, s(O_t), \mathbf{v}_t) = \begin{cases} 1, & \mathbf{p}_{t+1} = \mathbf{p}_g \\ -0.25, & d_{\min} \le 0 \\ -0.1 + 0.5 d_{\min}, & 0 < d_{\min} \le 0.2 \\ 0, & otherwise \end{cases}$$

where $d_{\min}$ is the minimal distance between the robot and the obstacles in O_t during the time duration $[t\Delta t, (t+1)\Delta t]$.

2.2. Obstacle localization attacks (OLAs)

2.2.1. Threat model

In this paper, we consider OLAs against a robot. Particularly, as shown in Fig. 1, a robot has two ways to retrieve the surrounding obstacles' positions. The first one is to retrieve the obstacles' positions via the equipped sensors, such as cameras and LiDARs. The second one is via communication: When a robot can communicate with the surrounding obstacles, such as other robots in a multi-robot system, it can also retrieve the obstacles' positions via communication. We assume that the retrieved positions may be malicious and can be modified by the adversary. Several reasons make this assumption realistic. (1) The equipped sensors may suffer from attacks, such as jamming attacks and spoofing attacks, such that they cannot be used to localize the positions of obstacles. (2) The communication network among robots is vulnerable to communication attacks [40, 41], and the attacker can modify data-in-transit or even cut off communication among robots [42]. (3) Some robots may be intruded into via the system vulnerabilities of the robot and send malicious messages to others [26].

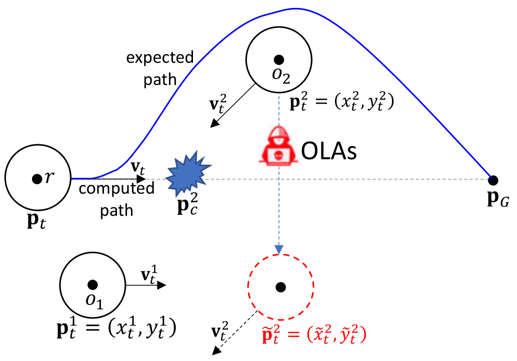


Figure 2. *Effects of position errors on the motion planning of a robot. The circles represent the real positions of the two obstacle robots. The arrows indicate their motion velocities. The red dashed circle represents r's received position of o_2.*

We assume that a robot can be under attack at some time instant while attack-free at another time instant. Hence, as shown in Fig. 1, we assume that during the motion of robots, the attacker has one of the following three capabilities: (1) interfering with the related sensors (e.g., LiDAR) such that they cannot localize the surrounding obstacles accurately, (2) modifying the positions transmitted in the communication network, and (3) manipulating a robot to send a fake position to the communication network. We also assume the attacker cannot access the control software and hence cannot bypass `ObsGAN-DRL`. This assumption can be guaranteed via different technologies, such as partitioning operating systems [43]. In addition, all robots are assumed to be able to locate their own positions accurately. Several technologies can guarantee this assumption, such as multiple sensor fusion [44].

2.2.2. Example of OLAs

Under OLAs, a robot may receive fake positions of the obstacles. As shown in Fig. 2, o_1 and o_2 are two robots that can communicate with the robot r. At the current time t, the real position of o_2 is $\mathbf{p}_t^2 = (x_t^2, y_t^2)$. However, due to OLAs, the received position by r is $\tilde{\mathbf{p}}_t^2 = (\tilde{x}_t^2, \tilde{y}_t^2)$.

2.3. Motivation and problem statement

In the case of OLAs, there are errors between the real positions and the retrieved positions of the surrounding obstacles. Such errors may lead the robot to make a wrong decision and cause collisions. For example, as shown in Fig. 2, the robot and the two obstacles are at $\mathbf{p}_t$, $\mathbf{p}_t^1$, and $\mathbf{p}_t^2$, respectively. Therefore, the robot is expected to plan the blue path to move to the target $\mathbf{p}_G$. However, due to OLAs, the robot r received a wrong position of o_2, that is, $\tilde{\mathbf{p}}_t^2$. According to the wrong position, r will move directly to the target. Consequently, the robot r collides with o_2 at the position $\mathbf{p}_c^2$.

Therefore, to generate a collision-free trajectory in an adversarial environment with OLAs, the robot needs to mitigate the detrimental effects of OLAs. Hence, the problem studied in this paper can be described as follows:

Problem 1: *Given a robot navigating through an adversarial environment with OLAs and varying obstacles, design a robust motion planning method such that the robot can move toward its target safely and efficiently without any attack detector.*

Remark 1. *Note that following other research* [1, 13, 14, 15, 18], *we focus on the design of a motion planner for a robot and assume that the functionalities of other modules can always work well. It means that given an input, the corresponding module can output the right output with respect to the input. However, these inputs may be attacked, resulting in fake outputs from the modules. Specifically, in this*

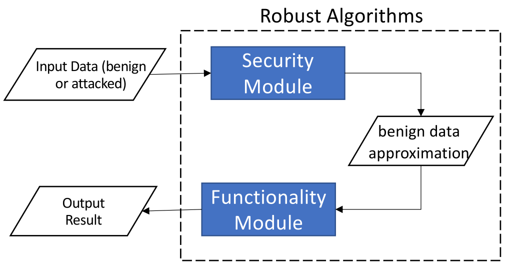


Figure 3. *The general architecture of our robust motion planning method.*

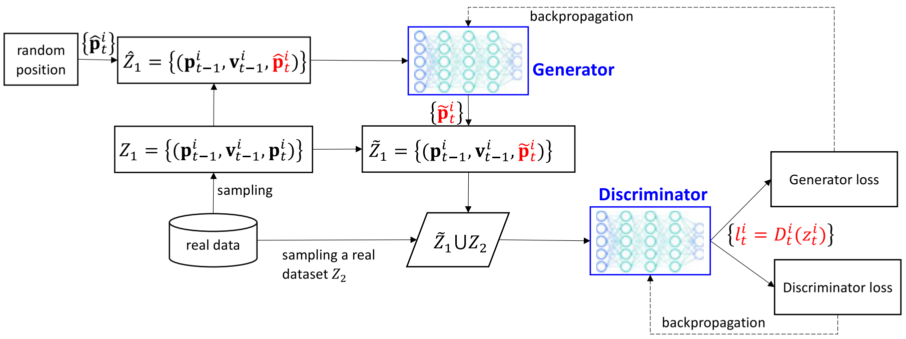

random position
$\{\hat{\mathbf{p}}_t^i\}$
$\hat{Z}_1 = \{(\mathbf{p}_{t-1}^i, \mathbf{v}_{t-1}^i, \hat{\mathbf{p}}_t^i)\}$
backpropagation
Generator
$\{\tilde{\mathbf{p}}_t^i\}$
$Z_1 = \{(\mathbf{p}_{t-1}^i, \mathbf{v}_{t-1}^i, \mathbf{p}_t^i)\}$
$\tilde{Z}_1 = \{(\mathbf{p}_{t-1}^i, \mathbf{v}_{t-1}^i, \tilde{\mathbf{p}}_t^i)\}$
sampling
real data
sampling a real dataset Z_2
$\tilde{Z}_1 \cup Z_2$
Discriminator
Generator loss
$\{l_t^i = D_t^i(z_t^i)\}$
Discriminator loss
backpropagation

Figure 4. *The architecture and training process of generative adversarial network model for obstacle localization attacks mitigation.*

paper, we focus on the attacks against the input of the perception module and the communication network that can result in fake positions of the surrounding objects, e.g., other robots.

3. ObsGAN-DRL: robust approach against OLAs with GAN

In this section, we will introduce our robust motion planning algorithm. The main idea is to approximate the real positions of the obstacles before they are used to compute a collision-free method. Fig. 3 shows the general architecture. It contains two modules: the security module, which is used to mitigate the attacked data and generate the approximation of the benign data, and the functionality module, which is used to generate the motion command.

3.1. GAN-based attack mitigation module

In this section, we detail the GAN-based attack mitigation strategy. Besides the primary objective of GAN models, which is generating synthetic data by learning the underlying data distribution, our method should also ensure that the generated positions accord with the kinematic constraints. Hence, we need to modify the general loss function in GAN. To correct the robot's attacked positions, we train a GAN model to generate the potential positions of the robot under OLAs. The main training purpose is to guide the generator in learning the real data distribution and the latent features of the real positions, so the generator can approximate the real data based on historical records. Fig. 4 shows the training process of our GAN model. Since the position of an obstacle relies on the position and velocity at the previous time instant, the training data should contain the previous state to learn the latent features of the real positions. Hence, the training data can be described as $Z = \{(\mathbf{p}_{t-1}, \mathbf{v}_{t-1}, \mathbf{p}_t)\}$. At each episode, we sample a subset Z_1 to generate a set of synthetic data $\hat{Z}_1$ for the training of the generator and a subset Z_2 for the training of the discriminator.

