arXiv:2301.02061v1 [math.OC] 5 Jan 2023

不确定环境下多智能体系统分层屏障覆盖的分布式控制策略

Pengyang Fan、翟超 *

2022年8月

摘要

本文提出一种分布式多层环形屏障覆盖算法。为实现单层环形屏障覆盖，本文提出一种能够使监测概率最大化的分布式单层环形屏障覆盖算法。针对单层屏障覆盖的安全风险，本文设计了多层屏障之间的分布式调整机制，并将其与单层环形屏障覆盖算法相结合，提出一种分布式多层环形屏障覆盖算法。此外，本文对所提出的算法进行了理论分析，以证明其有效性和必要性。最后，通过数值仿真和实验验证了该算法。

1 引言

多智能体系统（MASs）由在环境中相互交互的智能体组成。每个智能体都是一个系统，而 MASs 则是由大量智能体组合而成、能够实现整体行为或活动的系统。智能体可以是自然生物 [1]，人工机器人或移动传感器[2]。MASs 旨在采用分布式方法解决一些规模庞大且复杂的问题。每个智能体都是能够独立感知环境、处理信息、进行通信、学习和做出决策的独立个体。由于每个智能体都采用独立策略，多智能体系统的协同控制

*Pengyang Fan 和翟超任职于中国地质大学自动化学院（中国武汉，430074），并隶属于湖北省复杂系统先进控制与智能自动化重点实验室和地球探测智能技术教育部工程研究中心（中国武汉，430074）。通讯作者：翟超（邮箱：zhaichao@amss.ac.cn）。

对于整个系统实现共同目标至关重要，因此这一领域吸引了众多学者开展研究。

多智能体覆盖控制是多智能体协调控制领域的热门研究课题。多智能体覆盖控制是指一组具备移动、通信、计算和学习能力的智能体，在给定或不确定的区域内，以分布式方式感知环境并执行指定任务，例如搜索与救援、导弹拦截、监测、扫掠等[5]。根据智能体所覆盖的区域，多智能体覆盖控制可分为区域覆盖、扫掠覆盖和屏障覆盖。区域覆盖是一系列操作，在这些操作中，每个智能体通过通信、计算和协调，确定其在区域内的最优状态，并借助某些控制科学方法达到该状态。最经典的方法是基于Voronoi划分的多智能体覆盖算法[3]，：每个智能体通过通信将一个凸区域划分为若干子区域，并采用移动至子区域质心的策略，以最大限度地提高覆盖质量。该方法能够最大限度地覆盖凸区域。在此基础上，许多学者发现了诸多问题，并提出了一些解决方案。例如，为解决非凸区域问题，一些学者基于测地Voronoi划分算法[4]，提出了适用于非凸区域的Voronoi质心覆盖算法；为解决时变密度函数问题，一些学者基于控制屏障函数[6]，提出了动态环境下的Voronoi质心覆盖方法；为解决不确定环境下的覆盖问题，一些学者基于贝叶斯估计[7]，提出了不确定环境下的Voronoi质心覆盖方法。扫掠覆盖不仅要求智能体到达指定区域，还要求智能体能够遍历整个区域，从而实现环境清理。例如，一些学者基于等任务划分方法，实现了某类区域的等任务扫掠覆盖，从而提高整体效率[8]。还有一些学者提出了基于温度场方法的多智能体扫掠覆盖算法[9]。此外，文献[10] 将Voronoi分割与温度场方法相结合，设计了一种分布式覆盖方法，使每个智能体的工作量相同。

多智能体屏障覆盖是指一组智能体在线上的覆盖，通常用于监测生物或物体是否越过该线，或拦截试图越线的物体。文献 [11] 提出了屏障覆盖和 k-屏障覆盖的定义。k-屏障覆盖进一步分为弱 k-屏障覆盖和强 k-屏障覆盖[11][12][13]。强 k-屏障覆盖意味着，无论入侵者通过何种路径进入或穿过目标区域，都能

被至少 k 个智能体探测到。此外，Chen 等人提出了局部屏障覆盖的概念。与全局围栏覆盖相比，这种方法可以减少智能体数量，也适用于一般情况[14]。然而，局部屏障覆盖存在安全风险，因为入侵者可能在不被探测到的情况下穿过该区域。目前已有许多用于屏障覆盖和 k-屏障覆盖的算法。例如，文献 [19] 提出的基于覆盖的方法能够为拦截入侵物分配入侵概率，从而实现对目标的保护。此外，研究人员设计了一种分布式算法，可在两个地标之间实现均匀的屏障覆盖[15]。Ban 等人采用基于网格的方法，研究了开放带状区域中移动传感器网络的强 k-屏障覆盖问题 [16]。然而，该算法只能实现两点之间直线上的覆盖，无法实现曲线或闭曲线上的覆盖。在实际应用中，如果需要对区域内的目标进行全方位保护或监测，就需要覆盖整个区域边界。对于封闭的目标区域，还需要在闭合带状区域内布置智能体。因此，Binay 等人设计了一种算法，使智能体移动到简单多边形的边界上，从而保护多边形内部区域[17]。此外，还有一种针对圆的屏障覆盖算法，可在有限通信条件下使智能体均匀分布在圆上[18]。但是，圆属于凸区域，如何在非凸区域的边界上实现屏障覆盖，正是本研究的出发点。此外，仅覆盖边界意味着一旦入侵者突破这一层，就会进入我们需要保护的区域，而系统也将失去监测入侵者的手段，这会增加安全风险。从安全角度来看，k-屏障覆盖比单层屏障覆盖更安全。

为此，我们首先设计了能够在一类非凸区域边界上执行屏障覆盖的算法。该算法应用于入侵者监测场景，并利用区域划分为每个智能体分配一个监测区域，这最终可能导致监测概率达到局部最大值。因此，我们希望设计一种具有多层屏障覆盖功能的多智能体控制算法来解决这一问题。当入侵者突破某一层时，内部仍有若干监测层可以继续监测入侵者。本文旨在设计一种分布式多智能体屏障覆盖算法，实现多层屏障覆盖，并能够自主调整每层的智能体数量，以优化整体监测质量

。本文的贡献如下。

1. 设计一种针对一类非凸区域、能够最大限度监测入侵者的多智能体屏障覆盖算法。2. 开发一种用于调整每层智能体数量的分布式调整机制，以优化多智能体系统的监测概率。3. 结合单层围栏覆盖算法与多层智能体数量的分布式调整机制，提出多层屏障覆盖算法。

本文其余部分的结构安排如下：第2节首先介绍一种适用于非凸区域边界的单层屏障覆盖算法，随后给出多层覆盖区域中智能体数量的分布式调整机制和多层屏障覆盖算法。第3节对第2节提出的单层屏障覆盖算法和多层屏障覆盖算法进行理论验证，并介绍将我们的算法应用于圆形区域的情况。第4节对算法进行仿真，并在Robotarium上开展实验验证。最后，我们在第5节总结全文。

2 问题描述

本节将介绍分布式多智能体屏障覆盖算法。考虑一个闭曲线区域 D，它可以用极坐标表示，圆心 D 表示为 O。不失一般性，我们可以将圆心设为原点，即 $O = (0, 0)$。圆形区域的边界 D 表示为 ∂D。闭曲线区域的半径表示为 $R(\theta)$ $\theta \in [0, 2\pi)$。

2.1 单层屏障覆盖

在入侵者监测应用中，多层屏障覆盖比单层屏障覆盖更有效。因此，本文提出一种多层屏障覆盖算法。由于该算法基于单层屏障覆盖算法，本小节首先介绍单层屏障覆盖算法。

在第 k 层，有 N_k 个能够通信和监控的移动智能体。第 k 层可表示为 $R_k(\theta)$ $\theta \in [0, 2\pi)$。我们假设，同一层上的智能体都能够彼此通信

。我们用 I_{N_k} 表示该层上的智能体数量，$I_{N_k} = \{1, 2, ..., N_k\}$。我们用 $\rho(\theta)$ 表示该层上的每个点被入侵者入侵的概率。我们用 $P = \{p_1, p_2, ..., p_{N_k}\}$ 表示智能体的位置。我们定义智能体 i_k 相对于圆心 O 的角度，记作 φ_{i_k}、$i_k = 1$、2、...、N_k。我们用 $f(d(\varphi_{i_k}, \theta))$ 表示智能体 i_k 探测到入侵者的概率模型，其中 $d(\varphi_{i_k}, \theta)$ 是关于 φ_{i_k} 和 θ 的距离函数，且该距离函数是Lipschitz连续的，并且 $d(\varphi_{i_k}, \theta) \propto \delta(\varphi_{i_k}, \theta)$。函数 $\delta(\varphi_{i_k}, \theta)$ 表示为

$$\delta(\varphi_{i_k}, \theta) = \begin{cases} |\varphi_{i_k} - \theta - 2\pi| & if \quad \varphi_{i_k} - \theta > \pi \\ |\varphi_{i_k} - \theta + 2\pi| & if \quad \varphi_{i_k} - \theta \leq -\pi \\ |\varphi_{i_k} - \theta| & else \end{cases} \tag{1}$$

此外，$f(d(\varphi_{i_k}, \theta))$需要满足以下条件：

1. $f(d(\varphi_{i_k}, \theta))$是可微的。
2. $f(d(\varphi_{i_k}, \theta))$是单调递减的。

现在给出 φ_{i_k} 的计算方法如下

$$\varphi_{i_k}(t) = \Psi(p_{i_k}^T(t)). \tag{2}$$

其中，$\Psi(x, y)$是我们指定的一种运算，其计算如下

$$\Psi(x, y) = \begin{cases} \arctan(\frac{y}{x}) + \pi & x < 0 \\ \arctan(\frac{y}{x}) & y \geqslant 0 \quad x > 0 \\ \arctan(\frac{y}{x}) + 2\pi & y < 0 \quad x > 0 \\ \frac{\pi}{2} & y > 0 \quad x = 0 \\ -\frac{\pi}{2} & y < 0 \quad x = 0 \\ 0 & y = 0 \quad x = 0 \end{cases} \tag{3}$$

