frontiers in Robotics and AI

ORIGINAL RESEARCH
published: 20 April 2021
doi: 10.3389/frobt.2021.640446

Aerial Swarm Defense by StringNet Herding: Theory and Experiments

Vishnu S. Chipade[1]*, *Venkata Sai Aditya Marella*[2] *and Dimitra Panagou*[1]

[1] *Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI, United States,* [2] *Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, United States*

This paper studies a defense approach against one or more swarms of adversarial agents. In our earlier work, we employed a closed formation ("StringNet") of defending agents (defenders) around a swarm of adversarial agents (attackers) to confine their motion within given bounds, and guide them to a safe area. The adversarial agents were assumed to remain close enough to each other, i.e., within a prescribed connectivity region. To handle situations when the attackers no longer stay within such a connectivity region, but rather split into smaller swarms (clusters) to maximize the chance or impact of attack, this paper proposes an approach to learn the attacking sub-swarms and reassign defenders toward the attackers. We use a "Density-based Spatial Clustering of Application with Noise (DBSCAN)" algorithm to identify the spatially distributed swarms of the attackers. Then, the defenders are assigned to each identified swarm of attackers by solving a constrained generalized assignment problem. We also provide conditions under which defenders can successfully herd all the attackers. The efficacy of the approach is demonstrated via computer simulations, as well as hardware experiments with a fleet of quadrotors.

Keywords: decision making, swarm defense, autonomous robots and drones, cooperative control feedback, multi-agent system

OPEN ACCESS

Edited by:
George C. Karras,
University of Thessaly, Greece

Reviewed by:
Philip M. Dames,
Temple University, United States
Ali Emre Turgut,
Middle East Technical University, Turkey

Correspondence:
Vishnu S. Chipade
vishnuc@umich.edu

Specialty section:
This article was submitted to Multi-Robot Systems, a section of the journal Frontiers in Robotics and AI

Received: *11 December 2020*
Accepted: *08 March 2021*
Published: *20 April 2021*

Citation:
Chipade VS, Marella VSA and Panagou D (2021) Aerial Swarm Defense by StringNet Herding: Theory and Experiments. Front. Robot. AI 8:640446. doi: 10.3389/frobt.2021.640446

1. INTRODUCTION

Rapid advancements in swarm technology and its increasing presence in airspace pose significant threat to safety-critical infrastructure such as government facilities, airports, and military bases. The presence of adversarial swarms nearby such entities, with the aim of causing physical damage or collecting critical information, can lead to catastrophic consequences. This necessitates solutions for the protection of safety-critical infrastructure against such attacks, particularly in populated areas.

Counteracting an adversarial swarm by means of physical interception, as studied in Chen et al. (2017), Coon and Panagou (2017), and Shishika et al. (2020), may not be desirable at low altitudes in an urban environment due to human presence. Under the assumption of risk-averse and self-interested adversarial agents (attackers) that tend to move away from the defending agents (defenders) and from other dynamic objects, *herding* can be used as an indirect way of guiding the attackers to some safe area in order to safe-guard a safety-critical area (protected area).

In our recent work Chipade and Panagou (2019, 2020b), we developed a herding algorithm, called "StringNet Herding," to herd a swarm of attackers away from a protected area. A closed formation ("StringNet") of defending agents connected by string barriers is formed around a swarm of attackers to confine their motion within given bounds, and guide them to a safe area. However, the assumption that the attackers stay together in a circular region, and that they react to the defenders collectively as a single swarm while attacking the protected area, can be quite conservative in practice.

In this paper, we build upon our earlier work on "StringNet Herding" (Chipade and Panagou, 2020b) and study the problem of defending a protected area from attackers that *may* or *may not* stay together throughout their attack. We propose a "Multi-Swarm StringNet Herding" approach that uses clustering-based defender assignment, and the "StringNet Herding" method to herd the multiple swarms adversarial attackers' to known safe areas.

1.1. Related Work

Herding has been studied earlier in the literature, see for instance (Haque et al., 2011; Paranjape et al., 2018; Pierson and Schwager, 2018). The approach in Paranjape et al. (2018) uses an *n*-wavefront algorithm to herd a flock of birds away from an airport, where the birds on the boundary of the flock are influenced based on the locations of the airport and a safe area.

The herding method in Pierson and Schwager (2018) utilizes a circular-arc formation of herders to influence the non-linear dynamics of the herd based on a potential-field approach, and designs a point-offset controller to guide the herd close to a specified location. In Haque et al. (2011), biologically-inspired strategies are developed for confining a group of agents; the authors develop strategies based on the "wall" and "encirclement" methods that dolphins use to capture a school of fish. In addition, they compute regions from which this confinement is possible; however, the results are limited to constant-velocity motion. A similar approach called *herding by caging* is adopted in Varava et al. (2017), where a cage of high potential is formed around the attackers. An RRT approach is used to find a motion plan for the agents; however, the cage is assumed to have already been formed around the agents, while the caging of the agents thereafter is only ensured with constant velocity motion under additional assumptions on the distances between the agents. Forming such a cage could be more challenging in case of self-interested, risk-averse attackers under non-constant velocity motion.