(1) Generator. The generator G is a multi-layer perceptron (MLP). The input of G is a faked data set $\hat{Z}_1$ from the sampled real data set Z_1, where

$$\hat{Z}_1 = \{\hat{z}_t^i = (\mathbf{p}_{k-1}^i, \mathbf{v}_{t-1}^i, \hat{\mathbf{p}}_t^i) | \forall z_t^i = (\mathbf{p}_{t-1}^i, \mathbf{v}_{t-1}^i, \mathbf{p}_t^i) \in Z_1\}$$

where $\hat{\mathbf{p}}_t^i = (\hat{x}_t^i, \hat{y}_t^i)$ is a random value simulating the results of OLAs; $\mathbf{p}_{t-1}^i = (x_{t-1}, y_{t-1})$ and $\mathbf{v}_{t-1}^i = (vx_{t-1}, vy_{t-1})$ are the recorded position and velocity at the previous time step. Note that during the inference stage, the position at $t-1$ is the position generated by the generator at $t-1$, rather than the retrieved one. The output of G is the potential position of an obstacle: $\tilde{\mathbf{p}}_t^i = (\tilde{x}_t, \tilde{y}_t)$. Hence, we have

$$\tilde{\mathbf{p}}_t^i = G(z_t; \theta_G) \tag{5}$$

where θ_G are the parameters of G that need to be learned. According to the generated positions, we can obtain a set of generated samples: $\tilde{Z}_1 = \{\tilde{z}_t^i = (\mathbf{p}_{t-1}^i, \mathbf{v}_{t-1}^i, \tilde{\mathbf{p}}_t^i) | \tilde{\mathbf{p}}_t^i = G(\hat{z}_t; \theta_G)\}$.

(2) Discriminator. The discriminator D is another MLP. It takes a real dataset $Z_2 = \{z_t^i = (\mathbf{p}_{t-1}^i, \mathbf{v}_{t-1}^i, \mathbf{p}_t^i)\}$ and the generated dataset $\tilde{Z}_1 = \{\tilde{z}_t^i = (\mathbf{p}_{t-1}^i, \mathbf{v}_{t-1}^i, \tilde{\mathbf{p}}_t^i)\}$ as inputs and generates the classification result $l_t^i = D(z; \theta_D)$ for each $z \in \tilde{Z}_1 \bigcup Z_2$, where θ_D are the parameters of D to be learnt.

(3) Loss Functions. As the general GAN models, the loss function for the discriminator can be written as:

$$Loss_D = \mathbb{E}_{z \in Z_2}[\log D(z; \theta_D)] + \mathbb{E}_{\tilde{z} \in \tilde{Z}_1}[\log (1 - D(\tilde{z}, \theta_D))]$$

The loss function for the generator is written as:

$$Loss_G = \mathbb{E}_{\tilde{z} \in \tilde{Z}_1}[\log (1 - D(\tilde{z}; \theta_D))] + \frac{\sum_{z_t^i \in Z_1} \|z_t^i - \tilde{z}_t^i\|}{|Z_1|}$$

Note that $\tilde{z}$ is function of θ_G.

Hence, the training process is to maximize $Loss_D$ while minimizing $Loss_G$, which is shown in Algorithm 1. It is performed by optimizing the following two optimization problems in sequence via gradient descent.

$$\min_{\theta_D} -Loss_D \tag{6}$$

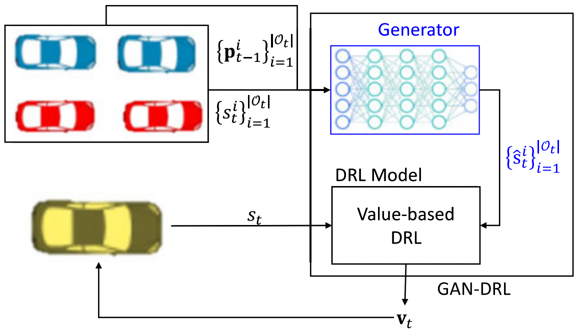
$$\min_{\theta_G} Loss_G \tag{7}$$

3.2. Robust motion planning

Following the GAN-based mitigation strategy, we propose our robust motion planning method `ObsGAN-DRL`. The architecture of `ObsGAN-DRL` is shown in Fig. 5, and the detailed motion planning process with `ObsGAN-DRL` is given in Algorithm 2. At any time instant, suppose the detected obstacles' states are $S_t = \{s_t^i = (\mathbf{p}_t^i, \mathbf{v}_t^i, \rho^i) | o_i \in O_t\}$. Under `ObsGAN-DRL`, S_t is processed by the well-trained generator (Lines 5–10); then, the resulting states $\tilde{S}_t = \{\tilde{s}_t^i = (\tilde{\mathbf{p}}_t^i, \mathbf{v}_t^i, \rho^i) | o_i \in O_t\}$, where $\tilde{\mathbf{p}}_t^i$ is generated by the generator, are sent to the DRL model to generate the corresponding command (Lines 11–17). Note that the training of the DRL model can be referred to ref. [16]. In the DRL model, the obstacles are sorted in ascending order with respect to their collision criticality based on the states $\tilde{S}_t$; then, the corresponding LSTM model takes the sorted obstacles' states as input and generates a unified hidden state; finally, the MLP model in the DRL model takes the robot's state and the hidden state as input, computes the corresponding values for all action candidates, and returns the one with the maximal value. The motion is

Algorithm 1: Training of GAN model.

```
   Input: Training episodes $\eta_{gan}$ for GAN.
   Output: The trained Generator model.
 1 Collect a training dataset $Z$ via conventional motion planning algorithms, such as ORCA [45];
 2 Initialize the generator model $G$ and the discriminator model $D$;
 3 for $i = 1 : \eta_{gan}$ do
 4     Slice up the training data into mini-batches;
 5     for each mini-batch $Z_2$ do
           /* Train D */
 6         Randomly select a data set from $Z$ with the same number of samples in $Z_2$, denoted as $Z_1$;
 7         For each sample in $Z_1$, replace its current position with a random value, and the new set is
             denoted as $\hat{Z}_1$;
 8         Update $D$ by solving Equation 6 using gradient descent based on $\hat{Z}_1$ and $Z_2$;
           /* Train G */
 9         Randomly select a data set from $Z$ with the same number of samples in $Z_2$, denoted as $Z_3$;
10         Replace the current positions in $Z_3$ by the output of the Generator and denote the new set as $\tilde{Z}_3$;
11         Update $G$ by solving Equation 7 using gradient descent based on $\tilde{Z}_3$ and $Z_3$;
12 return G
```



$\{\mathbf{p}_{t-1}^i\}_{i=1}^{|O_t|}$
$\{s_t^i\}_{i=1}^{|O_t|}$
Generator
$\{\hat{s}_t^i\}_{i=1}^{|O_t|}$
DRL Model
Value-based DRL
s_t
GAN-DRL
$\mathbf{v}_t$

Figure 5. *The framework of* `ObsGAN-DRL`.

completed if the robot reaches its destination within a given error tolerance ε or a collision is detected during the movement from $\mathbf{p}_t$ to $\mathbf{p}_{t+1}$ (Line 21).

4. Experimental evaluation

In this section, we conduct simulations to validate the performance of attack mitigation of `ObsGAN-DRL` in different scenarios. Specifically, we first evaluate the performance of `ObsGAN-DRL` with a varying number of obstacles under two OLAs (Section 4.2). Then, we compare our GAN-based mitigation strategy with the KF and PF, which are the state-of-the-art mitigation methods for OLAs (Section 4.3). Thirdly, we evaluate the compatibility of the GAN-based with the socially attentive reinforcement learning (SARL) method [15], another DRL method for motion planning, and optimal reciprocal collision avoidance (ORCA), a conventional motion planning method (Section 4.4). Finally, in Section 4.5, we evaluate our method via realistic simulation experiments on the well-established simulator Gazebo with Robot Operating System (ROS). Three *AscTec Firefly* drones are simulated in Gazebo, where one is controlled by our method and the other two are obstacles.