结合(2)和(3)，容易得知 $\varphi_{i_k} \in [0, 2\pi)$，其中 $i_k = 1$、2、...、N_k。此外，智能体具有以下编号规则

$$0 \leq \varphi_1(0) < \varphi_2(0) < ... < \varphi_{N_k}(0) < 2\pi$$

随着距离增加，智能体的检测概率会降低。因此，我们规定智能体只检测其负责的区域。因此，我们提出一种

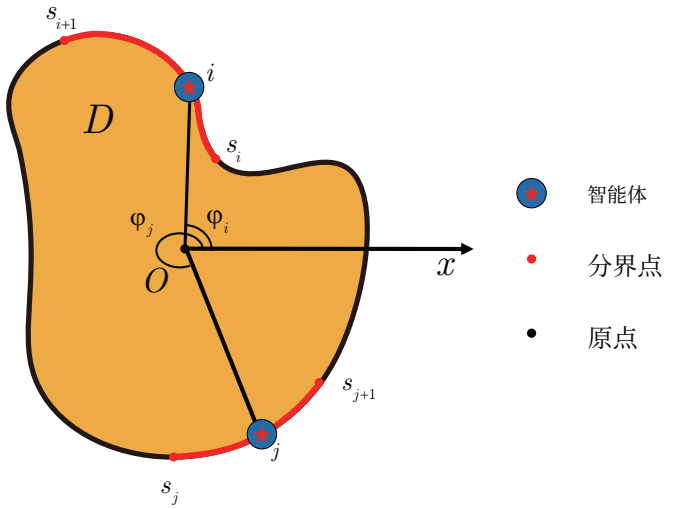

图1：覆盖区域 D、智能体通信半径和监测半径；五角星的颜色表示智能体的工作状态。

划分方法，可将层 k 划分为 N_k 个子区域。为了实现划分，我们在层 k上设置 N_k 个分割点，记为 S $=\{s_1, s_2, ..., s_{N_k}\}$， $s_{i_k} \in [0, 2\pi)$ 表示这些分割点的相位。

这些分割点将层 k 划分为 N_k 个子区域，记为 E $=\{E_1, E_2, ..., E_{N_k}\}$， E_{i_k} 表示如下

$$E_{i_k} = \begin{cases} \{(R_k(\theta), \theta) | s_{i_k} \le \theta < s_{i_k+1}\} & if \quad s_{i_k} < s_{i_k+1} \\ \{(R_k(\theta), \theta) | s_{i_k} \le \theta < 2\pi, 0 \le \theta < s_{i_k+1}\} & otherwise \end{cases} \tag{4}$$

和 $s_{N+1} = s_1$。入侵者从 E_{i_k} 入侵的概率记为 m_{i_k}， m_{i_k} 计算如下

$$m_{i_k} = \begin{cases} \int_{s_{i_k}}^{s_{i_k+1}} \rho(\theta)\, d\theta & if \quad s_{i_k} < s_{i_k+1} \\ \int_{s_{i_k}}^{2\pi} \rho(\theta)\, d\theta + \int_0^{s_{i_k+1}} \rho(\theta)\, d\theta & otherwise \end{cases} \tag{5}$$

同理，我们用 $\mathcal{T}_{i_k}$ 表示分界点的位置，如下所示：

$$\mathcal{T}_{i_k} = \begin{cases} \{(R_k(\theta), \theta) | \varphi_{i_k} \le \theta < \varphi_{i_k+1}\} & if \quad \varphi_{i_k} < \varphi_{i_k+1} \\ \{(R_k(\theta), \theta) | \varphi_{i_k} \le \theta < 2\pi, 0 \le \theta < \varphi_{i_k+1}\} & otherwise \end{cases} \tag{6}$$

其中 $\varphi_{N_k+1} = \varphi_1$。

根据全概率定律，我们可以给出

多智能体系统的监测概率，如下所示

$$H(\varphi,\mathcal{S})=\sum_{i=1}^{N}\int_{E_{i_k}} f(d(\varphi_{i_k},\theta))\rho(\theta)d\theta \tag{7}$$

不难发现，质量函数（$\varphi,\mathcal{S}$）表示的是检测到入侵者入侵的概率。对于检测入侵者的应用而言，方程（7）的值越大，系统性能越好。因此，入侵者检测问题可以转化为以下优化问题

$$\max\left(H\left(\varphi,\mathcal{S}\right)\right) \tag{8}$$

为了符合实际情况，我们用以下非完整约束运动方程表示智能体动力学方程

$$\begin{cases} x_{i_k}=r_{i_k}\cos\left(\varphi_{i_k}\right) \\ y_{i_k}=r_{i_k}\sin\left(\varphi_{i_k}\right) \\ \dot{\varphi}_{i_k}=\omega_{i_k} \\ \dot{r}_{i_k}=u_{i_k}^{r} \end{cases} \tag{9}$$

其中， r_{i_k} 表示智能体与圆心之间的距离，即 $r_{i_k}=\|p_{i_k}\|$。x_{i_k}和 y_{i_k}是 p_{i_k}的水平和垂直坐标，而 $p_{i_k}(t)=(x_{i_k}(t)\ y_{i_k}(t))^{T}$表示时间步 $t\in\mathbb{R}^{+}$时智能体的位置。 ω_{i_k} 表示智能体的角速度，稍后将对此进行介绍。$u_{i_k}^{r}$ 定义如下

$$u_{i_k}^{r}=\kappa_r\left(R(\varphi_{i_k})-r_{i_k}\right), \tag{10}$$

其中， κ_r 是一个可调参数。根据（9）和（10），可以得到智能体的动力学方程为

$$\begin{cases} \dot{x}_{i_k}=\frac{\partial x_{i_k}}{\partial r_{i_k}}\dot{r_{i_k}}+\frac{\partial x_{i_k}}{\partial \varphi_{i_k}}\dot{\varphi_{i_k}}=u_{i_k}^{r}\cos\left(\varphi_{i_k}\right)-r_{i_k}\omega_{i_k}\sin\left(\varphi_{i_k}\right) \\ \dot{y}_{i_k}=\frac{\partial y_{i_k}}{\partial r_{i_k}}\dot{r_{i_k}}+\frac{\partial y_{i_k}}{\partial \varphi_{i_k}}\dot{\varphi_{i_k}}=u_{i_k}^{r}\sin\left(\varphi_{i_k}\right)+r_{i_k}\omega_{i_k}\cos\left(\varphi_{i_k}\right) \end{cases} \tag{11}$$

对式（7）使用梯度法，可得

$$\dot{H}=\sum_{i=1}^{N}\int_{E_{i_k}}\frac{\partial f\left(d(\varphi_{i_k},\theta)\right)}{\partial\varphi_{i_k}}\rho\left(\theta\right)d\theta\cdot\omega_{i_k}+\sum_{i=1}^{N}\frac{\partial H}{\partial s_{i_k}}\dot{s}_{i_k} \tag{12}$$

为使 H 最大化，我们可以将 ω_{i_k} 设置如下

$$\omega_{i_k}=\kappa_{\omega}\int_{E_{i_k}}\frac{\partial f\left(d(\varphi_{i_k},\theta)\right)}{\partial\varphi_{i_k}}\rho\left(\theta\right)d\theta \tag{13}$$

算法1 单层屏障覆盖算法

初始化： $\kappa_{\quad r}$, κ_ω, κ_s, T^*

对于 $i_k \in I_{N_k}$，第 i_k个智能体执行如下操作

1: 对于 $t=1$： T^* 执行

2: 计算 $\varphi_{i\quad k}$ 根据(2)和(3);

3: 当 $d(\varphi_{i_k}, s_{i_k}) - d(\varphi_\alpha, s_{i_k}) < \varepsilon$ 满足时

4: 更新 $s_{i\quad k}$ 使用 (15);

5: 结束循环

6: 更新 $p_{i\quad k}$ 使用 (15)、(10) 和 (9);

7: 结束循环

其中， κ_ω 是一个可调常数，且式（13）可以保证式（12）不小于零，这意味着 H 不会减小。

我们构造如下分割点控制输入

$$\dot{s}_{i_k} = \kappa_s(d(\varphi_{i_k}, s_{i_k}) - d(\varphi_{i_k-1}, s_{i_k})) \tag{14}$$

其中 κ_s 为正数常量。为了将该算法应用于多层屏障覆盖算法，我们需要对控制输入进行一些调整。我们可以发现，对于执行屏障覆盖的智能体i_k，只需智能体 i_k-1 和智能体 i_k+1 的状态即可完成该算法。此外，智能体 i_k-1 和智能体 i_k+1 分别是智能体 i_k顺时针方向和逆时针方向上相对位置最近的智能体。因此，在多层覆盖问题中，我们用 α 和 β 表示智能体 i_k-1 和智能体 i_k+1。我们可以将该智能体的控制输入改写如下

$$\begin{aligned}
&\dot{s}_{i_k} = \kappa_s(d(\varphi_{i_k}, s_{i_k}) - d(\varphi_\alpha, s_{i_k})) \\
&\omega_{i_k} = \begin{cases} \kappa_\omega \int_{s_{i_k}}^{s_\beta} \frac{\partial f\left(d(\varphi_{i_k},\theta)\right)}{\partial \varphi_{i_k}} \rho(\theta)\, d\theta, & if \;\; s_{i_k} < s_\beta \\ \kappa_\omega \int_{s_{i_k}}^{2\pi} \frac{\partial f\left(d(\varphi_{i_k},\theta)\right)}{\partial \varphi_{i_k}} \rho(\theta)\, d\theta + \int_0^{s_\beta} \frac{\partial f\left(d(\varphi_{i_k},\theta)\right)}{\partial \varphi_{i_k}} \rho(\theta)\, d\theta. & otherwise \end{cases}
\end{aligned} \tag{15}$$