In Licitra et al. (2017, 2018), the authors discuss herding using a switched-system approach; the herder (defender) chases targets (evaders/attackers) sequentially by switching among them so that certain dwell-time conditions are satisfied to guarantee stability of the resulting trajectories. However, the assumption that only one of the targets is influenced by the herder at any time might be limiting and non-practical in real applications. The authors in Deptula et al. (2018) use approximate dynamic programming to obtain suboptimal control policies for the herder to chase a target agent to a goal location. A game-theoretic formulation is used in Nardi et al. (2018) to address the herding problem by constructing a virtual barrier similar to Pierson and Schwager (2018). However, the computational complexity due to the discretization of the state and control-action spaces limits its applicability.

Most of the aforementioned approaches for herding are limiting due to one or some of the following aspects: (1) simplified motion models (Varava et al., 2017; Pierson and Schwager, 2018), (2) absence of obstacles in the environment (Licitra et al., 2017, 2018; Paranjape et al., 2018), (3) no consideration of intra-team collisions (Varava et al., 2017; Pierson and Schwager, 2018), (4) assumption on a particular form of potential field to model the repulsive motion of the attackers with respect to the defenders (Licitra et al., 2017, 2018; Paranjape et al., 2018; Pierson and Schwager, 2018).

We have addressed the above issues in our recent work Chipade and Panagou (2019, 2020b), which develops a method termed as "StringNet Herding," for defending a protected area from a swarm of attackers in a 2D obstacle environment. In "StringNet Herding," a closed formation of strings ("StringNet") is formed by the defenders to surround the swarm of attackers. It is assumed that the attackers will stay together within a circular footprint as a swarm and collectively avoid the defenders. It is also assumed that the string between two defenders serves as a barrier through which the attackers cannot escape (e.g., a physical straight-line barrier, or some other mechanism). The StringNet is then controlled to herd the swarm of attackers to a safe area. The control strategy for the defenders in "StringNet Herding" is a combination of time-optimal control actions and finite-time, state-feedback, bounded control actions, so that the attackers can be herded to safe area in a timely manner.

Clustering of data points is a popular machine learning technique (Xu and Tian, 2015). There are various types of clustering algorithms: partition based (K-means; MacQueen et al., 1967), hierarchy based (BIRCH; Zhang et al., 1996), density based (DBSCAN; Ester et al., 1996), stream based (STREAM; O'Callaghan et al., 2002), graph based (CLICK; Sharan and Shamir, 2000). In the context of dynamical systems, authors in Cai et al. (2017) develop a clustering method based on quasi-consensus motions of dynamic agents where agents belonging to a particular cluster are expected to aggregate together. This method however converges to clustering results asymptotically. In this paper, we are interested in spatial proximity of the agents during a finite future time. So, we focus mostly on the density based approaches, for example, DBSCAN, to solve the clustering problem in this paper.

Assignment problems have also been studied extensively (Burkard et al., 2012). A detailed survey of an assignment problem pertaining to defense scenarios called "weapon-target assignment (WTA)" problem is provided in Kline et al. (2019). In general, assignment problems are NP-hard, and hence can be solved only approximately for large number of decision variables. For example, authors in Rezende et al. (2018) provide a greedy approach based on ant colony system to solve the WTA problem. Multi-agent defense problems are difficult to solve optimally because the problem becomes computationally intractable for large number of agents. In such cases, all possible pairwise games are first solved, and then an assignment problem is solved to assign defending agents against attacking ones based on the cost of the pairwise games. For example in Chen et al. (2017) and Coon and Panagou (2017), after solving the pairwise games, the defenders are assigned to attackers by solving a bipartite matching problem using Hungarian algorithm (Kuhn, 1955). Similarly, in Yan et al. (2019) authors solve a mixed integer program to find assignment of defenders to attackers in a multiplayer reach-avoid game played in a convex domain. In this paper, we are interested in a generalized assignment problem (GAP) (Öncan, 2007), in which there are more number of objects than knapsacks to be filled, because we aim to assign groups of defenders to a number of attackers' clusters, which are typically small in number than the number of defenders. Similar to the standard assignment problems, GAP is known to be NP-hard, but there are approximation algorithms to solve an arbitrary instance of GAP (Öncan, 2007).

1.2. Overview of the Proposed Approach

In the preliminary work presented in Chipade and Panagou (2020c), we extended the "StringNet Herding" approach to scenarios where attackers no longer stay together and may split into smaller swarms in reaction to the defenders' presence. The proposed approach involves: (1) identification of the clusters (swarms) of the attackers that stay together, (2) distribution and assignment of the defenders to each of the identified swarms of the attackers, (3) use of "StringNet Herding" approach by the defenders to herd each identified swarm of attackers to the closest safe area.

More specifically, we use the "Density based Spatial Clustering of Application with Noise (DBSCAN)" algorithm (Ester et al., 1996) to identify swarms of the attackers based on the proximity of the attackers to each other. We then formulate a generalized assignment problem with additional constraints on the connectivity of the defenders, to find which defender should go against which swarm of attackers and herd it to one of the safe areas. This connectivity constrained generalized assignment problem (C2GAP) is modeled as a mixed integer quadratically constrained program (MIQCP) to obtain an optimal assignment solution. Additionally, we provide a hierarchical algorithm to find the assignment quickly.

In this paper, we further improve the clustering based multi-swarm herding approach by developing a decentralized variant of the MIQCP that is used to assign the defenders to the identified swarms of the attackers. Furthermore, we address the question of whether the defenders starting at some known positions can gather on the shortest path of the attackers, starting at some states, to the protected area and can successfully herd the attackers to safe areas. We also demonstrate the "StringNet Herding" algorithm for single attacking swarm case via hardware experiments with a fleet of quadrotor vehicles that are capable of flying autonomously in an outdoor aerial robotics facility at the University of Michigan campus.