Algorithm 2: Motion planning with `ObsGAN-DRL`.

```
Input: The well-trained Generator model $G$ and the Value-based DRL model $V$, action space $\mathcal{A}$, the
       reference speed $v_f$, the probability for greedy selection $\epsilon$, the initial and target positions $\mathbf{p}_0$ and $\mathbf{p}_g$,
       maximal motion steps $T_m$ and discrete time step $\Delta t$.
Output: The trajectory $\mathcal{P}$.
1  $done = False$; $t = 0$; $s_t = (\mathbf{p}_0, \mathbf{0}, \mathbf{p}_g \ v_f, \rho)$;
2  $\mathcal{P} = \mathcal{P} \bigcup \{s_t\}$;
3  while not done do
4      Retrieve the observable states of the surrounding obstacles $S_t$;
5      $\tilde{S}_t = \emptyset$;
6      for $s_t^i = (p_t^i, v_t^i, \rho^i) \in S_t$ do
7          Retrieve the previous state sorted in the robot;
8          Compute the approximate position $\tilde{\mathbf{p}}_t^i$ using $G$;
9          $\tilde{S}_t = \tilde{S}_t \cup \{(\tilde{\mathbf{p}}_t^i, \mathbf{v}_t^i, \rho^i)\}$;
10         Update the previously stored state to $(\tilde{\mathbf{p}}_t^i, \mathbf{v}_t^i)$;
11     Predict the next states of the obstacles $S'_{t+1}$ based on $\tilde{S}_t$;
12     Sort $S'_{t+1}$ based on their collision criticality;
13     Generate a random value $p$ between $[0, 1]$;
14     if $p < \epsilon$ then
15         Select $\mathbf{v}_t$ randomly from $\mathcal{A}$;
16     else
17         $\mathbf{v}_t = \arg\max_{\mathbf{v} \in \mathcal{A}} R(s_t, \tilde{S}_t, \mathbf{v}) + \gamma^{\Delta t v_f} V(\bar{s}_{t+1,\mathbf{v}}, S'_{t+1})$;
18     Move to the next position $\mathbf{p}_{t+1}$ with $\mathbf{v}_t$;
19     $s_{t+1} = (\mathbf{p}_{t+1}, \mathbf{v}_t, \mathbf{p}_g, v_f, \rho)$;
20     $\mathcal{P} = \mathcal{P} \bigcup \{s_{t+1}\}$;
21     if $\|\boldsymbol{p}_{t+1} - \boldsymbol{p}_g\| \le \varepsilon$ or $collision(\boldsymbol{p}_t, \boldsymbol{p}_{t+1})$ or $t \le T_m$ then
22         $done = True$
23     $t = t + 1$;
24 return $\mathcal{P}$.
```

4.1. Simulation setup

For the training of the GAN model, we collect 1,055,791 samples via ORCA [45]. The network architectures of the generator and the discriminator are (6, 128, 64, 2) and (6, 128, 64, 1), respectively. We train the generator and the discriminator iteratively for 50 epochs, with a batch size of 64. Their initial learning rates are 0.001. In the training phase of the generator, to simulate OLAs, the attacked position $\hat{\mathbf{p}}_t$ in each z_t is generated from the Gaussian distribution $N(0, 1)$.

For training of the DRL model, the preferred speed $v_f = 1$, the safe radius $\rho = 0.3$, and the action space $\mathcal{A} = \{(v_i \cos\theta_j, v_i \sin\theta_j) | v_i = (e^{i/5} - 1)/(e - 1), \theta_j = j/8, i = 0, 1, \cdots, 5, j = 0, \cdots, 7\}$. In each replay, each obstacle is randomly located and needs to move to the opposite location with respect to the origin of the coordinates. In addition, $\eta_{gan} = 10,000$, $\mu = 1$, $\omega = 50$, $T_m = 100$, $\Delta t = 0.25$, and $\gamma = 0.9$. The exploration rate of ϵ-greedy policy decreases linearly from 0.5 to 0.1 in the first 4,000 episodes and stays at 0.1 for the remaining episodes. The dimension of the LSTM's hidden state is 50, and it is initialized with a zero vector. The architecture of the value network is (150, 100, 100, 1). The LSTM model and value network are trained simultaneously with 10 obstacles. The number of obstacles during the testing varies from 1 to 14. During the training phase, the learning rate is 0.001. All models

Table I. *Overall performance analysis of our generative adversarial network (GAN)-based mitigation.*

	Success Rate	**Collision Rate**	**Timeout Rate**	**Motion Time**	**Cumulative Reward**
Normal	0.954	0.045	6.67E-05	7.78	0.342
Normal+GAN	0.897	0.103	6.67E-05	7.81	0.313
Disturbance	0.619	0.381	0	7.86	0.155
Replacement	0.453	0.547	0	8.04	0.057
Disturbance+GAN	0.898	0.102	6.67E-05	7.81	0.313
Replacement+GAN	0.906	0.094	6.67E-05	7.80	0.317

Note: The sum of success, collision, and timeout rates may not equal 1 due to rounding.

are implemented in PyTorch. Please refer to ref. [16] for the performance of the DRL-based planner under a different number of obstacles.

To show the mitigation capacity of `ObsGAN-DRL`, we investigate two kinds of attacks. The first one is disturbance attacks, where the position of each obstacle is modified by adding a random value. The second kind is replacement attacks, where the positions of other robots are replaced by random values. All the random values are generated from the standard normal distribution. Moreover, the attacks are launched at the 5th time step.

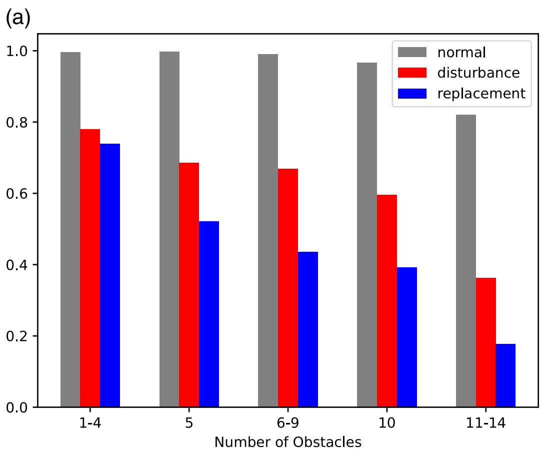
For testing scenarios, we randomly generate *10 test sets*, each of which contains *five groups* based on the number of obstacles in the test cases: {1, 2, 3, 4}, {5}, {6, 7, 8, 9}, {10}, and {11, 12, 13, 14}. Each group in each set contains 300 test cases, resulting in each test set containing 1,500 test cases. We perform six experiments on the 10 test sets:

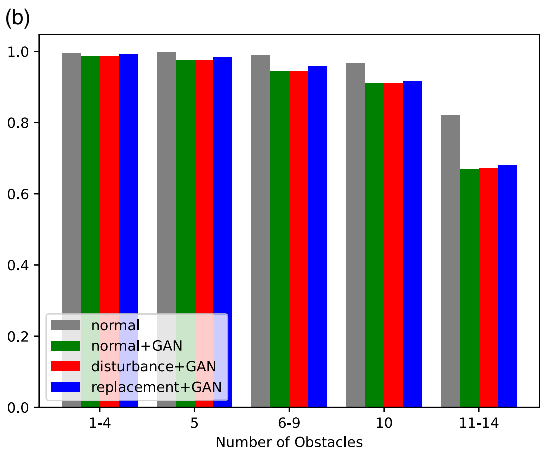
- *normal*: It is the baseline where the test cases are tested without any attacks or mitigations;
- *disturbance*: It is an attacked situation where the test cases are executed under disturbance attacks;
- *replacement*: It is another attacked situation where the test cases are executed under replacement attacks;
- *normal+GAN*: It is the situation in which each test case is pre-processed by the generator;
- *disturbance+GAN*: It is the application of `ObsGAN-DRL` under disturbance attacks;
- *replacement+GAN*: It is the application of `ObsGAN-DRL` under replacement attacks.

4.2. Overall performance of `ObsGAN-DRL`

In this section, we present the experimental results of `ObsGAN-DRL`. Table I illustrates the overall performance across the six experiments on the ten test sets. Notably, the last two columns provide specific insights into the quality of the generated paths. From the table, we can find that the attacks not only significantly diminish the success rate (0.954 vs. 0.619 for the disturbance attack and 0.954 vs. 0.453 for the replacement attack) but also degrade the path quality, manifested in increased motion time and reduced cumulative rewards. Therefore, an attack mitigation method is necessary for adversarial environments with OLAs.