最后，我们给出表1所示的单层障碍分布式覆盖算法。在该算法中，我们确保分割点位于各智能体之间曲线的中点。由于分割点是虚拟的，因此在实际执行中，这一步非常快速。接下来将介绍多层屏障覆盖算法。

算法 **2** N_k 计算算法

1: 对于 $j = 1 : N$，执行

2: 如果 $a_j = 1$，则

3: $k = k_j$;

4: $N_k = N_k + 1$;

5: $I_{N_k} = I_{N_k} \cup \{j\}$;

6: 结束 if

7: 结束 for

2.2 多层屏障覆盖

本小节将介绍一种基于第2.1小节的分布式屏障覆盖控制算法。

考虑需要覆盖的 K^* 区域层。我们使用 $R_1(\theta)$ $R_2(\theta)$ ..., $R_{K^*}(\theta)$表示这些区域层的极坐标方程，并用 $0 < R_1(\theta) < R_2(\theta) < ... < R_{K^*}(\theta) < R_{max}$表示，对于$\theta \in (0, 2\pi]$。其中， R_{max} 为正常数。根据第2.1小节，第 k 层上的智能体数量表示为 N_k。同样，该层上所有智能体的数量表示为N_L，并且 $N_L = \sum_{k=1}^{K^*} N_k$.

我们不希望固定每层的智能体数量，而是希望找到一种分布式方法，能够自动分配每层的智能体数量。我们将此功能称为层交换。当智能体即将进行层交换时，它将获得一个目标层。当智能体移动到其目标层时，很容易发现该智能体不属于任何层。我们将这类智能体称为空闲智能体。另一方面，属于某一层的智能体称为层智能体。此外，我们认为，初始时刻智能体的状态不应存在差异，除了它们的位置不同。因此，在我们的工作中，初始时刻所有智能体都是空闲智能体。我们可以将空闲智能体视为干细胞，将层智能体视为分化细胞。空闲智能体转变为层智能体的过程类似于细胞分化。空闲智能体的数量表示为 N_F。所有智能体的数量表示为N，并且 $N = N_L + N_F$。在初始时刻， $N = N_F$。此处我们将所有智能体编号为$I_N = \{1, 2, ..., N\}$。我们使用 a_i 表示智能体 i的类型；当 $a_i = 0$时，该智能体为空闲智能体，而当 $a_i = 1$时，它是层智能体。我们使用算法 2计算每个智能体的 N_k，这构成了我们工作的基础。

这与第2.1小节中所示的类似， φ_i 而 $p_i = (x_i, y_i)^T$ 表示相位角

和智能体 i的位置。我们用 k_i 表示智能体 i的目标层，并且$k_i \in \{0, 1, 2, ..., K\}$。 $k_i = 0$表示智能体 i没有目标层。在这种情况下，为了寻找目标层，智能体 i将按如下方式移动

$$\begin{cases} \dot{x}_i = -r_i\omega_0 \sin(\varphi_i), \\ \dot{y}_i = r_i\omega_0 \cos(\varphi_i), \end{cases} \tag{16}$$

其中 ω_0 是一个常数。当 $k_i \neq 0$时，与控制输入 (16) 类似，智能体将按如下方式移动

$$\begin{cases} \dot{x}_i = \kappa_r(R(\varphi_i) - r_i)cos(\varphi_i), \\ \dot{y}_i = \kappa_r(R(\varphi_i) - r_i)sin(\varphi_i). \end{cases} \tag{17}$$

在第2.1小节中，我们已为智能体编号。然而，本小节中的智能体编号有所不同。在第 k层中， $I_{N_k} = \{j | a_j \times k_j = 1\}$。正如算法 2 中所计算的，我们用 I_{N_k} 表示第 k层的编号集合。如果所有智能体都在各层工作，则存在如下关系： $I_N = \bigcup_{k=1}^{K^*} I_{N_k}$。

当智能体靠近某一层时，需要考虑它是否可以加入该层的覆盖任务。我们用 r_k 表示第 k层的范围，且 $r_k := \{(rcos(\theta)\ rsin(\theta)) | R_k(\theta) - \Delta \leq r \leq R_k(\theta) + \Delta\}$。并且 Δ 是一个足够小的常数。当智能体进入 r_k时，我们认为该智能体靠近第 k层。此外，我们认为，智能体 i 能否进入第 k层取决于智能体 j 是否已在第 k层工作，而非智能体 i 本身。只有当智能体 j 批准其进入时，智能体 i 才能进入第 k层执行检测任务；否则，智能体 i 应尝试移动到其他层。如果第 k层中没有智能体，该智能体便可顺利进入第 k 层。

现在，我们将介绍智能体 i执行检测任务时的检测状态。智能体 i 的检测状态是判断层外智能体能否进入该层执行任务的关键变量。显而易见，当智能体承担的工作过多时，其检测能力会下降。另一方面，当某一层中的智能体数量充足时，进入该层的智能体所作出的贡献不如进入其他层的智能体。因此，我们用 c_i 表示智能体 i 的检测状态，如下所示

$$c_i = \begin{cases} 1 & \eta_i > h \\ 0 & otherwise \end{cases} \tag{18}$$

其中 $h \in (0, 1)$ 是一个可调参数，并且

$$\eta_i = \frac{\int_{E_i} f(d(\varphi_i, \theta))\rho(\theta)d\theta}{\int_{E_i} \rho(\theta)d\theta}, \tag{19}$$

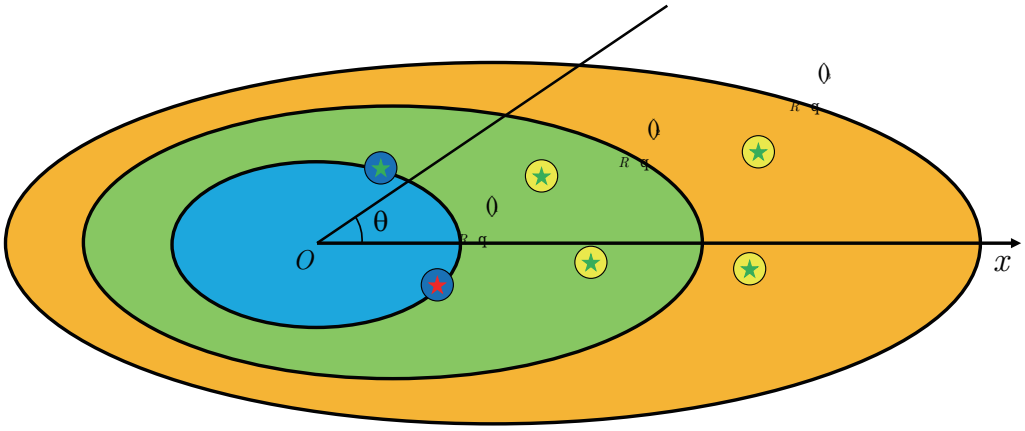

图 2：覆盖区域 D、智能体通信半径和监测半径。

η_i 可解释为任务完成率。它也可以解释为在入侵者侵入区域 E_i 的条件下被智能体 i 检测到的概率。

如图2所示，需要覆盖的区域分为三层。星形的颜色表示智能体的检测状态，圆形的颜色表示智能体是否为空闲智能体。蓝色圆形表示该智能体正在执行检测任务，黄色圆形表示该智能体为空闲智能体并正在移动到其目标层。红色星形表示该智能体不允许其他智能体进入该层。绿色星形表示该智能体允许其他智能体进入该层。如果 $c_i = 1$，星形将变为红色。

每个空闲智能体都有一个目标层，我们用 k_i 表示智能体 i的目标层。如何帮助智能体找到目标层是算法的基础。我们提出算法3来帮助智能体实现这一功能。我们的思路是让所有智能体首先以第一层为目标。如果无法进入第一层，则考虑第二层，以此类推，直到第 K^*层。在考虑是否将第 k层作为目标层时，如果第k层没有智能体，智能体 i 就会选择以第 k层为目标。如果第 k层有智能体，智能体i 则需要预测第 k 层的智能体是否允许其进入。

现在，我们需要考虑如何让智能体进入某一层。如上所述，智能体能否进入该层取决于该层中的智能体。因此，我们使用算法 4来实现这一功能。在算法 4中，为了避免智能体的相位相同，从而导致后续分界点难以设置，智能体发现相位相同时会改变自己的相位。为避免智能体改变相位时被其他智能体抢占，我们需要先设置 $a_i = 1$ 。

不难发现，算法1需要智能体 α 和智能体 β 来执行。因此，

算法3 目标层识别算法

智能体 i 执行如下操作

对 $k=1:K^*$ 执行
  如果 $N_k=0$ 则
    $k_i=k$;
  else
    对 $j\in I_{N_k}$ **do**
      如果 $(\varphi_i,R_k(\varphi_i))\in E_j$ 则
        如果 $c_j=0$ 则
          $k_i=k_j$;
        结束如果
      结束如果
    结束循环
  结束如果
结束循环
返回 k i

我们设计了算法5，为智能体 i寻找智能体 α 和智能体 β。在算法5中，我们设计了一个类似于式（1）的操作，如下所示

$$\delta^*(\theta_1,\theta_2)=\begin{cases}\theta_1-\theta_2-2\pi & if \quad \theta_1-\theta_2>\pi\\ \theta_1-\theta_2+2\pi & if \quad \theta_1-\theta_2\le-\pi\\ \theta_1-\theta_2 & else\end{cases} \tag{20}$$