1.3. Summary of our Contributions

In summary, compared to the prior literature and our prior work presented in the conference version (Chipade and Panagou, 2020c), the novelties and contribution of this work are:

- a decentralized cooperative algorithm to group and assign the defenders to herd the identified different swarms of the attackers to the closest safe areas;
- a set of conditions under which the defenders are able to gather on the shortest path of the oncoming attackers to the protected area before the attackers could reach the gathering location and thereafter herd all the attackers to the safe areas;
- a demonstration of the "StringNet Herding" approach for a single attacking swarm case in hardware experiments using a fleet of quadrotor vehicles equipped with autonomous flight capability.

1.4. Structure of the Paper

Section 2 describes the mathematical modeling and problem statement. The "StringNet Herding" approach to herd a single swarm of attackers is briefly discussed in section 3. The approach on identification of attackers' swarms (clusters) and the defenders' assignment to these identified swarms for multiple-swarm herding is discussed in section 4. Simulations are provided in section 7, and the hardware experiments are discussed in section 8. Finally, the paper is concluded in section 10.

2. MODELING AND PROBLEM STATEMENT

Notation: We use $\mathbf{r}$, $\mathbf{v}$ and $\mathbf{u}$ to denote position, velocity and input acceleration vector, respectively. We use $\boldsymbol{\xi}$ and $\boldsymbol{\eta}$ to denote desired position and velocity vector, respectively. We use A and D to denote the indices of attackers and defenders, respectively, while I denotes order set of positive integers starting at 1. The variables ai, ac_k, dj, dc_k used as subscripts of the above variables correspond to the i^{th} attacker, center of mass of k^{th} swarm of attackers, j^{th} defender and the k^{th} group of defenders, respectively. Similarly, subscripts p, sm denote the protected area and m^{th} safe area, respectively. We use subscript d to denote common variables that correspond to all the defenders and subscripts a to denote common variables corresponding to all the attackers. We use sn and sb as a subscripts to denote StringNet and string barrier, respectively. Any variable with superscript g, s, e, h correspond to gathering, seeking, enclosing and herding phase, respectively.

The set of integers greater than 0 is denoted by $\mathbb{Z}_{>0}$. $\|.\|$ denotes the Euclidean norm of its argument. $|.|$ denotes the absolute value of a scalar, and cardinality if the argument is a set. $\lfloor\cdot\rfloor$ gives the largest integer smaller than the argument number. A ball of radius ρ centered at the origin is defined as $\mathcal{B}_\rho = \{\mathbf{r} \in \mathbb{R}^2 | \|\mathbf{r}\| \le \rho\}$.

We consider N_a attackers $\mathcal{A}_i$, $i \in I_a = \{1, 2, ..., N_a\}$, and N_d defenders $\mathcal{D}_j$, $j \in I_d = \{1, 2, ..., N_d\}$, operating in a 2D environment $\mathcal{W} \subseteq \mathbb{R}^2$ that contains a protected area $\mathcal{P} \subset \mathcal{W}$, defined as $\mathcal{P} = \{\mathbf{r} \in \mathbb{R}^2 \mid \left\|\mathbf{r} - \mathbf{r}_p\right\| \le \rho_p\}$, and N_s safe areas $\mathcal{S}_m \subset \mathcal{W}$, defined as $\mathcal{S}_m = \{\mathbf{r} \in \mathbb{R}^2 \mid \|\mathbf{r} - \mathbf{r}_{sm}\| \le \rho_{sm}\}$, for all $m \in I_s = \{1, 2, ..., N_s\}$, where $(\mathbf{r}_p, \rho_p)$ and $(\mathbf{r}_{sm}, \rho_{sm})$ are the centers and radii of the corresponding areas, respectively. The attackers aim to reach the protected area $\mathcal{P}$. The attackers may use flocking controllers (Dai and Li, 2014) to stay together, or they may choose to split into different smaller swarms (Raghuwaiya et al., 2016; Goel et al., 2019). The defenders aim to herd each of these attackers to one of the safe areas in $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_{N_s}\}$ before they reach $\mathcal{P}$.

The agents $\mathcal{A}_i$ and $\mathcal{D}_j$ are modeled as discs of radii ρ_a and $\rho_d(\le \rho_a)$, respectively and move under double integrator (DI)

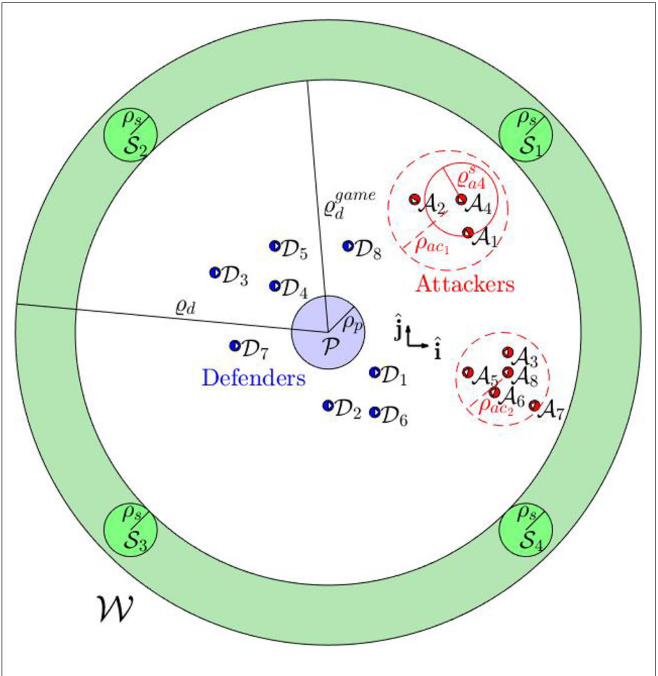

FIGURE 1 | Problem formulation.