From Table I, we also note a marginal decrease in the success rates for the GAN-based scenarios. To identify the reasons, we investigate the average success rates for each test group. Recall that the test cases in each test set are grouped based on the number of obstacles present. The results are given in Fig. 6. The figure shows that in the *normal* experiment, the average group success rate for the group with 11 to 14 obstacles is 0.821, significantly lower than the success rates for other groups (≥ 0.96). This, in turn, causes the performance of the GAN-based method to degrade from an average of 0.955 in the first





Average success rate for each group without GAN-based mitigation. Average success rates for each group with GAN-based mitigation.

Figure 6. *Average success rates for different groups in different experiments.*

four groups to 0.668 in the group with 11 to 14 obstacles. Moreover, in the GAN-based experiments, the average group success rates after GAN mitigation in this group are 0.671 and 0.680 for disturbance and replacement attacks, respectively, while the corresponding average success rates without mitigation are 0.362 and 0.178, resulting in 85.4% and 282% improvement in the average success rate. We can find that our strategy can still mitigate attacks significantly in the group with 11–14 obstacles. The primary reason for the lower success rate than other groups after mitigation is that the current normal DRL method may not perform optimally in the scenarios with 11–14 obstacles. From the results, we can conclude that the success rate of our method is dependent on the success rate of the original motion planning methods. The results indicate that the effectiveness of a mitigation method depends on both the mitigation technique itself and the original motion planning methods. Therefore, it is essential to focus not only on attack mitigation strategies but also on the development of effective motion planning methods.

Therefore, in the subsequent analysis, we focus on the first four groups, that is, test cases where the number of obstacles varies from 1 to 10. It is important to note that it does not compromise the effectiveness of our method. Table II shows the overall performance of `ObsGAN-DRL` on the refined test cases, that is, test cases with 1–10 obstacles, and Fig. 7 shows the corresponding average success rate in each test set (10 test sets in total). Note that *disturbance+GAN* and *normal+GAN* show very similar performance, so their success rates are almost overlapped in Fig. 7. From the results of *normal* and *normal+GAN*, we can find that the GAN-based security module does not significantly reduce the performance of the DRL-based functionality module (the average success rates are 0.988 and 0.956, respectively, and the average motion time for a scenario is 7.77 s and 7.80 s). Compared with *disturbance* and *disturbance+GAN*, `ObsGAN-DRL` improves the performance significantly under disturbance attacks, increasing the success rate by 42.75% (0.955 vs. 0.669). Similarly, `ObsGAN-DRL` significantly improves the performance of the DRL model under replacement attacks, from 0.573 to 0.963. Moreover, according to the average motion time, the attacks increase the motion time of success scenarios and reduce the reward significantly due to the high collision rate, while the motion time and reward are similar to the normal case after GAN-based mitigation.

In the sequel, we show the results of an example with two obstacles. Fig. 8 shows the paths under different situations, and Fig. 9 virtualizes the actual and attacked paths of the two obstacles. Fig. 8a shows the paths of the obstacles and the robot in the normal situation, while Figs. 8b and 8c show the robot's paths under disturbance and replacement attacks, respectively. Due to the attacks, the robot collides with the obstacles without any mitigation strategy. Figs. 8d, 8e, and 8f are the paths of the robot in normal and under disturbance and replacement attacks, respectively. From the figures, we can find that the mitigated

Table II. *Overall performance analysis for test cases with 1–10 obstacles.*

	Success Rate	Collision Rate	Timeout Rate	Motion Time	Cumulative Reward
Normal	0.988	0.012	8.33E-05	7.77	0.367
Disturbance	0.683	0.317	0	7.86	0.194
Replacement	0.522	0.478	0	8.02	0.097
Normal+GAN	0.955	0.045	8.33E-05	7.80	0.350
Disturbance+GAN	0.955	0.045	8.33E-05	7.80	0.350
Replacement+GAN	0.963	0.037	8.33E-05	7.79	0.354

Note: The sum of success, collision, and timeout rates may not equal 1 due to rounding.

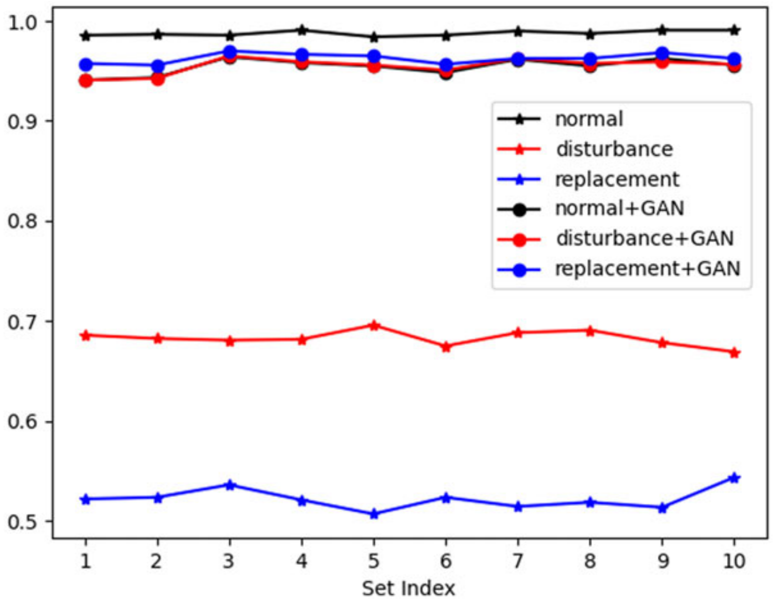


Figure 7. *Average success rates for the test cases with 1–10 obstacles in the 10 sets.*

paths can achieve a similar performance to the normal one. Note that as shown in Fig. 9, the two attacks are different.

To further validate the significant improvement of `ObsGAN-DRL`, the analysis of statistical significance between the situations without and with GAN mitigation is conducted. In this paper, we perform t-test for equal means. The results are shown in Table III. According to the t-test results, we can conclude that the results are significant, which means that `ObsGAN-DRL` can mitigate attacks rather than by chance.

Finally, we show the mitigation performance of our strategy under different attacks. In detail, we generate new attacked positions from a new Gaussian distribution $N(0, 2)$ and a uniform distribution $Uniform(-4, 4)$, respectively, for the disturbance and replacement attacks. The comparison results are given in Table IV. From the results, we can find that even though the original motion planning method shows different performance under different attacks, our GAN-based mitigation strategy can guarantee that the motion planning method achieves similar success rates to the normal one.

Therefore, we can conclude that `ObsGAN-DRL` is a general mitigation method that can effectively mitigate different OLAs against the surrounding obstacles' positions.

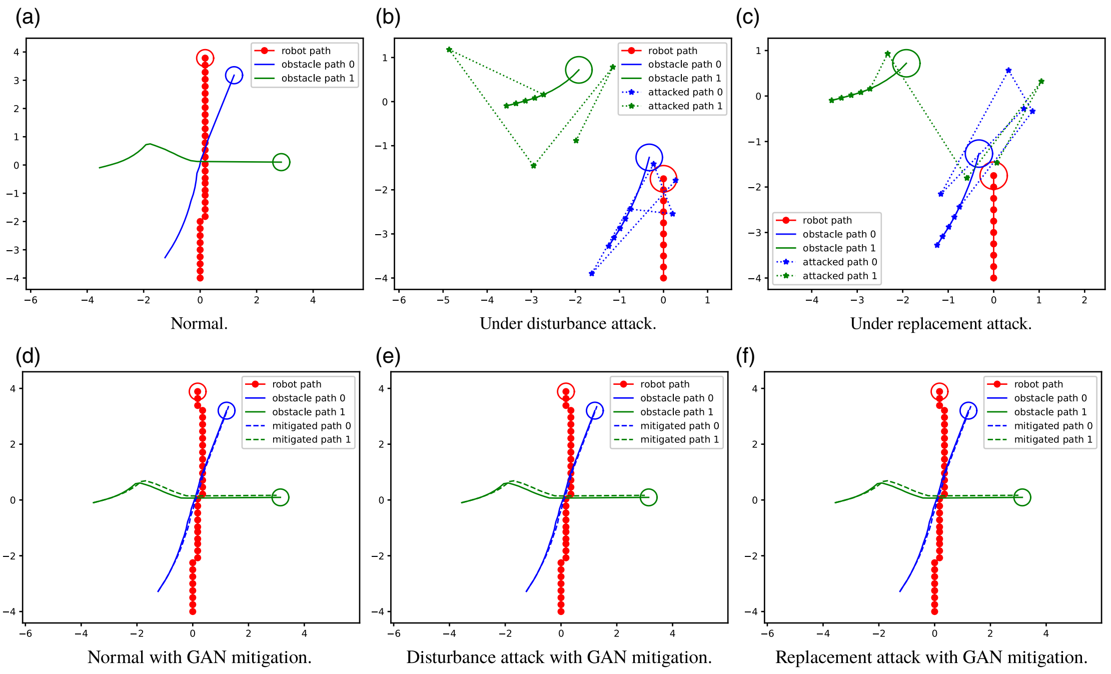


Figure 8. *An illustrative example to show the effectiveness of* `ObsGAN-DRL`*, where the circles are the end positions of the robot and the obstacles.*