实际上，$\delta(\theta_1,\theta_2)=|\delta^*(\theta_1,\theta_2)|$。该操作可以计算从 θ_1 到θ_2 沿逆时针方向的相位差。我们可以发现，$\delta^*(\theta_1,\theta_2)\in(-\pi,\pi]$，这在算法中难以比较。因此，我们使用 $\Psi(cos(\delta^*(\theta_1,\theta_2))\ sin(\delta^*(\theta_1,\theta_2)))$使所有相位差均为正值。然后，通过求这些相位差的最小值，可以确定沿逆时针方向的智能体 α 。同样，我们也可以得到另一个方向上的智能体 β。

当顺时针方向上的邻居智能体 α 发生变化时，分界点 s_i 也应相应改变。对于逆时针方向，情况也相同，但分界点 s_i不会改变。当智能体 α 不发生变化时，智能体 i 可以执行算法1。这确保了算法1能够高效运行。

此外，　　各层的重要性应有所不同。一般来说，

算法 4 入口请求算法

如果 $N_k = 0$ 则
    $a_i = 1$;
    $\alpha = \beta = i$;
    $s_i = \varphi_i + \pi$;
    如果 $s_i > 2\pi$ 则
        $s_i = s_i - 2\pi$;
    结束如果
否则
    对于 $j \in I_{N_k}$ **do**
        如果 $(\varphi_i, R_k(\varphi_i)) \in E_j$ 则
            如果 $c_j = 1$ 则
                $k_i = 0$;
            else
                $a_i = 1$;
                当 $\varphi_i = \varphi_j$ 时
                    按照 (16) 移动;
                    根据 (2) 和 (3) 计算 φ_i ;
                结束循环
            结束如果
        结束如果
    结束循环
结束如果

内层越重要。因此，当智能体数量不足时，应优先覆盖内层。如图3所示，当外层智能体发现内层智能体需要帮助时，即使自身工作正常，也会离开外层并前往内层。我们使用算法 7 来实现此功能。当智能体 i 发现内层智能体的 $\varphi_i \in E_j$、 $a_j = 1$ 和$c_j = 0$ 时，智能体 i 会将自身转变为空闲智能体，而内层则是目标层。

最后，我们在算法8中给出多层屏障覆盖算法。当智能体i 没有目标层时，会先寻找目标层。如果智能体 i 在相位不变的情况下无法找到目标层，就会改变其相位。找到目标层后，智能体将

算法5 邻居搜索算法

输入： I_{N_k}, 智能体 i, j状态

输出： α, β

对于 $j \in I_{N_k}$&& $j \neq i$，执行

  如果 $N_k = 2$，则

    $\alpha = \beta = j$;

  else

    如果 $\Psi(cos(\delta^*(\varphi_i, \varphi_j))\ sin(\delta^*(\varphi_i, \varphi_j))) < \Psi(cos(\delta^*(\varphi_i, \varphi_\alpha))\ sin(\delta^*(\varphi_i, \varphi_\alpha)))$ 则

      $\alpha = j$;

    结束如果

    如果 $\Psi(cos(\delta^*(\varphi_j, \varphi_i))\ sin(\delta^*(\varphi_j, \varphi_i))) < \Psi(cos(\delta^*(\varphi_\beta, \varphi_i))\ sin(\delta^*(\varphi_\beta, \varphi_i)))$ 则

      $\beta = j$;

    结束如果

  结束条件判断

结束循环

返回 α 和 β;

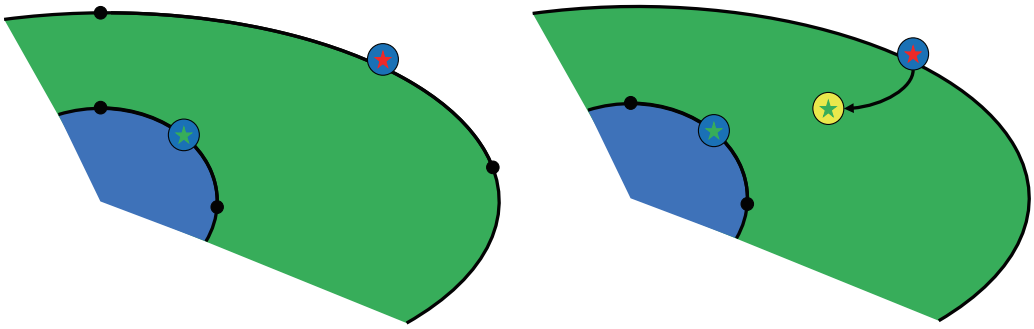

图3：智能体移动到其他层的示意图

移动到目标层。到达目标层后，智能体会请求进入该目标层。如果请求被拒绝，智能体会寻找另一个目标层。如果请求获准，智能体便进入该层执行覆盖任务。为确保算法顺利进行，智能体始终获取邻居信息。最后，当智能体发现内层需要帮助时，会停止覆盖，转而帮助内层。我们用 P 表示检测

算法 6 分界点集算法

初始化： $z = \alpha$

运行算法5；

如果 $z \neq \alpha$ ，则

    分界点设为$s_i = \varphi_i - \frac{1}{2}\delta(\varphi_i, \varphi_\alpha)$；

    如果 $s_i < 0$ ，则

        $s_i = s_i + 2\pi$；

    结束如果

否则

    运行算法1；

结束如果

算法 7 基于权重的层变更算法

如果 $k_i > 1$ 则

    运行算法 2；

    对于 $j \in I_{N_{k_i-1}}$ **do**

        如果 $\varphi_i \in E_j$ && $a_j = 1$ && $c_j = 0$ 则

            设置 $k_i = k_j$ 和 $a_i = 0$；

        结束如果

    结束循环

结束条件判断

算法的概率。 P 计算如下

$$P = 1 - \prod_{k=1}^{N_k}(1 - P_k), \tag{21}$$

其中， P_k 是第 k层的检测概率。

在下一节中，我们将从理论上证明所提出算法的有效性。

3 主要结果

引理 3.1. 对于固定的智能体位置， $\mathcal{T}$ 的中点集合保证联合监测概率 H达到最大值。

算法8 多层屏障覆盖算法

初始化： $k=1$, $a_i=0$, $c_i=0$

对于 $i \in I_N$，第 i 个智能体执行如下操作

当 $a_i=0$ 时
  当 $k_i=0$ 时
    运行算法 2;
    运行算法3;
    按照（16）移动;
  结束循环
  while $p_i \notin r_{k_i}$ do
    按照（17）移动到目标层;
  结束循环
  运行算法 2;
  运行算法 4;
  当 $a_i=1$ 时
    运行算法 2;
    运行算法 6;
    运行算法 7;
    用（18）和（19）更新 c_i ;
  结束循环
结束循环

证明. 对 H 关于 s_i求偏导，可得

$$\frac{\partial H}{\partial s_i}=\left[f\left(d(\varphi_{i-1},s_i)\right)-f\left(d(\varphi_i,s_i)\right)\right]\rho\left(s_i\right).$$

可以观察到，对于 $i=1,2,...,N$，若 $f(d(\varphi_{i-1},s_i))=f(d(\varphi_i,s_i))$，则可得 $\frac{\partial H}{\partial S}=0$。根据第二节中对 $f(\cdot)$ 性质的描述，这意味着只需满足 $d(\varphi_{i-1},s_i)=d(\varphi_i,s_i)$，对于 $i=1,2,...,N$，即可实现 $\frac{\partial H}{\partial S}=0$ 。由于智能体的位置各不相同，$d(\varphi_{i-1},s_i)=d(\varphi_i,s_i)$ 意味着分界点是 $\mathcal{T}_i$ 的中点。此外，覆盖质量函数 (7) 的海森矩阵满足

$$\begin{aligned}\nabla^2 H&=[\frac{\partial^2 U}{\partial s_i \partial s_j}]\in R^{N\times N}\\&=\mathrm{diag}(\alpha_1,\alpha_2,...,\alpha_N),\end{aligned}$$

其中 $\alpha_i = (\frac{\partial f(d(\varphi_{i-1}, s_i))}{\partial s_i} - \frac{\partial f(d(\varphi_i, s_i))}{\partial s_i})\rho(s_i)$。由于 $f(\cdot)$ 单调递减，且 $d(\varphi_i, s_i) \propto \delta(\varphi_i, s_i)$，结合方程 (1)，可得 $\frac{\partial f(d(\varphi_{i-1}, s_i))}{\partial s_i} < 0$ 且 $\frac{\partial f(d(\varphi_i, s_i))}{\partial s_i} \rho(s_i) > 0$。这意味着对于 $i = 1, 2, ..., N$，有 $\alpha_i < 0$。因此，可得 $\nabla^2 H < 0$，这就证明了该引理。

□

定理 3.1. 动态系统 (9) 和 (14) 确保函数 (7) 达到局部最大值。

证明。构造如下李雅普诺夫函数

$$V(\varphi, \mathcal{S}) = \frac{1}{H}.$$

由于 $s_i \in [0, 2\pi)$，$f(d(\varphi_i, \theta)) > 0$ 和 $\rho(\theta) \geq 0$，我们可以得出 $V > 0$。此外，$f(d(\varphi_i, \theta))$ 和 $\rho(\theta)$ 有界，这意味着 V_1 有界。对李雅普诺夫函数求导，可得

$$\dot{V}(\varphi, \mathcal{S}) = -\frac{1}{H^2} \cdot \dot{H},$$

由 (7) 可知，$V(\varphi, \mathcal{S}) > 0$。关于复合动力学 (9) 和 (14)，(7) 的时间导数为

$$\begin{aligned}
\dot{H} &= \sum_{i=1}^{N} \frac{\partial H}{\partial \varphi_i} \omega_i + \sum_{i=1}^{N} \frac{\partial H}{\partial s_i} \dot{s}_i \\
&= \sum_{i=1}^{N} \left(\int_{E_i} \frac{\partial f\left(d(\varphi_i, \theta)\right)}{\partial \varphi_i} \rho\left(\theta\right) d\theta \right)^2 \\
&+ \kappa_s \sum_{i=1}^{N} \left[f\left(d(\varphi_{i-1}, s_i)\right) - f\left(d(\varphi_i, s_i)\right) \right] \left(d(\varphi_i, s_i) - d(\varphi_{i-1}, s_i) \right) \rho\left(s_i\right)
\end{aligned}$$