dynamics with quadratic drag (damped double integrator):

$$\dot{\mathbf{r}}_{ai} = \mathbf{v}_{ai}, \qquad \dot{\mathbf{v}}_{ai} = \mathbf{u}_{ai} - C_D \left\|\mathbf{v}_{ai}\right\| \mathbf{v}_{ai}; \tag{1}$$

$$\dot{\mathbf{r}}_{dj} = \mathbf{v}_{dj}, \qquad \dot{\mathbf{v}}_{dj} = \mathbf{u}_{dj} - C_D \left\|\mathbf{v}_{dj}\right\| \mathbf{v}_{dj}; \tag{2}$$

$$\left\|\mathbf{u}_{ai}\right\| \le \bar{u}_a, \quad \left\|\mathbf{u}_{dj}\right\| \le \bar{u}_d, \tag{3}$$

where C_D is the drag coefficient, $\mathbf{r}_{ai} = [x_{ai}, \ y_{ai}]^T$ and $\mathbf{r}_{dj} = [x_{dj}, \ y_{dj}]^T$ are the position vectors of $\mathcal{A}_i$ and $\mathcal{D}_j$, respectively; $\mathbf{v}_{ai} = [v_{x_{ai}}, \ v_{y_{ai}}]^T$, $\mathbf{v}_{dj} = [v_{x_{dj}}, \ v_{y_{dj}}]^T$ are the velocity vectors, respectively, and $\mathbf{u}_{ai} = [u_{x_{ai}}, \ u_{y_{ai}}]^T$, $\mathbf{u}_{dj} = [u_{x_{dj}}, \ u_{y_{dj}}]^T$ are the accelerations (the control inputs), respectively. This model poses a speed bound on each player with limited acceleration control, i.e., $v_{ai} = \left\|\mathbf{v}_{ai}\right\| < \bar{v}_a = \sqrt{\frac{\bar{u}_a}{C_d}}$ and $v_{dj} = \left\|\mathbf{v}_{dj}\right\| < \bar{v}_d = \sqrt{\frac{\bar{u}_d}{C_d}}$. The defenders are assumed to be faster than the attackers, i.e., $\bar{v}_a < \bar{v}_d$ (i.e., $\bar{u}_a < \bar{u}_d$). The number of defenders is assumed to be no less than that of attackers, i.e., $N_d \ge N_a$.

There is a distributed navigation system that senses the position $\mathbf{r}_{ai}$ and velocity $\mathbf{v}_{ai}$ of the attacker $\mathcal{A}_i$ that lies inside a circular sensing zone $\mathcal{Z}_d = \{\mathbf{r} \in \mathbb{R}^2 | \ \left\|\mathbf{r} - \mathbf{r}_{pa}\right\| \le \varrho_d\}$ for all $i \in I_a$, where $\varrho_d > 0$ is the radius of the defenders' sensing zone. The navigation system communicates the sensed information to the defenders $\mathcal{D}_j$, for all $j \in I_d$. Every attacker $\mathcal{A}_i$ has a local sensing zone $\mathcal{Z}_{ai} = \{\mathbf{r} \in \mathbb{R}^2 \mid \left\|\mathbf{r} - \mathbf{r}_{ai}\right\| \le \varrho_{ai}\}$, where $\varrho_{ai} > 0$ is the radius of the attacker $\mathcal{A}_i$'s sensing zone (**Figure 1**). This navigation system can include sensors such as radars, lidars, cameras that are spatially distributed around the protected area and provide measurements of positions and velocities of the attackers and the defenders. The defenders are also assumed to have sufficient computational power available on board to solve the assignment problems that are discussed later in the paper.

Formally, we consider the followin g problems.

Problem 1 (Swarm Identification). *Identify the swarms $\{\mathcal{A}_{c_1}, \mathcal{A}_{c_2}, ..., \mathcal{A}_{c_{N_{ac}}}\}$ of the attackers for some unknown $N_{ac} \ge 1$ such that attackers in the same swarm $\mathcal{A}_{c_k}$, and only them, are physically close to each other and satisfy prescribed conditions (described later) on spatial density, where $\mathcal{A}_{c_k} = \{\mathcal{A}_i | i \in A_{c_k}\}$, $A_{c_k} \subseteq I_a$, for all $k \in I_{ac} = \{1, 2, ..., N_{ac}\}$.*

Problem 2 (Multi-Swarm Herding). *Find subgroups $\{\mathcal{D}_{c_1}, \mathcal{D}_{c_2}, ..., \mathcal{D}_{c_{N_{ac}}}\}$ of the defenders and their assignment to the attackers' swarms identified in Problem 1, such that all the defenders in the same subgroup are connected via string barriers to enclose and herd the assigned attacker's swarm.*

Problem 3 (Defenders' Dominance Region). *Given the initial positions of the defenders $\mathbf{r}_{dj}(0)$, for all $j \in I_d$, provide conditions on the initial positions $\mathbf{r}_{ai}(0)$, for all $i \in I_a$, of the attackers for which the defenders are able to gather as a specified formation centered at a point on the expected path of the attackers before any attacker reaches the center of the formation.*

Before we discuss the solutions to the above three problems, we first briefly describe the "StringNet Herding" approach used to herd a single swarm of the attackers in the following section.