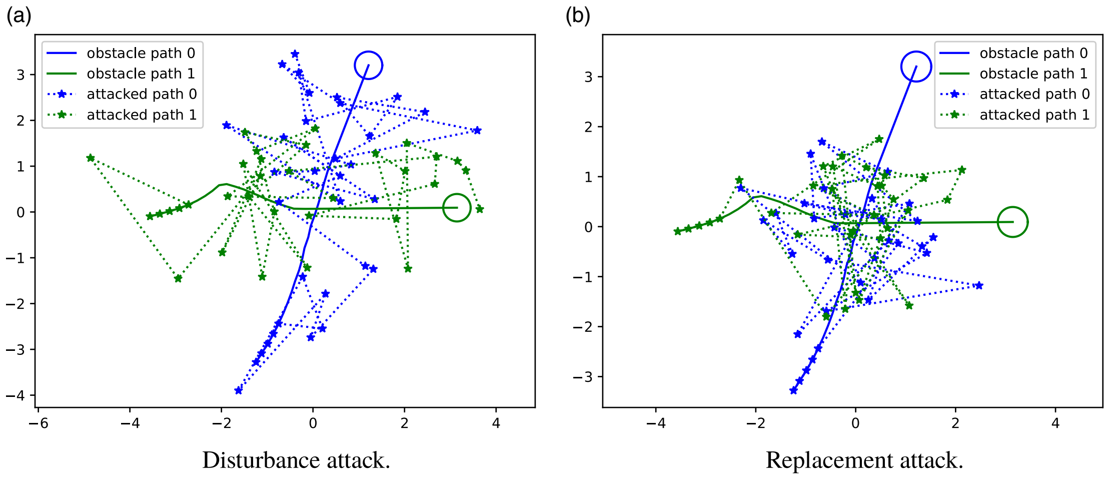


Figure 9. *The attacks generated in Fig. 8.*

4.3. Performance comparison with other mitigation methods

Currently, there is little work on the mitigation of OLAs in terms of obstacles. Hence, in this section, we show the performance comparison with the KF [37] method and the PF [39] method. In detail, KF combines the sensor readings and the predicted positions to provide more accurate position estimations. PF uses a set of weighted particles to increase localization robustness and accuracy. We use Matlab's System Identification Toolbox to generate the dynamics for the prediction stage of KF. For PF, we initialize 50 particles using the Monte Carlo method. Fig. 10 shows the comparison of the average success rate for three mitigation methods in each test set. From the results, we can find that the proposed GAN

Table III. *Significance test for the deep reinforcement learning (DRL) model and `ObsGAN-DRL` under different attacks.*

Attack	Method	Mean	Variance	T-Test
Disturbance	w/o GAN	0.683	5.46E-5	3.86E-24
	w/ GAN	0.955	5.75E-5	
Replacement	w/o GAN	0.522	1.03E-4	1.82E-27
	w/GAN	0.963	2.17E-5	

Table IV. *Average success rate for test cases with 1–10 obstacles under different attack strengths.*

		Method			
Attack Strength	# Obstacles	Disturbance	Replacement	Disturbance+GAN	Replacement+GAN
N(0,1)	1-4	0.78	0.739	0.987	0.992
	5	0.686	0.521	0.976	0.985
	6-9	0.67	0.436	0.946	0.959
	10	0.596	0.392	0.911	0.915
	avg.	0.683	0.522	0.955	0.963
N(0,2)	1-4	0.91	0.75	0.987	0.991
	5	0.83	0.631	0.974	0.983
	6-9	0.761	0.55	0.947	0.959
	10	0.625	0.482	0.912	0.916
	avg.	0.782	0.603	0.955	0.962
Uniform(−4,4)	1-4	0.96	0.917	0.976	0.991
	5	0.891	0.785	0.966	0.984
	6-9	0.803	0.749	0.948	0.957
	10	0.797	0.671	0.905	0.915
	avg.	0.838	0.781	0.949	0.962

method outperforms the other two. Specifically, compared with KF, the average success rate of the GAN method increases from 0.694 to 0.955 for disturbance, and increases from 0.693 to 0.963 for replacement. Compared with PF, the GAN method also shows a significant performance improvement (0.596 vs 0.955 for disturbance and 0.600 vs 0.963 for replacement). Table V shows the performance of the three mitigation methods in each group in terms of the average success rate. From the table, we can find that under KF, the performance reduces significantly when the number of obstacles increases, from 0.796 to 0.625 under disturbance attacks and from 0.790 to 0.619 under replacement attacks, while our GAN method can keep a relatively stable success rate. For PF, a similar diminishing trend can be observed, decreasing from 0.663 to 0.561 under disturbance attacks and from 0.666 to 0.560 under replacement attacks when the number of obstacles increases.

To investigate the reason for the outperformance of the GAN model, we further calculate the mitigation error of three mitigation methods. The results show that the GAN method significantly outperforms the other two methods, achieving the lowest average mitigation errors of 0.032 and 0.022 for disturbance attack and replacement attack, respectively. The reason is that by precisely correcting the attacked positions via the well-trained generator, the positions received by the DRL model are closer to the real ones. Therefore, `ObsGAN-DRL` can generate better motion commands.

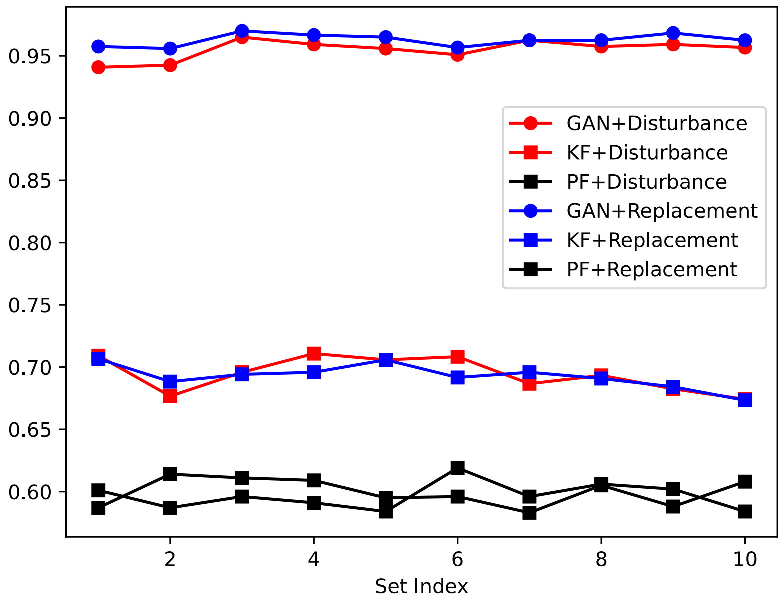


Figure 10. *Comparison of success rates of the generative adversarial network, Kalman filter, and particle filter mitigation strategies in each test set.*

Finally, to further validate the performance of `ObsGAN-DRL`, we perform t-test for equal mean checking. The results are shown in Table VI, which indicates the results are significant. It means that `ObsGAN-DRL` can achieve better performance than KF-based and PF-based methods.

4.4. Mitigation performance with other motion planning methods

Finally, we evaluate the compatibility of the GAN-based mitigation with other motion planning algorithms. In this paper, we select ORCA [45] and SARL [15]. The former is a conventional motion planning algorithm. In ORCA, each robot takes the responsibility of avoiding pairwise collisions evenly, and then the optimal action for each agent is determined by solving a low-dimensional linear program. The latter is a DRL-based method, which contains four MLPs: the first one is an embedding model, which transfers the input state to an embedding vector; the second is an attention model, which takes the embedding vector as input and computes the attention score for each obstacle; the third one is a feature module, which generates a feature vector for each obstacle; and the last is the value network, which takes the weighted feature and the robot's state as input to compute the values. The architecture of each model can be found in ref. [15]. The parameters for the training are the same as those given in Section 4.1.