由于 $[f(d(\varphi_{i-1}, s_i)) - f(d(\varphi_i, s_i))](d(\varphi_i, s_i) - d(\varphi_{i-1}, s_i)) \geq 0$，我们可以得出 $\frac{dH}{dt} \geq 0$。另一方面，李雅普诺夫函数的导数满足 $V_1 \leq 0$。根据局部不变集定理，系统状态将收敛到集合 $\{(\varphi, s) | V_1 = 0\}$。由方程 (7) 可知，$H$ 有界。因此，当且仅当 $\frac{dH}{dt} = 0$, $\dot{V}_1 = 0$。在这种情况下，分界点位于智能体之间弧长的中点处。同时，智能体位于集合 $\{\varphi_i | \int_{E_i} \frac{\partial f_{(d(\varphi_i,\ \theta))}}{\partial \varphi} \rho(\theta) d\theta = 0\}$ 中。因此，V 达到局部最小值。另一方面，H 达到局部最大值。

□

引理 3.2. 对于工作智能体 i，智能体 i 永远不会与分界点发生碰撞。

证明。假设智能体 i 在时间 t^* 进入该层，我们可以得到以下关系

$$\delta^*(s_\beta, \varphi_i) > 0, \delta^*(\varphi_i, s_\alpha) > 0$$

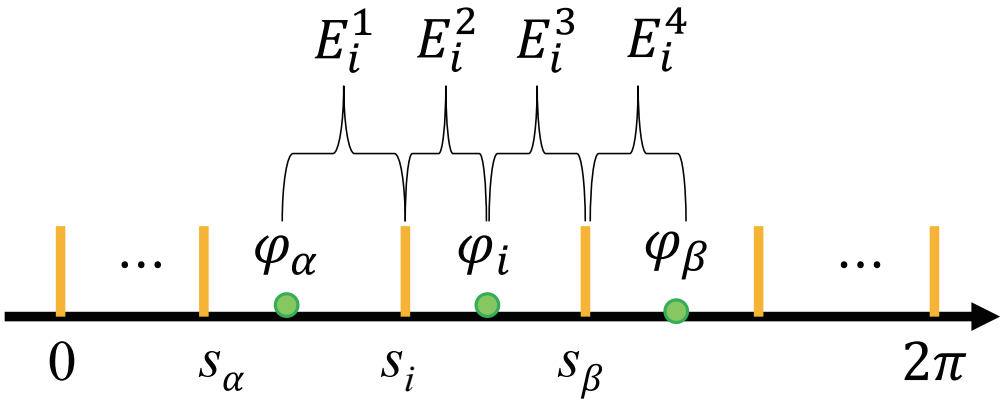

图 4：智能体 i 的相位位置示意图

首先考虑智能体 i 不会与分界点 s_β 发生碰撞的情况。如图 4 所示，我们用 E_i^1、E_i^2、E_i^3、E_i^4 表示两个相位之间的区域。我们用 L_β 表示智能体 i 与分界点 s_β 之间的距离，L_β 表示如下

$$L_\beta = \mathcal{K}(\delta^*(s_\beta, \varphi_i)),$$

其中，$\mathcal{K}(\cdot)$ 是一个 $\mathcal{K}$ 类函数。接下来，结合公式（20），对 L_β 求导可得

$$\begin{aligned}\dot{L}_\beta &= \mathcal{K}_1 \cdot (\dot{s_\beta} - \dot{\varphi_i}) \\ &= \mathcal{K}_1 \cdot (\kappa_s(\delta^*(\varphi_\beta, s_\beta) - \delta^*(s_\beta, \varphi_i)) - \int_{E_i} \frac{\partial f(d(\varphi_i, \theta))}{\partial \varphi_i} \rho(\theta) d\theta) \\ &= \mathcal{K}_1 \cdot (\kappa_s(\delta^*(\varphi_\beta, s_\beta) - \delta^*(s_\beta, \varphi_i)) - \int_{E_i^3} \frac{\partial f(d(\varphi_i, \theta))}{\partial \varphi_i} \rho(\theta) d\theta - \int_{E_i^2} \frac{\partial f(d(\varphi_i, \theta))}{\partial \varphi_i} \rho(\theta) d\theta)\end{aligned}$$

其中，$\mathcal{K}_1 = \frac{\partial \mathcal{K}(\delta^*(s_\beta, \varphi_i))}{\partial \delta^*(s_\beta, \varphi_i)} > 0$。由于 $\varphi_i \longrightarrow s_\beta$，我们可以发现 $\int_{E_i^3} \frac{\partial f(d(\varphi_i, \theta))}{\partial \varphi_i} \rho(\theta) d\theta \longrightarrow 0$，$\frac{\partial f(d(\varphi_i, \theta))}{\partial \varphi_i} < 0$ 位于 E_i^2 范围内，且 $\delta^*(\varphi_\beta, s_\beta) > 0$。因此，我们可以得到

$$\dot{L}_\beta > -\mathcal{K}_1 \cdot \kappa_s(\delta^*(s_\beta, \varphi_i)),$$

这意味着 $L_\beta > 0$ 在 $t \geq t^*$的区间内成立。同理，我们用 L_α表示智能体 i 与分界点 s_i之间的距离，即 $L_\alpha = \mathcal{K}(\delta^*(\varphi_i, s_i))$，并且可以

得到以下结果

$$\begin{aligned}\dot{L}_i &= \mathcal{K}_2 \cdot (\dot{\varphi}_i - \dot{s}_i) \\ &= \mathcal{K}_2 \cdot (\int_{E_i} \frac{\partial f(d(\varphi_i,\theta))}{\partial \varphi_i}\rho(\theta)d\theta - \kappa_s(\delta^*(\varphi_i,s_i) - \delta^*(s_i,\varphi_\alpha))) \\ &= \mathcal{K}_2 \cdot (\int_{E_i} \frac{\partial f(d(\varphi_i,\theta))}{\partial \varphi_i}\rho(\theta)d\theta - \kappa_s(\delta^*(\varphi_i,s_i) - \delta^*(s_i,\varphi_\alpha))) \\ &= \mathcal{K}_2 \cdot (\int_{E_i^2} \frac{\partial f(d(\varphi_i,\theta))}{\partial \varphi_i}\rho(\theta)d\theta + \int_{E_i^3} \frac{\partial f(d(\varphi_i,\theta))}{\partial \varphi_i}\rho(\theta)d\theta + \kappa_s\delta^*(s_i,\varphi_\alpha) - \kappa_s\delta^*(\varphi_i,s_i))\end{aligned}$$

其中，$\mathcal{K}_2 = \frac{\partial \mathcal{K}(\delta^*(s_i,\varphi_i))}{\partial \delta^*(s_i,\varphi_i)} > 0$。由于 $\varphi_i \longrightarrow s_i$，我们可以发现 $\int_{E_i^2} \frac{\partial f(d(\varphi_i,\theta))}{\partial \varphi_i} \rho(\theta) d\theta \longrightarrow 0$，$\frac{\partial f(d(\varphi_i,\theta))}{\partial \varphi_i} > 0$ 位于 E_i^3的范围内，且 $\delta^*(s_i,\varphi_\alpha) > 0$。因此，我们可以得到

$$\dot{L}_\alpha > -\mathcal{K}_2 \cdot \kappa_s(\delta^*(\varphi_i,s_i)),$$

这意味着 $L_\alpha > 0$ 在区间 $t \geq t^*$内成立。由于 $L_\alpha > 0$ 和 $L_\beta > 0$，智能体不会与分界点发生碰撞。 □

引理 3.3. 分界点之间永远不会发生碰撞。

证明. 根据引理 3.2，我们可以知道，分界点 s_i 和s_β 之间的距离可表示为 $L_i = L_\alpha + L_\beta$。显然，L_i 是 E_i的长度。因此，我们可以得到 $L_i > 0$在区间 $t \geq t^*$内成立，这意味着该引理得证。

□

我们用 L^k 表示层 k 的长度。为证明我们的结论，我们给出以下引理。

引理 3.4. 对于在层 k 工作的智能体 i，若 $L_i = min\{j \in I_{N_k} | L_j\}$，则有 $L_i \leq \frac{L^k}{N_k}$。

证明. 根据表 6，第 k 层的区域将由在第 k 层工作的智能体无剩余地划分。因此，我们可以得到以下关系

$$L^k = \sum_{j \in I_{N_k}} L_j.$$

根据引理 3.3，当 $i \in I_{N_k}$ 时，有 $L_i > 0$。由于 $L_i = min\{j \in I_{N_k} | L_j\}$，当 $j \in I_{N_K}$ 时，有 $L_j \geq L_i$。因此，上式可以改写为

$$L^k = \sum_{j \in I_{N_k}} L_j \geq N_k \times L_i$$

这意味着 $L_i \leq \frac{L_k}{N_k}$。 □

引理 3.5. 对于在层 k工作的智能体 i，当 $t \longrightarrow \infty$ 时，若 $L_i = max\{j \in I_{N_k} | L_j\}$ 且 $N_k \geq 2$，则有 $\frac{L^k}{N_k} \leq L_i \leq \frac{L^k}{2}$。