3. HERDING A SINGLE SWARM OF ATTACKERS

To herd a swarm of attackers to $\mathcal{S}$, we use "StringNet Herding," developed in Chipade and Panagou (2020b). StringNet is a closed net of strings formed by the defenders as shown in **Figure 3**. The strings are realized as impenetrable and extendable line barriers (e.g., spring-loaded pulley and a rope or other similar mechanism; Mirjan et al., 2016) that prevent attackers from passing through them. The extendable string barrier allows free relative motion of the two defenders connected by the string. The string barrier can have a maximum length of $\bar{R}_{sb} > 0$. If the string barrier were to be physical one, then it can be established between two defenders $\mathcal{D}_j$ and $\mathcal{D}_{j'}$ only when they are close to each other and have almost same velocity, i.e., $\left\|\mathbf{r}_{dj} - \mathbf{r}_{dj'}\right\| \le \epsilon_1 < \bar{R}_{sb}$ and $\left\|\mathbf{v}_{dj} - \mathbf{v}_{dj'}\right\| \le \epsilon_2$, where ϵ_1 and ϵ_2 are small numbers. The underlying graph structure for the two different "StringNet" formations defined for a subset of defenders $\mathcal{D}' = \{\mathcal{D}_j \mid j \in I'_d\}$, where $I'_d \subseteq I_d$, are defined as follows:

Definition 1 (Closed-StringNet). *The Closed-StringNet $\mathcal{G}^{cl}_{sn}(I'_d) = (\mathcal{V}^{cl}_{sn}(I'_d), \mathcal{E}^{cl}_{sn}(I'_d))$ is a cycle graph consisting of: 1) a subset of defenders as the vertices, $\mathcal{V}^{cl}_{sn}(I'_d) = \{\mathcal{D}_j \mid j \in I'_d\}$, 2) a set of edges, $\mathcal{E}^{cl}_{sn}(I'_d) = \{(\mathcal{D}_j, \mathcal{D}_{j'}) \in \mathcal{V}^{cl}_{sn}(I'_d) \times \mathcal{V}^{cl}_{sn}(I'_d) | \mathcal{D}_j \overset{s}{\longleftrightarrow} \mathcal{D}_{j'}\}$, where the operator $\overset{s}{\longleftrightarrow}$ denotes an impenetrable line barrier between the defenders.*

Definition 2 (Open-StringNet). *The Open-StringNet $\mathcal{G}^{op}_{sn}(I'_d) = (\mathcal{V}^{op}_{sn}(I'_d), \mathcal{E}^{op}_{sn}(I'_d))$ is a path graph consisting of: 1) a set of vertices, $\mathcal{V}^{op}_{sn}(I'_d)$ and 2) a set of edges, $\mathcal{E}^{op}_{sn}(I'_d)$, similar to that in Definition 1.*

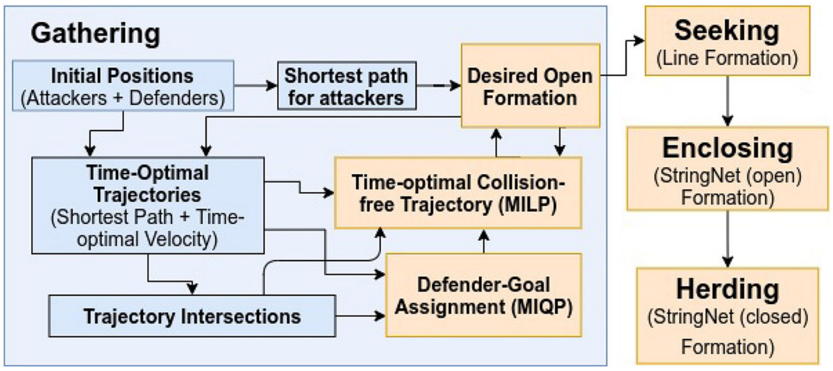
Gathering
Initial Positions (Attackers + Defenders)
Shortest path for attackers
Desired Open Formation
Time-Optimal Trajectories (Shortest Path + Time-optimal Velocity)
Time-optimal Collision-free Trajectory (MILP)
Defender-Goal Assignment (MIQP)
Trajectory Intersections
Seeking (Line Formation)
Enclosing (StringNet (open) Formation)
Herding (StringNet (closed) Formation)

FIGURE 2 | Block diagram for "StringNet Herding".

The StringNet herding consists of four phases: (1) gathering, (2) seeking,(3) enclosing, and (4) herding to a safe area, see the block diagram in **Figure 2**.

These phases are discussed as follows.

3.1. Gathering

We assume that the attackers start as single swarm that stays together and they may start splitting into smaller groups as they sense the defenders in their path. The aim of the defenders is to converge to an open formation $\mathscr{F}_d^g$ centered at the gathering center $\mathbf{r}_{dc^g}$ located on the expected path of the attackers, where the expected path is defined as the shortest path of the attackers to the protected area, before the attackers reach $\mathbf{r}_{dc^g}$ (see **Figure 3**). Let $\mathscr{R}_d(N_a) : \mathbb{Z}_{>0} \to \mathbb{Z}_{>0}$ be the resource allocation function that outputs the number of the defenders that can be assigned to the given N_a attackers. We make the following assumption about the resource allocation function.