Fig. 11 shows the success rates of either method in each test set. For ORCA, the GAN-based mitigation strategy improves the average success rate from 0.172 to 0.908 under the disturbance attack, and from 0.15 to 0.91 under the replacement attack. For SARL, the GAN-based mitigation strategy improves the average success rate from 0.66 to 0.95 under the disturbance attack, and from 0.446 to 0.956 under the replacement attack. We can find that on one hand, learning-based methods are more robust against OLAs than conventional methods; on the other hand, our proposed mitigation method can significantly improve the success rates for both conventional and learning methods. Hence, our GAN-based mitigation strategy can be compatible with other motion planning methods.

4.5. Experiments in Gazebo

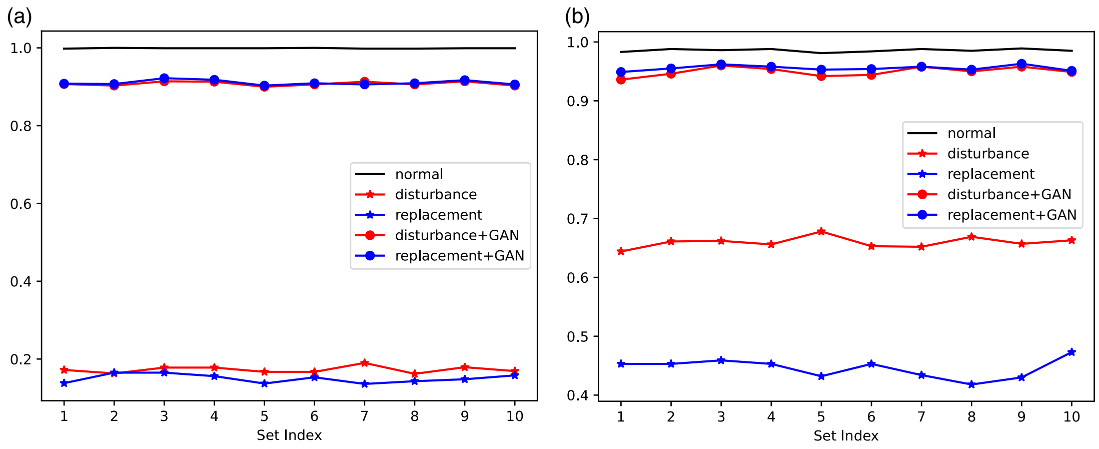
We evaluate our algorithm on *RotorS*, a well-established and high-fidelity Gazebo simulator. Gazebo is the mainstream open-source platform that can accurately reflect the physical characteristics of real-world

Table V. *Comparison of average success rates in each group under different mitigation strategies.*

Attack	**Method**	**1–4**	**5**	**6–9**	**10**	**Average Success Rate**	**Average Mitigation Error**
Disturbance	GAN	0.987	0.976	0.946	0.911	0.955	0.032
	KF	0.796	0.700	0.657	0.625	0.694	0.419
	PF	0.663	0.574	0.588	0.561	0.596	0.778
Replacement	GAN	0.992	0.985	0.959	0.915	0.963	0.022
	KF	0.790	0.702	0.659	0.619	0.693	0.661
	PF	0.666	0.575	0.597	0.560	0.600	0.834

Table VI. *Significance test for generative adversarial network (GAN), Kalman filter (KF), and particle filter (PF) mitigation strategies under different attacks.*

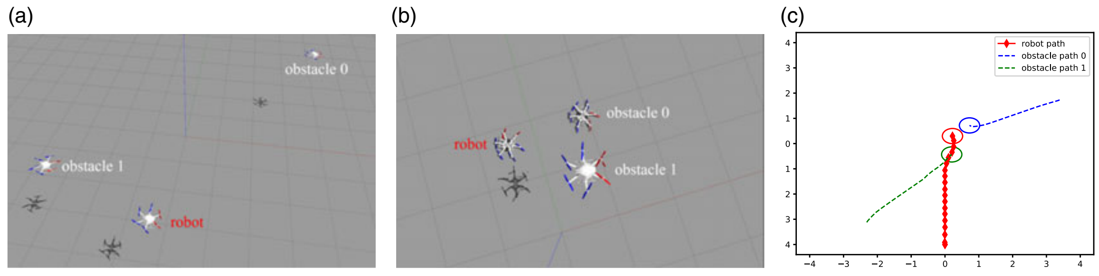
Attack	**Mitigation**	**Mean**	**Variance**	**T-Test**
Disturbance	GAN/KF	0.955/0.694	5.75E-5/1.74E-4	1.101E-17
	GAN/PF	0.955/0.596	5.75E-5/8.40E-5	3.286E-24
Replacement	GAN/KF	0.963/0.693	2.17E-5/8.6E-5	3.101E-24
	GAN/PF	0.963/0.600	2.17E-5/1.08E-04	3.286E-24



ORCA. SARL.

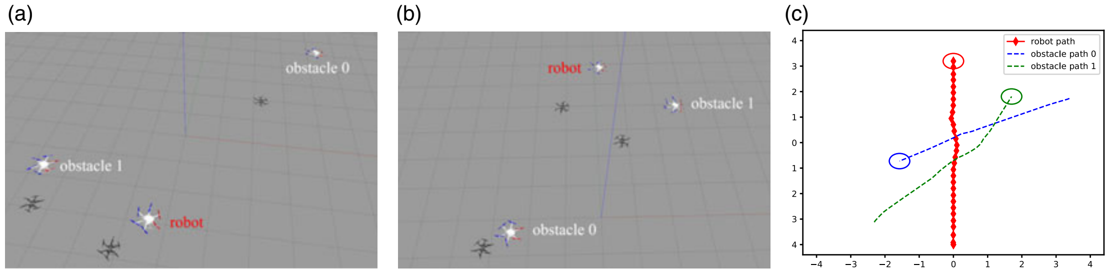
Figure 11. *The success rates of the generative adversarial network mitigation strategy with different motion planning algorithms.*

robots. Three *AscTec Firefly* drones are developed to simulate the robot and obstacles in an environment. They can communicate via the topic subscription and publication in ROS. Each drone is simulated with a ROS node deployed on Ubuntu 18.04 with ROS Melodic, to execute the motion planning algorithms and generate motion commands. The safety radius of each drone is 0.3 m, and the robot drone is required to move from $(0, -4)$ to $(0, 4)$. Note that since the rotation of rotors will cause changes in the surrounding airflow, the safety radius is larger than the physical radius of the drones. All videos of simulations can be found at https://obsgan-drl.github.io/.



Initial state. Collision state. Traveled trajectory.

Figure 12. *RotorS experiments with two obstacles under the disturbance attack. The robot causes a collision with the obstacle and falls completely.*



Initial state. Target state. Traveled trajectory.

Figure 13. *RotorS experiments with two obstacles under the disturbance attack and mitigation. The robot arrives at the target position successfully.*

4.5.1. Experiments with disturbance attack

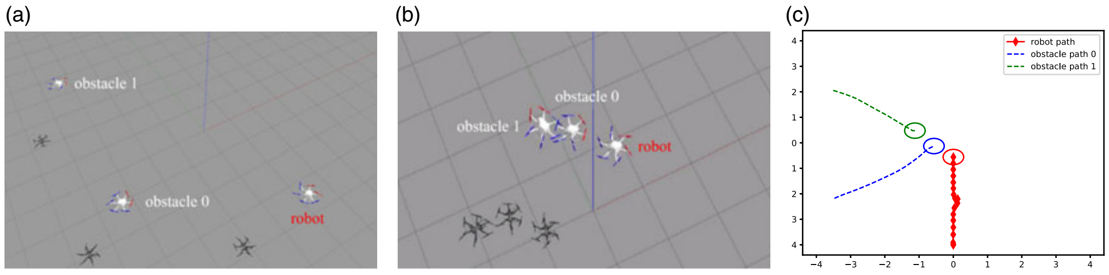
In this scenario, the two obstacle drones are required to move from (3.389, 1.731) to (−2.067, −0.997) and from (−2.313, −3.121) to (1.939, 2.319), respectively. Fig. 12a shows the initial states of three drones. As shown in Fig. 12b, due to the disturbance attack, the robot drone causes a collision with obstacle 0 and falls to the ground. Fig. 12c shows the real trajectories of the three drones. As shown in Figs. 13a–13c, with the proposed GAN-based mitigation strategy, the robot drone can correct attacked positions of the surrounding obstacles and navigate itself to its target successfully.