证明。与引理 3.4 类似，由于 $L_i = max\{j \in I_{N_k} | L_j\}$，我们有 $L_j \leq L_i$，对于 $j \in I_{N_K}$，这意味着 $L_i \geq \frac{L_k}{N_k}$。

如图 4 所示，我们用 l_i 表示 $E_i^3 \cup E_i^4$ 的长度。可以得到以下关系

$$L^k = \sum_{j \in I_{N_k}} l_j.$$

根据引理 3.1 和控制输入 (14)，由于 $t \longrightarrow \infty$，智能体 i 有如下关系

$$L_i = 0.5(l_i + l_\alpha),$$

由于 $l_i + l_\alpha \leq L^k$，我们得到 $L_i \leq \frac{L^k}{2}$。 □

在我们的算法中，第 k 层上的智能体数量并非固定。显然，我们可以得到如下关系 $P_k(t) \geq 0$，对于 $t \geq 0$。

引理 3.6。对于层 k，如果智能体 i 离开该层且 $N_k \geq 2$，则第 k 层的检测概率最大降幅可按如下方式计算

$$\begin{aligned} P_k{}' = & \int_{E_i^2} \left(f\left(d\left(\varphi_i, \theta\right)\right) - f\left(d\left(s_i, \varphi_i\right) + d\left(s_i, \theta\right)\right)\right) \rho\left(\theta\right) d\theta \\ & + \int_{E_i^3} \left(f\left(d\left(\varphi_i, \theta\right)\right) - f\left(d\left(s_\beta, \varphi_i\right) + d\left(s_\beta, \theta\right)\right)\right) \rho\left(\theta\right) d\theta \end{aligned}$$

证明。在我们的算法中，当智能体 i 进入或离开时，层 k中所有区域的 E_i^2 和 E_i^3 的监测概率只会发生变化。假设智能体 i 在时间 ti 离开第 k 层，新的分界点位于 φ_i的位置。根据引理 3.1，我们可以得到以下公式

$$\begin{aligned} P_k(t_i + \varepsilon) \geq P_k(t_i) - & \int_{E_i^2} \left(f\left(d\left(\varphi_i, \theta\right)\right) - f\left(d\left(\varphi_\alpha, \theta\right)\right)\right) \rho\left(\theta\right) d\theta \\ & + \int_{E_i^3} \left(f\left(d\left(\varphi_i, \theta\right)\right) - f\left(d\left(\varphi_\beta, \theta\right)\right)\right) \rho\left(\theta\right) d\theta, \end{aligned}$$

其中 ε 是一个无穷小量。由表 1 可知， s_i 和s_β 将收敛到 l_α 和 l_i的中点。如图 4 所示， E_i^1 的长度与 E_i^2相同，而

E_i^3 的长度与 E_i^4相同。因此，第k层的检测概率变化可改写如下

$$P_k{}' \geq \int_{E_i^2} \left(f\left(d\left(\varphi_i,\theta\right)\right) - f\left(d\left(s_i,\varphi_i\right) + d\left(s_i,\theta\right)\right)\right)\rho\left(\theta\right)d\theta + \int_{E_i^3} \left(f\left(d\left(\varphi_i,\theta\right)\right) - f\left(d\left(s_\beta,\varphi_i\right) + d\left(s_\beta,\theta\right)\right)\right)\rho\left(\theta\right)d\theta,$$

这就证明了该引理。 □

引理 3.7. 对于层 k，如果智能体 i进入该层，则第 k 层检测概率的最小增量可计算如下

$$P_k' = \int_{E_i^2 \cup E_i^3} f\left(d\left(\varphi_i,\theta\right)\right)\rho\left(\theta\right)d\theta - \int_{s_\beta^*}^{s_\beta} f\left(d\left(\varphi_\beta,\theta\right)\right)\rho\left(\theta\right)d\theta - \int_{s_i}^{s_\beta^*} f\left(d\left(\varphi_\alpha,\theta\right)\right)\rho\left(\theta\right)d\theta$$

其中， s_β^* 是智能体 i 进入层 k之前， l_α 中的分界点；如果 $N_k = 0$，那么 $d(\varphi_\alpha, s_i) = 0$，$d(\varphi_\alpha, \theta) = 0$。

证明。正如引理 3.6 所述，智能体进入只会改变第 k 层中 E_i^2 和 E_i^3的检测概率。假设智能体在时间 t_i之后进入层 k，则该区域的检测概率如下

$$P_k(t_i) = P_k^*(t_i) + \int_{s_i}^{s_\beta^*} f\left(d\left(\varphi_\alpha,\theta\right)\right)\rho\left(\theta\right)d\theta + \int_{s_\beta^*}^{s_\beta} f\left(d\left(\varphi_\beta,\theta\right)\right)\rho\left(\theta\right)d\theta$$

其中， P_k^* 表示第 k 层不考虑 E_i^2 和E_i^3的监测概率。智能体 i 进入第 k 层后，根据定理 3.1，上述公式可改写为

$$P_k(t_i+\varepsilon) \geq P_k^*(t_i+\varepsilon) + \int_{E_i^2 \cup E_i^3} f\left(d\left(\varphi_i,\theta\right)\right)\rho\left(\theta\right)d\theta$$

其中， $P_k^*(t_i+\varepsilon) = P_k^*(t_i)$。因此，可得到 P_k'，如下所示

$$P_k' \geq \int_{E_i^2 \cup E_i^3} f\left(d\left(\varphi_i,\theta\right)\right)\rho\left(\theta\right)d\theta - \int_{s_\beta^*}^{s_\beta} f\left(d\left(\varphi_\beta,\theta\right)\right)\rho\left(\theta\right)d\theta - \int_{s_i}^{s_\beta^*} f\left(d\left(\varphi_\alpha,\theta\right)\right)\rho\left(\theta\right)d\theta$$

这意味着该引理成立。 □

根据上述结论，我们可以得到以下定理

定理 3.2. 对于具有 N_k 层的多智能体多层屏障覆盖系统，若在第 k 层工作的智能体 i 满足以下不等式，

$$P_k' < \frac{(1-P_k)P_v'}{1-P_v-P_v'}$$

检测 如果智能体 i 进入第 v 层，系统的概率 (21) 将会增加 r.

证明。不失一般性，我们可以假设，在时间 t_1，多智能体覆盖系统中的智能体 i 在第 k 层工作；在时间 t_2，智能体 i 在第 v 层工作。此外，在这两个时刻，除智能体 i 的工作位置不同外，其他智能体仍在同一层工作。因此，根据 (21)，我们可以得到以下等式

$$P(t_1) = 1 - (1 - P_1)(1 - P_2)...(1 - P_k)...(1 - P_v)...(1 - P_{N_k}).$$

由引理 3.6 和引理 3.7，可得以下等式

$$P(t_2) \geq 1 - (1 - P_1)(1 - P_2)...(1 - P_k + P_k\prime)...(1 - P_v - P_v\prime)...(1 - P_{N_k})$$

令 $1-(1-P_1)(1-P_2)...(1-P_k+P_k\prime)...(1-P_v-P_v\prime)...(1-P_{N_k})) > P(t_1)$，可得$P(t_1) > P(t_2)$，这意味着该定理成立。化简上述公式，得到

$$P_k' < \frac{(1 - P_k)P_v'}{1 - P_v - P_v'}$$

证明完毕。 □

推论 3.1. 对于具有固定分割点的单层屏障覆盖系统，当层为半径为 R_0 的圆时， d采用该层上的测地距离，且智能体的检测模型为高斯概率模型，即 $f(d) = e^{-d^2/\gamma^2}$，若半径R_0 满足 $R_0 \leq \frac{\sqrt{2}\gamma}{2\pi}$，则动态系统 (9) 可确保函数 (7) 达到最大值。

证明。对（7）关于 φ_i 求偏导数，得到

$$\frac{\partial H}{\partial \varphi_i} = \int_{E_i} \frac{\partial f\left(d(\varphi_i, \theta)\right)}{\partial \varphi_i} \rho\left(\theta\right) d\theta$$

将 $f(d) = e^{-d^2/\gamma^2}$ 代入上述公式，得到

$$\frac{\partial H}{\partial \varphi_i} = \int_{E_i^1} e^{-\frac{d^2}{\gamma^2}} \left(-2\frac{d}{\gamma^2}\right) \frac{\partial d}{\partial \varphi_i} \rho\left(\theta\right) + \int_{E_i^2} e^{-\frac{d^2}{\gamma^2}} \left(-2\frac{d}{\gamma^2}\right) \frac{\partial d}{\partial \varphi_i} \rho\left(\theta\right)$$

由于测地距离 d 在 $\theta = \varphi_i$ 时不可导，因此积分被分段。并且，当 $\theta \in E_i^2$ 时，$\frac{\partial d}{\partial \varphi_i} = R_o$ $\frac{\partial d}{\partial \varphi_i} = -R_o$ 当 $\theta \in E_i^3$。

W我们对上述公式关于 φ_i 求偏导，得到

$$\frac{\partial^2 H}{\partial {\varphi_i}^2} = \int_{E_i^2} e^{-\frac{d^2}{\gamma^2}} \left(4\frac{d^2}{\gamma^4} - \frac{2}{\gamma^2}\right) \left(\frac{\partial d}{\partial \varphi_i}\right)^2 \rho\left(\theta\right) d\theta + \int_{E_i^3} e^{-\frac{d^2}{\gamma^2}} \left(4\frac{d^2}{\gamma^4} - \frac{2}{\gamma^2}\right) \left(\frac{\partial d}{\partial \varphi_i}\right)^2 \rho\left(\theta\right) d\theta$$