Assumption 1. *The resource allocation function is a strictly monotonically increasing function, i.e.,* $\mathscr{R}_d(N_a) < \mathscr{R}_d(N_a + 1)$ *and satisfies* $\mathscr{R}_d(N_a) \geq N_a$.

Assumption 1 ensures that there are adequate number of defenders to go after each attacker in the event the attackers in the swarm disintegrate into singular swarms (swarms with less than 3 attackers). In the case of large number of singular swarms, herding may not be the most economical way of defense as there needs to be at least 3 defenders to form a Closed-StringNet. In which case, one could employ a different mechanism to counteract the attack, for example, physical capture or interception of the attacker. In this paper we only the consider the attackers' swarms with greater than or equal to 3 attackers. The case of singular swarms will be studied in our future work.

The open formation $\mathscr{F}_d^g$ is characterized by the positions $\boldsymbol{\xi}_l^g$, for all $l \in I_{dc_0} = \{1, 2, ..., \mathscr{R}_d(N_a)\}$, and is chosen to be a straight line formation[1] (see **Figure 3**). Once the defenders arrive at these positions, the defenders get connected by strings as follows: the defender at $\boldsymbol{\xi}_l^g$ gets connected to the defender at $\boldsymbol{\xi}_{l+1}^g$ for all $l \in I_{dc_0}^- = \{1, 2, ..., \mathscr{R}_d(N_a) - 1\}$ (see **Figure 3**). The angle made by the normal to the line joining $\boldsymbol{\xi}_1^g$ and $\boldsymbol{\xi}_{N_d}^g$ (clockwise from $\boldsymbol{\xi}_1^g$, see **Figure 3**) is the orientation ϕ of the formation. The formation $\mathscr{F}_d^g$ is chosen such that its orientation is toward the attackers on their expected path (defined above), see the blue formation in **Figure 3**. The desired positions $\boldsymbol{\xi}_l^g$ on $\mathscr{F}_d^g$ centered at $\mathbf{r}_{dc^g}$ are:

$$\boldsymbol{\xi}_l^g = \mathbf{r}_{dc^g} + R_l \hat{\mathbf{o}}(\theta_{dc^g} + \tfrac{\pi}{2}), \quad \text{for all } l \in I_{dc_0}; \tag{4}$$

where $R_l = 0.5\left(\mathscr{R}_d(|A_{c_k}|) - 2l + 1\right) R_{sb}^g$, $\hat{\mathbf{o}}(\theta) = [\cos(\theta), \ \sin(\theta)]^T$ is the unit vector making an angle θ with x-axis, $\theta_{dc^g} = \theta_{a_{cm}}^* + \pi$, where $\theta_{a_{cm}}^*$ is the angle made by the line segment joining the attackers' center of mass (ACoM) to the center of the protected area (the shortest path from the initial position of ACoM to $\mathcal{P}$) with x-axis. These positions are static, i.e., $\dot{\boldsymbol{\xi}}_l^g = \ddot{\boldsymbol{\xi}}_l^g = \mathbf{0}$. The gathering center $\mathbf{r}_{dc^g} = \rho_{df}^g \hat{\mathbf{o}}(\theta_{dc^g})$ is such that $\rho_{df}^g > \rho_p$. We define the defender-goal assignment as:

Definition 3 (Defender-Goal Assignment)**.** *An injective mapping* $\beta_0 : \{1, 2, ..., \mathscr{R}_d(N_a)\} \to I_d$ *such that the defender* $\mathcal{D}_{\beta_0(l)}$ *is assigned to go to the goal* $\boldsymbol{\xi}_l^g$.

As discussed in Algorithm 1 in Chipade and Panagou (2020b) and as shown in **Figure 2**, we design a time-optimal motion plan so that the defenders gather at the desired formation $\mathscr{F}_d^g$ as early as possible and before the attackers reach close to $\mathscr{F}_d^g$. The idea in Algorithm 1 in Chipade and Panagou (2020b) is to iteratively solve a mixed integer quadratic program (MIQP) until a gathering center for the gathering formation is found which is as far as possible from the protected area and such that the defenders are able to gather at the formation $\mathscr{F}_d^g$ centered at the gathering center with bounded acceleration before the attackers can.

[1]This is a better choice compared to a semicircular formation as chosen in Chipade and Panagou (2020b). Because, the semicircular formation, for a given length constraint on the string barrier ($\bar{R}_{sb}$), creates smaller blockage to the attackers as compared to the line formation. Although, Completing a circular formation starting from a semicircular formation of the same radius is faster. It is a trade-off between effectiveness and speed.

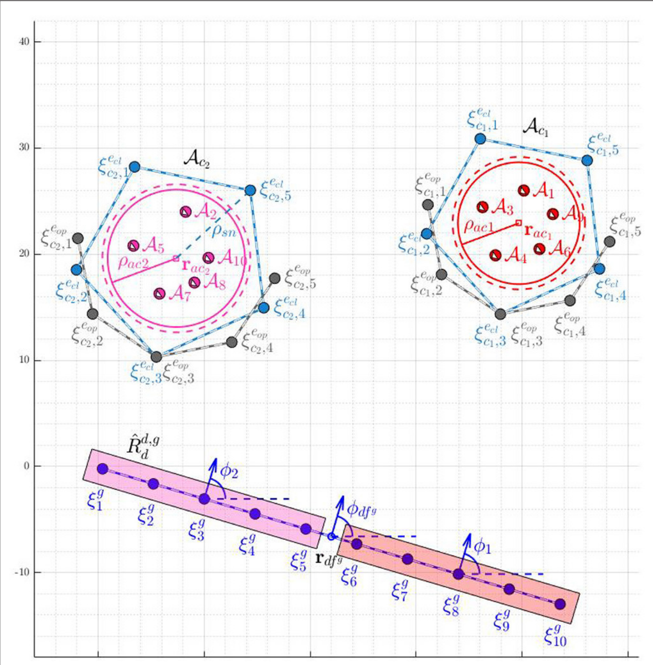

FIGURE 3 | Assignment of defenders to the attackers' swarms.