4.5.2. Experiments with replacement attack

In this scenario, the two obstacle drones are required to move from (−3.471, −2.182) to (2.651, 1.628) and from (−3.510, 2.054) to (2.171, −1.354), respectively. Figure 14a shows the initial states of three drones. As shown in Fig 14b, due to the replacement attack, the robot drone affected the rotors of obstacle drone 0 first, and then obstacle drone 0 affected the rotors of obstacle drone 1, losing the stability of the three drones. Figure 14c shows the traveled trajectories of three drones. From Figs. 15a–15c, we can find that the GAN-based mitigation strategy can deal with the attack successfully and navigate the robot drone to its target.

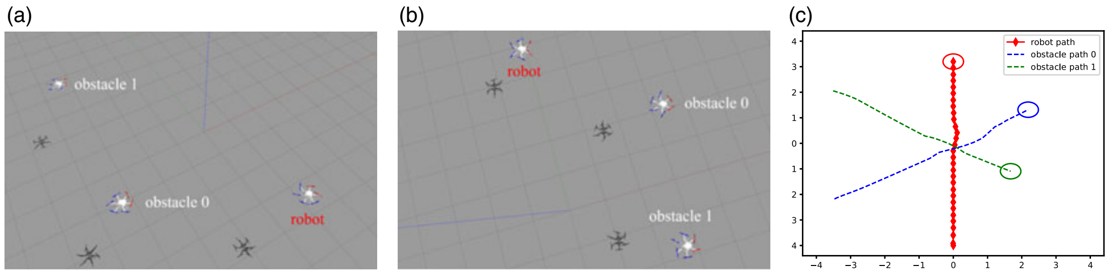
4.6. Discussion

In this paper, we proposed a GAN-based strategy to deal with position attacks against the surrounding obstacles. Our simulation experiments show that the proposed method can deal with such attacks efficiently. However, it also suffers from some threats concerning the training of the GAN model. On one hand, the available attack data might be limited in terms of the types and diversity of attacks in the real



Initial state. Collision state. Traveled trajectory.

Figure 14. *RotorS experiments with two obstacle robots under the replacement attack. The rotors of the robot and the obstacle cause a collision.*



Initial state. Target state. Traveled trajectory.

Figure 15. *RotorS experiments with two obstacle robots under the replacement attack and mitigation. The robot arrives at its target successfully.*

world, and it is challenging to collect comprehensive datasets that cover the entire spectrum of potential attacks. They will affect the generalization of the trained GAN model. To mitigate this limitation, we focus on the final influence of attacks, that is, the value of obstacle positions, which can reduce the influence of attack types and diversity; we also implement two ways to generate attacked locations. On the other hand, our attack data are generated from a simulation environment, which introduces the sim-to-real gap. Therefore, the GAN model trained in simulations may not perform as expected when deployed in the real world. To mitigate this limitation, we generate diverse data in simulations to enhance the model's ability to handle real-world variations.

5. Conclusion

In this paper, we propose a detector-agnostic method `ObsGAN-DRL` based on GAN and DRL models to mitigate OLAs in environments with a varying number of obstacles. The proposed method contains a security module, which leverages the GAN model to generate approximate accurate positions of the surrounding obstacles, and a functionality module, which leverages the LSTM-guided DRL method to deal with a varying number of obstacles and generate collision-free commands. The results show that the proposed can mitigate OLAs with good performance and compatibility.

In the future, we will integrate `ObsGAN-DRL` with more DRL methods and compare their performance and scopes of application. We will also study more complex attack scenarios and design unified attack mitigation methods to enhance the security and robustness of mobile robot systems. In addition, we will focus on obtaining more effective and robust motion planning methods.

Author contributions. All authors conceived and designed the study. Fenghua Wu and Yuan Zhou implemented the research, completed the computational experiments, and wrote the manuscript. Wenbing Tang developed the GAN-based attack mitigation

module, completed the simulation experiments, and wrote the manuscript. Shang-Wei Lin, Zuohua Ding, and Yang Liu monitored the research and edited the manuscript.

Financial support. This work was supported in part by the National Research Foundation, Singapore, and DSO National Laboratories under the AI Singapore Programme (AISG Award No: AISG2-GC-2023-008), the NRF Investigatorship NRF-NRFI06-2020-0001, Academic Research Fund Tier 2 by Ministry of Education in Singapore under Grant Nos. MOE-T2EP20120-0004, the Ministry of Education, Singapore under its Academic Research Fund Tier 2 (Proposal ID: T2EP20223-0043; Project ID: MOE-000613-00), the Natural Science Foundation of China under Grant Nos. 61972150 and 62132014, and Zhejiang Provincial Key Research and Development Program of China under Grant 2022C01045. Any opinions, findings and conclusions, or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Ministry of Education, Singapore.

Competing interests. The authors declare that no conflicts of interest exist.

Ethical approval. None.

References

[1] B. Hichri, A. Gallala, F. Giovannini and S. Kedziora, "Mobile robots path planning and mobile multirobots control: A review," *Robotica* **40**(12), 4257–4270 (2022).
[2] A. Krnjak, I. Draganjac, S. Bogdan, T. Petrović, D. Miklić and Z. Kovačić, "Decentralized Control of Free Ranging AGVs in Warehouse Environments," **In:** *IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, USA (2015) pp. 2034–2041.
[3] R. Gonzalez, M. Kloetzer and C. Mahulea, "Comparative Study of Trajectories Resulted from Cell Decomposition Path Planning Approaches," **In:** *International Conference on System Theory, Control and Computing (ICSTCC)*, Sinaia, Romania (2017) pp. 49–54.
[4] S. Kemna, J. G. Rogers, C. Nieto-Granda, S. Young and G. S. Sukhatme, "Multi-Robot Coordination Through Dynamic Voronoi Partitioning for Informative Adaptive Sampling in Communication-Constrained Environments," **In:** *IEEE International Conference on Robotics and Automation (ICRA)*, Singapore (2017) pp. 2124–2130.
[5] S. M. LaValle, Rapidly-exploring random trees: A new tool for path planning, Iowa State University, IA, Ames, (Oct. 1998). Tech. Rep. TR 98-11.
[6] P. Vlantis, C. Vrohidis, C. P. Bechlioulis and K. J. Kyriakopoulos, "Robot Navigation in Complex Workspaces Using Harmonic Maps," **In:** *IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, Australia (2018) pp. 1726–1731.
[7] J. v. d. Berg, S. J. Guy, M. Lin and D. Manocha, "Reciprocal n-Body Collision Avoidance," **In:** *Robotics Research*, (Springer, 2011) pp. 3–19.
[8] Y. Zhou, H. Hu, Y. Liu, S.-W. Lin and Z. Ding, "A real-time and fully distributed approach to motion planning for multirobot systems," *IEEE Trans Syst Man Cybern Syst* **49**(12), 2636–2650 (2017).
[9] K. Jiao, J. Chen, B. Xin, L. Li, Y. Zheng and Z. Zhao, "Three-dimensional path planning with enhanced gravitational search algorithm for unmanned aerial vehicle," *Robotica*, 1–35 (2024). doi: 10.1017/S0263574724000869.
[10] W. Tang, Y. Zhou, T. Zhang, Y. Liu, J. Liu and Z. Ding, "Cooperative collision avoidance in multirobot systems using fuzzy rules and velocity obstacles," *Robotica* **41**(2), 668–689 (2023).
[11] Y. Tan, J. Ouyang, Z. Zhang, Y. Lao and P. Wen, "Path planning for spot welding robots based on improved ant colony algorithm," *Robotica* **41**(3), 926–938 (2023).
[12] Y. F. Chen, M. Liu, M. Everett and J. P. How, "Decentralized Non-Communicating Multiagent Collision Avoidance with Deep Reinforcement Learning," **In:** *IEEE International Conference on Robotics and Automation (ICRA)*, Singapore (2017) pp. 285–292.
[13] Y. F. Chen, M. Everett, M. Liu and J. P. How, "Socially Aware Motion Planning with Deep Reinforcement Learning," **In:** *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, Canada (2017) pp. 1343–1350.
[14] M. Everett, Y. F. Chen and J. P. How, "Motion Planning among Dynamic, Decision-Making Agents with Deep Reinforcement Learning," **In:** *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain (2018) pp. 3052–3059.
[15] C. Chen, Y. Liu, S. Kreiss and A. Alahi, "Crowd-Robot Interaction: Crowd-Aware Robot Navigation with Attention-Based Deep Reinforcement Learning," **In:** *International Conference on Robotics and Automation (ICRA)*, Montreal, QC, Canada (2019) pp. 6015–6022.
[16] L. Xu, F. Wu, Y. Zhou, H. Hu, Z. Ding and Y. Liu, "Criticality-Guided Deep Reinforcement Learning for Motion Planning," **In:** *China Automation Congress (CAC)*, Kunming, China, (2021) pp. 3378–3383.
[17] S. H. Semnani, H. Liu, M. Everett, A. De Ruiter and J. P. How, "Multi-agent motion planning for dense and dynamic environments via deep reinforcement learning," *IEEE Robot Autom Lett* **5**(2), 3221–3226 (2020).
[18] N. Khlif, K. Nahla and B. Safya, "Reinforcement learning with modified exploration strategy for mobile robot path planning," *Robotica* **41**(9), 2688–2702 (2023).