由引理 3.5 可得 $d \leq \pi R_0$。若 $R_0 \leq \frac{\sqrt{2}\gamma}{2\pi}$，则有 $\frac{\partial^2 H}{\partial \varphi_i{}^2} < 0$。此外，覆盖质量函数 (7) 的海森矩阵满足

$$\nabla^2 H = [\frac{\partial^2 U}{\partial \varphi_i \partial \varphi_j}] = \mathrm{diag}(\frac{\partial^2 H}{\partial \varphi_1^2}, \frac{\partial^2 H}{\partial \varphi_2^2}, ..., \frac{\partial^2 H}{\partial \varphi_N^2}) \in R^{N \times N}.$$

这意味着 H 具有唯一的最大值。结合定理 3.1，动态系统 (9) 将确保函数 (7) 达到最大值。

□

4 案例研究

本节将给出一些仿真和实验结果，以验证我们的覆盖算法。我们使用 MATLAB 2022a 实现了该算法。现在，我们在表 8 中给出多智能体屏障覆盖算法。

4.1 数值仿真

我们设计了 3 个区域层。这三层共有 50 个智能体需要执行覆盖任务，以监测入侵者的入侵。这三层的设计如下

$$\begin{cases} R_1(\theta) = 1 + 0.15\sin(4\theta), \\ R_2(\theta) = 2 + 0.15\sin(10\theta), \\ R_3(\theta) = 3 + 0.15\sin(40\theta). \end{cases} \tag{22}$$

概率模型表示为 $f(d(\varphi_i, \theta)) = \exp(-d(\varphi_i, \theta)^2)$，其中距离函数$d$ 的计算如下

$$d(\varphi_i, \theta) = \begin{cases} \left| \int_{\varphi_i}^{\theta} \sqrt{R(\theta)^2 + R'(\theta)^2} d\theta \right|, & if \quad \left| \int_{\varphi_i}^{\theta} \sqrt{R(\theta)^2 + R'(\theta)^2} d\theta \right| \leq \frac{L_{k_i}}{2} \\ L_{k_i} - \left| \int_{\varphi_i}^{\theta} \sqrt{R(\theta)^2 + R'(\theta)^2} d\theta \right|. & otherwise \end{cases}$$

其中， d 是 Lipschitz连续的。密度函数为 $\rho(\theta) = \frac{\theta}{2\pi^2}$。我们将可调参数设置如下

$$\begin{cases} \kappa_r = 0.1, \\ \kappa_\omega = 0.01, \\ \kappa_s = 0.05. \end{cases}$$

如图5所示，我们将智能体放置在最内层内。所有智能体逐渐向外扩展，最终覆盖全部三层。我们截取了算法在4个时间点的位置结果，分别为0秒、8秒、16秒和24秒。

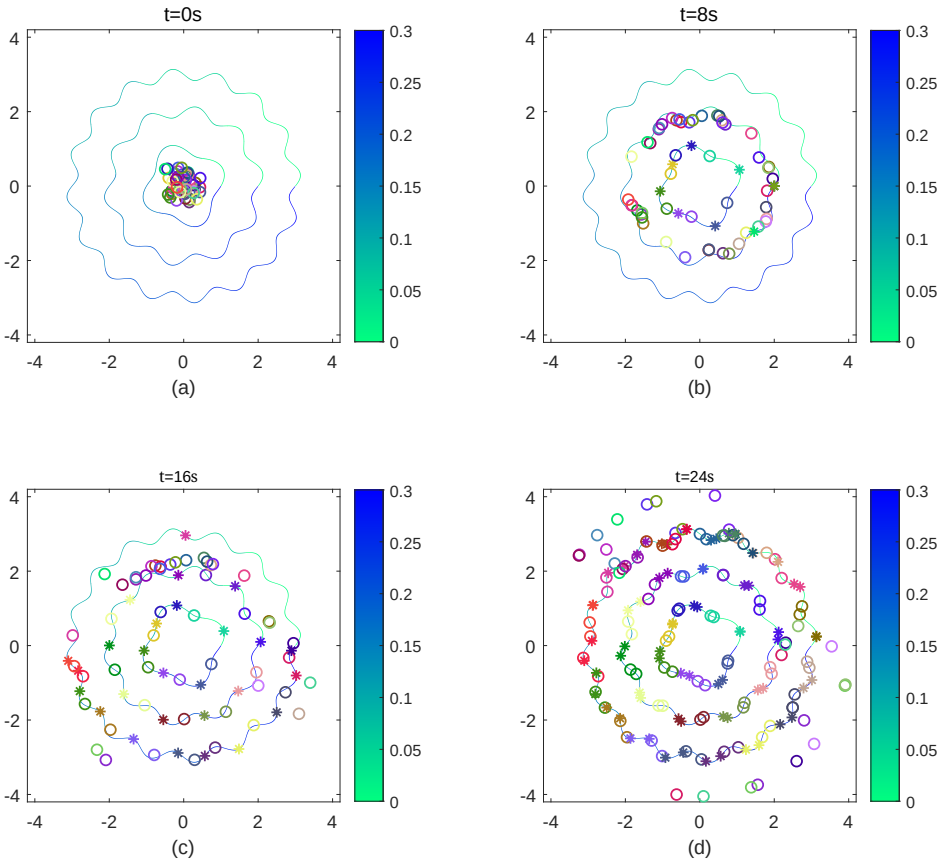

图5：仿真结果快照。圆表示移动智能体，星号表示分界点。

如图5所示，算法刚开始运行时，所有智能体都位于最内层的内区域。算法运行8秒后，第一层上已有6个智能体，且一些智能体已移动到第二层。结合图7可知，算法运行约13秒后，第三层的检测概率开始下降。算法运行至16秒时，我们发现一些智能体正从第三层移向第二层。这是因为在算法7中，当内层智能体无法很好地胜任其检测任务时，外层智能体会离开外层，前往内层帮助内层智能体。算法运行24秒后，多智能体系统基本趋于稳定，大多数智能体已在相应层上执行任务。在半径为4的圆周附近，一些智能体正在巡逻，寻找需要帮助的智能体；一旦发现，这些

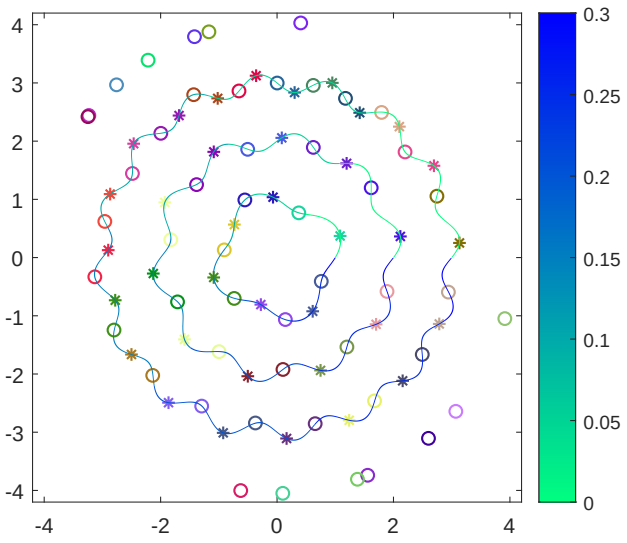

图 6：系统状态的最终结果。

巡逻智能体将采取行动。最后，我们在图 6 中给出了算法运行至系统最后时刻的结果。可以发现，在每一层中，入侵概率较高的区域智能体分布得更密集。此外，仍有空闲智能体在半径为 4 的圆周上巡逻。

图 7 展示了系统及各层的检测概率随时间的变化情况。可以看到，当第二层和第三层没有智能体时，系统的总检测概率与第一层的检测概率相同。当第二层和第三层的智能体先后开始工作时，智能体的监测概率显著提高。最后可以发现，多层围栏覆盖算法的检测概率超过 99.99%。

我们还针对多层屏障覆盖和单层屏障覆盖进行了对照实验。如图 8 所示，在初始阶段，多层屏障覆盖的检测概率低于单层围栏覆盖；但当智能体移动到第二层后，多层屏障覆盖的检测概率反超单层围栏覆盖，并在之后的整个过程中始终高于单层屏障覆盖。