3.2. Seeking

After the defenders accomplish gathering, suppose a group of defenders $\mathcal{D}_{c_k} = \{\mathcal{D}_j | j \in D_{c_k}\}$, $D_{c_k} \subseteq I_d$, is tasked to herd a swarm of attackers $\mathcal{A}_{c_k} = \{\mathcal{A}_i | i \in A_{c_k}\}$, $A_{c_k} \subseteq I_a$, the details are discussed later in section 4. Denote $I_{dc_k} = \{1, 2, ..., |\mathcal{D}_{c_k}|\}$ and let $\beta_k : I_{dc_k} \to D_{c_k}$ be the mapping that gives the indexing order of the defenders in $\mathcal{D}_{c_k}$ on the Open-StringNet line formation $\mathscr{F}^s_{dc_k}$ (similar to $\mathscr{F}^g_d$ but with $\bar{R}_{sb}$ being the distance of each defender from their immediate neighbor). In the seeking phase, the defenders in $\mathcal{D}_{c_k}$ maintain the line formation $\mathscr{F}^s_{dc_k}$ and try to get closer to the swarm of attackers $\mathcal{A}_{c_k}$ by using state-feedback, finite-time convergent, bounded control laws as discussed in Chipade and Panagou (2020b). The control actions in Chipade and Panagou (2020b) for the defenders in $\mathcal{D}_{c_k}$ are modified to incorporate collision avoidance from the other StringNet formations formed by $\mathcal{D}_{c_{k'}}$, for $k' \neq k$.

3.3. Enclosing: Closed-StringNet Formation

Once the Open-StringNet formation reaches close to the attackers' formation, the defenders start enclosing the attackers by moving to their desired positions on the enclosing formations while staying connected to their neighbors. We choose two formations for this phase that the defenders sequentially achieve: (1) Semi-circular Open-StringNet formation ($\mathscr{F}^{eop}_{dc_k}$), (2) Circular Closed-StringNet formation ($\mathscr{F}^{ecl}_{dc_k}$). When the defenders directly try to converge to a circular formation from a line formation during this phase, the defenders at the either end of the Open-StringNet formation will start coming closer to each other reducing the length of the overall barrier in the attackers' path significantly. This is because the desired positions of these terminal defenders in the circular formation would be very close to each other on the opposite side of the circular formation (see **Figure 3**) and collision avoidance part of the controller is only active locally near the circle of maximum radius $\bar{\rho}_{ac_k}$ around the swarm $\mathcal{A}_{c_k}$. So the defenders would first converge to a semi-circular formation and would converge to a circular formation after the former is achieved.

The desired position $\boldsymbol{\xi}^{eop}_{c_k,l}$ on the Open-StringNet formation $\mathscr{F}^{eop}_{dc_k}$ (**Figure 3**) is chosen on the circle with radius ρ_{sn_k} centered at $\mathbf{r}_{ac_k}$ as:

$$\boldsymbol{\xi}^{eop}_{c_k,l} = \mathbf{r}_{ac_k} + \rho_{sn_k}\hat{\mathbf{o}}(\theta_l), \text{ where } \theta_l = \theta^{e*}_{dc_k} + \frac{\pi}{2} + \frac{\pi(l-1)}{|\mathcal{D}_{c_k}|-1}, \quad (5)$$

for all $l \in I_{dc_k}$, where $\theta^{e*}_{dc_k} = \theta^{s*}_{dc_k}$. $\mathbf{r}_{ac_k} = \sum_{i \in I_{ac_k}} \frac{\mathbf{r}_{ai}}{|\mathcal{A}_{c_k}|}$ is the center of mass of $\mathcal{A}_{c_k}$. The radius ρ_{sn_k} should satisfy, $\bar{\rho}_{ac_k} + b_d < \rho_{sn_k}$, where $\bar{\rho}_{ac_k}$ is maximum radius of swarm $\mathcal{A}_{c_k}$. The parameter b_d is the tracking error for the defenders in this phase (Chipade and Panagou, 2020b).

Similarly, the desired positions $\boldsymbol{\xi}^{ecl}_{c_k,l}$ on the Closed-StringNet formation $\mathscr{F}^{ecl}_{dc_k}$ same as in Equation (5) with $\theta_l = \theta^{e*}_{dc_k} + \frac{\pi(2l-1)}{|\mathcal{D}_{c_k}|}$, for all $l \in I_{dc_k}$. Both formations move with the same velocity as that of the attackers' center of mass, i.e., $\dot{\boldsymbol{\xi}}^{eop}_{c_k,l} = \dot{\boldsymbol{\xi}}^{ecl}_{c_k,l} = \dot{\mathbf{r}}_{ac_k}$.