[19] J. Bao, G. Zhang, Y. Peng, Z. Shao and A. Song, "Learn multi-step object sorting tasks through deep reinforcement learning," *Robotica* **40**(11), 3878–3894 (2022).

[20] Z. Sun, S. Balakrishnan, L. Su, A. Bhuyan, P. Wang and C. Qiao, "Who is in control? practical physical layer attack and defense for mmwave-based sensing in autonomous vehicles," *IEEE Trans Inf Forensics Secur* **16**, 3199–3214 (2021).

[21] Z. Hong, X. Li, Z. Wen, L. Zhou, H. Chen and J. Su, "Esp spoofing: Covert acoustic attack on mems gyroscopes in vehicles," *IEEE Trans Inf Forensics Secur* **17**, 3734–3747 (2022).

[22] S. Dasgupta, M. Rahman, M. Islam and M. Chowdhury, "A sensor fusion-based gnss spoofing attack detection framework for autonomous vehicles," *IEEE Trans Intell Transp Syst* **23**(12), 23559–23572 (2022).

[23] J. Liu and J.-M. Park, ""seeing is not always believing": Detecting perception error attacks against autonomous vehicles," *IEEE Trans Dependable Secure Comput* **18**(5), 2209–2223 (2021).

[24] A. Khazraei, H. Meng and M. Pajic, "Stealthy Perception-Based Attacks on Unmanned Aerial Vehicles," **In:** *IEEE International Conference on Robotics and Automation (ICRA)*, London, UK (2023) pp. 3346–3352.

[25] W. Fu, J. Qin, Y. Shi, W. X. Zheng and Y. Kang, "Resilient consensus of discrete-time complex cyber-physical networks under deception attacks," *IEEE Trans Ind Electron* **16**(7), 4868–4877 (2020).

[26] G. Deng, Y. Zhou, Y. Xu, T. Zhang and Y. Liu, "An Investigation of Byzantine Threats in Multi-Robot Systems," **In:** *International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, San Sebastian, Spain (2021) pp. 17–32.

[27] W. Wang, Z. Han, K. Liu and J. Lü, "Distributed adaptive resilient formation control of uncertain nonholonomic mobile robots under deception attacks," *IEEE Trans Circuits Syst I Regul Pap* **68**(9), 3822–3835 (2021).

[28] Y. Wang, C. Zhao, H. Wang, J. Zhang and D. Wang, "Secure Localization of Autonomous Articulated Vehicles: Attack Detection and Recovery," **In:** *International Conference on Intelligent Transportation Systems (ITSC)*, Bilbao, Bizkaia, Spain (2023) pp. 2914–2919.

[29] M. Castellano-Quero, J.-A. Fernández-Madrigal and A. García-Cerezo, "Improving bayesian inference efficiency for sensory anomaly detection and recovery in mobile robots," *Expert Syst Appl* **163**, 113755 (2021).

[30] F. Akowuah, R. Prasad, C. O. Espinoza and F. Kong, "Recovery-by-Learning: Restoring Autonomous Cyber-Physical Systems from Sensor Attacks," **In:** *International Conference on Embedded and Real-time Computing Systems and Applications (RTCSA)*, Houston, TX, USA (2021) pp. 61–66.

[31] Y. Wang, N. Masoud and A. Khojandi, "Real-time sensor anomaly detection and recovery in connected automated vehicle sensors," *IEEE Trans Intell Transp Syst* **22**(3), 1411–1421 (2020).

[32] Q. Liu, Y. Mo, X. Mo, C. Lv, E. Mihankhah and D. Wang, "Secure Pose Estimation for Autonomous Vehicles Under Cyber Attacks," **In:** *IEEE Intelligent Vehicles Symposium (IV)*, Paris, France (2019) pp. 1583–1588.

[33] S. Yousuf and M. B. Kadri, "Information fusion of GPS, INS and odometer sensors for improving localization accuracy of mobile robots in indoor and outdoor applications," *Robotica* **39**(2), 250–276 (2021).

[34] Y. Xu, X. Han, G. Deng, J. Li, Y. Liu and T. Zhang, "Sok: Rethinking Sensor Spoofing Attacks Against Robotic Vehicles from a Systematic View," **In:** *European Symposium on Security and Privacy (EuroS& P)*, Delft, Netherlands (2023) pp. 1082–1100.

[35] F. Alkhawaja, M. A. Jaradat and L. Romdhane, "Low-cost depth/IMU intelligent sensor fusion for indoor robot navigation," *Robotica* **41**(6), 1689–1717 (2023).

[36] M. Elsisi, M. Altius, S.-F. Su and C.-L. Su, "Robust Kalman filter for position estimation of automated guided vehicles under cyberattacks," *IEEE Trans Instrum Meas* **72**, 1–12 (2023).

[37] C. Suliman, C. Cruceru and F. Moldoveanu, "Mobile robot position estimation using the Kalman filter," *Acta Maris Seria Technol* **6**, 75 (2009).

[38] N. Gosala, A. Bühler, M. Prajapat, C. Ehmke, M. Gupta, R. Sivanesan, A. Gawel, M. Pfeiffer, M. Bürki, I. Sa, R. Dubé and R. Siegwart, "Redundant Perception and State Estimation for Reliable Autonomous Racing," **In:** *IEEE International Conference on Robotics and Automation (ICRA)*, Montreal, Canada (2019) pp. 6561–6567.

[39] Q.-B. Zhang, P. Wang and Z.-H. Chen, "An improved particle filter for mobile robot localization based on particle swarm optimization," *Expert Syst Appl* **135**, 181–193 (2019).

[40] J.-P. A. Yaacoub, H. N. Noura, O. Salman and A. Chehab, "Robotics cyber security: Vulnerabilities, attacks, countermeasures, and recommendations," *Int J Inf Secur* **21**(1), 115–158 (2022).

[41] L. Zhou, V. Tzoumas, G. J. Pappas and P. Tokekar, "Distributed attack-robust submodular maximization for multirobot planning," *IEEE Trans Robot* **38**(5), 3097–3112 (2022).

[42] L. Zhou and V. Kumar, "Robust Multi-Robot Active Target Tracking Against Sensing and Communication Attacks," **In:** *American Control Conference (ACC)*, Atlanta, Georgia, USA (2022) pp. 4443–4450.

[43] Y. Zhao, D. Sanán, F. Zhang and Y. Liu, "Formal Specification and Analysis of Partitioning Operating Systems by Integrating Ontology and Refinement," *IEEE Trans Industr Inform* **12**(4), 1321–1331 (2016).

[44] Y. Han, I. H. Zhan, W. Zhao, J. Pan, Z. Zhang, Y. Wang and Y.-J. Liu, "Deep reinforcement learning for robot collision avoidance with self-state-attention and sensor fusion," *IEEE Robot Autom Lett* **7**(3), 6886–6893 (2022).

[45] J. Van Den Berg, S. J. Guy, M. Lin and D. Manocha, "Reciprocal n-Body Collision Avoidance," **In:** *Robotics Research: The 14th International Symposium ISRR* (Berlin, Heidelberg, Springer, 2011) pp. 3–19.

Cite this article: F. Wu, W. Tang, Y. Zhou, S.-W. Lin, Z. Ding and Y. Liu (2024). "Robust motion planning for mobile robots under attacks against obstacle localization", Robotica **42**, 2781–2800. https://doi.org/10.1017/S0263574724001115