我们统计了不同数量智能体下单层屏障覆盖和多层屏障覆盖的最终检测概率，如图 9 所示。可以发现

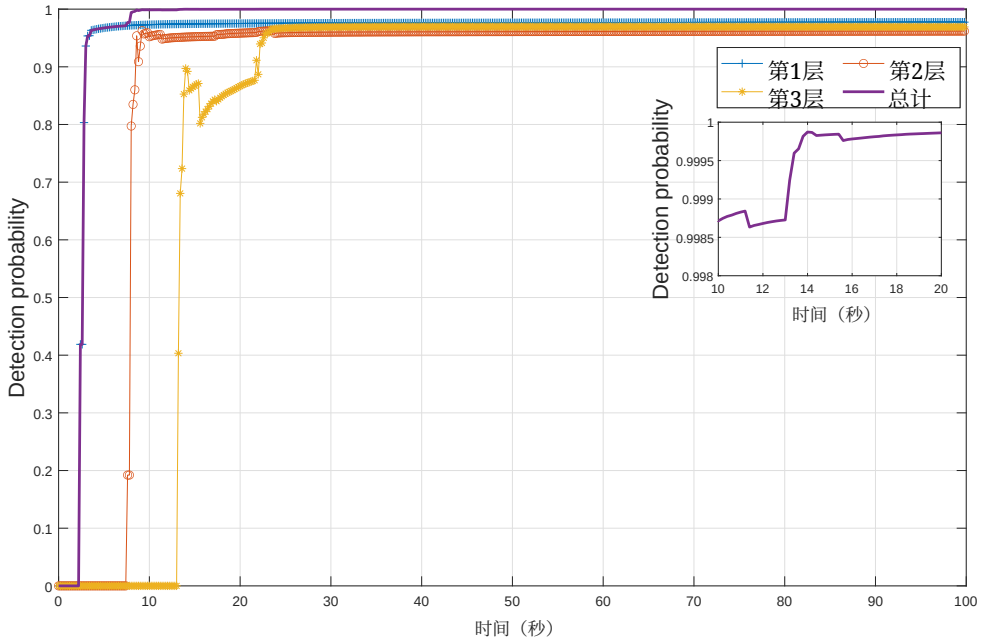

图7：各层及整个系统的检测概率。

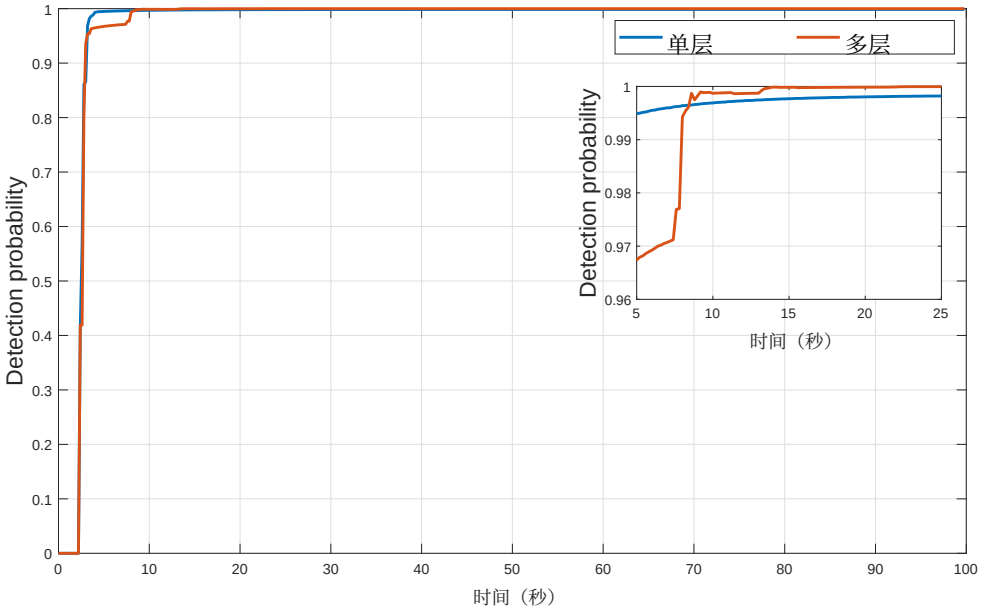

图8：相同智能体数量下单层屏障覆盖与多层屏障覆盖的差异。

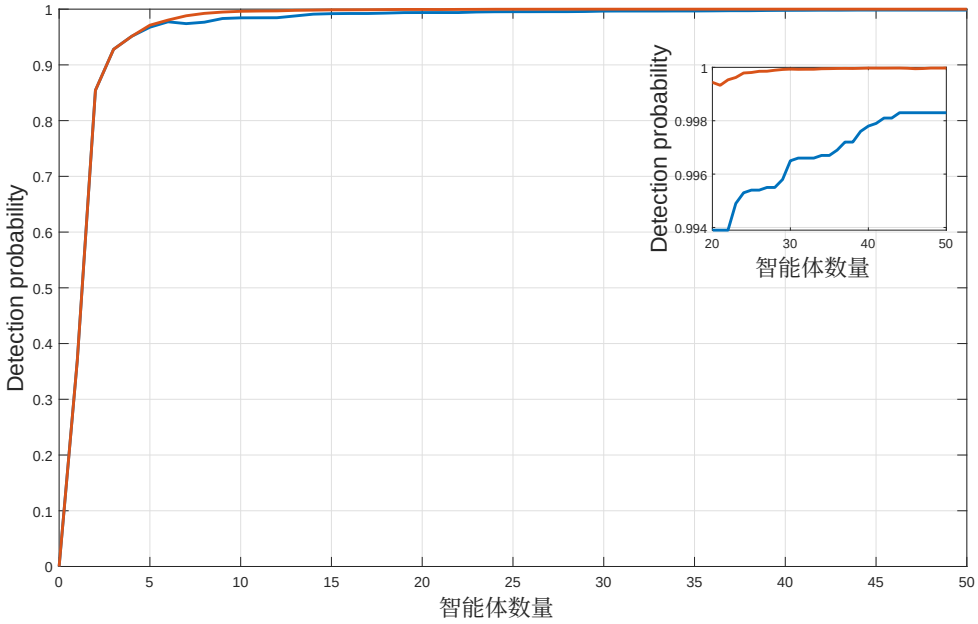

图9：相同智能体数量下单层屏障覆盖与多层屏障覆盖之间检测概率的差异。

当智能体数量较少时，单层屏障覆盖与多层屏障覆盖之间的检测概率没有差异。然而，随着智能体数量逐渐增加，多层屏障覆盖的检测概率显著高于单层屏障覆盖。当智能体数量足够多时，增加智能体数量对单层屏障覆盖的帮助很小。当智能体数量为50时，单层屏障覆盖的检测概率达到99.8%，而多层屏障覆盖的检测概率非常接近100%。

5 结论

本文提出了一种分布式多智能体屏障覆盖算法。首先，基于入侵概率模型设计了单层屏障覆盖质量函数，并基于梯度法设计了单层屏障覆盖算法。随后，在单层算法的基础上提出了层间调整机制，通过调整各层的智能体数量，提高整个系统的覆盖质量。然后进行了理论分析，从理论上验证了单层算法的稳定性和

有效性以及多层算法的必要性，并给出了若干特殊情况下的理论结果。最后，通过仿真验证了所提算法的有效性，并通过实验验证了该算法的实用性。

6 附录

致谢

本项目得到中国地质大学（武汉）中央高校基本科研业务费的资助。

参考文献

[1] Vicsek, Tam´as 等.《自驱动粒子系统中的一种新型相变》. Physical Review Letters 75.6 (1995): 1226.[2] Wilson S, Glotfelter P, Wang L 等.《The Robotarium：远程访问、多机器人系统分布式控制中的全球影响力机遇、挑战与经验教训》[J]. IEEE Control Systems Magazine, 2020, 40(1): 26-44.[3] Cortes J, Martinez S, Karatas T 等.《移动传感网络的覆盖控制》[J]. IEEE Transactions on Robotics and Automation, 2004, 20(2): 243-255.[4] Thanou M, Stergiopoulos Y, Tzes A.《利用测地度量实现非凸环境下的分布式覆盖》[C]. 2013 IEEE国际机器人与自动化会议. IEEE, 2013: 933-938.[5] Zhai C, Zhang H T, Xiao G.《多智能体系统的协同覆盖控制及其应用》[M]. Springer, 2021.[6] Santos M, Mayya S, Notomista G 等.《时变密度函数下的去中心化最小能量覆盖控制》[C]. 2019年国际多机器人与多智能体系统研讨会（MRS）. IEEE, 2019: 155-161.[7] Benevento A, Santos M, Notarstefano G 等.《未知空间场的估计与覆盖的多机器人协调》[C]. 2020年IEEE国际机器人与自动化会议（ICRA）. IEEE, 2020: 7740-7746。

[8] Zhai C, Hong Y.《具有工作负载不确定性的多智能体系统去中心化扫掠覆盖算法》[J]. Automatica, 2013, 49(7): 2154-2159.[9] Ivi´c, Crnkovi´c, Mezi´c I.《基于遍历性的势场多智能体协同区域覆盖》[J]. IEEE Transactions on Cybernetics, 2016, 47(8): 1983-1993.[10] Zheng Y, Zhai C.《利用热传导方程实现不确定环境下多智能体系统的分布式覆盖控制》[J]. arXiv 预印本 arXiv:2204.09289, 2022.[11] Kumar S, Lai T H, Arora A.《利用无线传感器实现屏障覆盖》[C]//第11届年度国际移动计算与网络会议论文集. 2005: 284-298.[12]Liu B, Dousse O, Wang J 等.《无线传感器网络的强屏障覆盖》[C]//第9届ACM移动自组网与计算国际研讨会论文集. 2008: 411-420.[13] Wang Z, Liao J, Cao Q 等.《在混合定向传感器网络中实现k-屏障覆盖》[J]. IEEE Transactions on Mobile Computing, 2013, 13(7): 1443-1455.[14] Chen A, Kumar S, Lai T H.《设计用于屏障覆盖的局部化算法》[C]//第13届ACM年度移动计算与网络国际会议论文集. 2007: 63-74.[15] Cheng T M, Savkin A V.《一种用于移动无线传感器网络覆盖的分布式自部署算法》[J]. IEEE Communications Letters, 2009, 13(11): 877-879.[16] Ban D, Jiang J, Yang W 等.《利用移动传感器实现强k-屏障覆盖》[C]//第6届国际无线通信与移动计算会议论文集. 2010: 68-72.[17] Bhattacharya B, Burmester M, Hu Y 等.《移动传感器在平面区域屏障覆盖中的最优移动》[J]. Theoretical Computer Science, 2009, 410(52): 5515-5528.[18] Song C, Fan Y.《圆形区域内有限通信范围移动传感器网络的覆盖控制》[J]. Automatica, 2018, 92: 155-161.[19] C. Zhai, F. He, Y. Hong, L. Wang and Y. Yao, 《针对涉及诱饵目标的多拦截器基于覆盖的拦截算法》. AIAA Journal of Guidance, Control, and Dynamics, 第1-7页, 2016。