The defenders $\mathcal{D}_{c_k}$ first track the desired goal positions $\boldsymbol{\xi}^{eop}_{c_k,l}$ by using the finite-time convergent, bounded control actions given in Chipade and Panagou (2020b). Once the defender $\mathcal{D}_{\beta_k(1)}$ and $\mathcal{D}_{\beta_k(|\mathcal{D}_{c_k}|)}$ reach within a distance of b_d from $\boldsymbol{\xi}^{eop}_{c_k,1}$ and $\boldsymbol{\xi}^{eop}_{c_k,|\mathcal{D}_{c_k}|}$, i.e., $\left\| \mathbf{r}_{d\beta_k(1)} - \boldsymbol{\xi}^{eop}_{c_k,1} \right\| < b_d$ and $\left\| \mathbf{r}_{d\beta_k(|\mathcal{D}_{c_k}|)} - \boldsymbol{\xi}^{eop}_{c_k,|\mathcal{D}_{c_k}|} \right\| < b_d$, respectively, the desired goal positions are changed from $\boldsymbol{\xi}^{eop}_{c_k,l}$ to $\boldsymbol{\xi}^{ecl}_{c_k,l}$ for all $l \in I_{dc_k}$. The StringNet is achieved when $\left\| \mathbf{r}_{d\beta_k(l)} - \boldsymbol{\xi}^{ecl}_{c_k,l} \right\| \leq b_d$ for all $l \in I_{dc_k}$ during this phase.

3.4. Herding: Moving the Closed-StringNet to Safe Area

Once a group of defenders $\mathcal{D}_{c_k}$ forms a StringNet around a swarm of attackers $\mathcal{A}_{c_k}$, they move while tracking a desired rigid closed circular formation $\mathscr{F}^h_{dc_k}$ centered at a virtual agent $\mathbf{r}_{dc^h_k}$ as discussed in Chipade and Panagou (2020b). The swarm is herded to the closest safe area $S_{\varsigma(k)}$, where $\varsigma(k) = \arg\min_{m \in I_s} \left\| \mathbf{r}_{dc^h_k} - \mathbf{r}_{sm} \right\|$.

4. MULTI-SWARM HERDING

In this section, we consider that the attackers split into smaller groups as they sense the defenders along their way to the protected area, to maximize the chance of at least some attackers reaching the protected area by circumnavigating the oncoming defenders. To respond to such strategic movements of the attackers, the defenders need to collaborate intelligently. In the approach presented in this paper, as shown in the block diagram in **Figure 4**, the defenders continuously keep checking

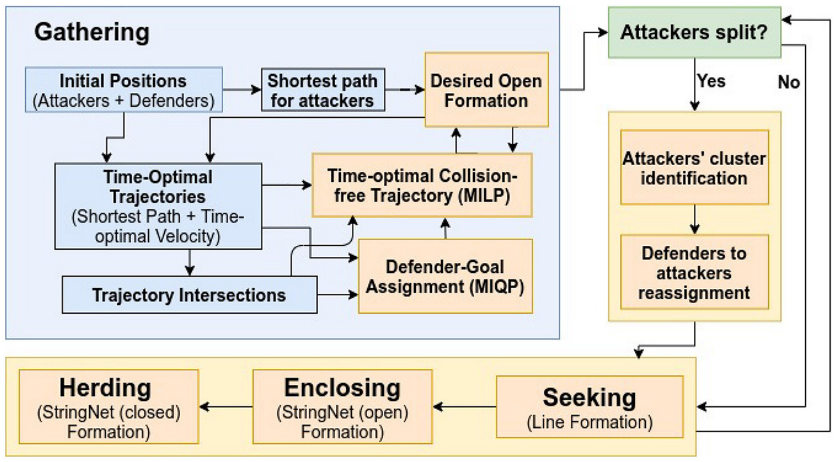
Gathering
Initial Positions (Attackers + Defenders)
Shortest path for attackers
Desired Open Formation
Time-Optimal Trajectories (Shortest Path + Time-optimal Velocity)
Time-optimal Collision-free Trajectory (MILP)
Trajectory Intersections
Defender-Goal Assignment (MIQP)
Attackers split?
Yes
No
Attackers' cluster identification
Defenders to attackers reassignment
Herding (StringNet (closed) Formation)
Enclosing (StringNet (open) Formation)
Seeking (Line Formation)

FIGURE 4 | Block diagram for "Multi-Swarm StringNet Herding".

whether the attackers have split, i.e., whether the attackers no longer satisfy certain spatial proximity constraints (defined later in the text). After a split event has happened, the defenders first identify the spatial clusters of the attackers. Then, the defenders distribute themselves into smaller connected groups, and these connected groups are assigned to the herd different spatial clusters (swarms) of the attackers to safe areas. Here, by "connected group of defenders" we mean that the defenders have already been connected via string barriers and established an Open-StringNet formation, see for example the defenders at the locations $\{\boldsymbol{\xi}_1^g, \boldsymbol{\xi}_2^g, ..., \boldsymbol{\xi}_5^g\}$ and $\{\boldsymbol{\xi}_6^g, \boldsymbol{\xi}_7^g, ..., \boldsymbol{\xi}_{10}^g\}$ as shown in **Figure 3**. In the next subsections, we discuss how the swarms (clusters) of the attackers are identified and how the defenders are assigned to these identified swarms of the attackers.

4.1. Identifying Swarms of the Attackers

In order to identify the spatially distributed clusters (swarms) of the attackers, the defenders utilize the Density Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm (Ester et al., 1996). Given a set of points, DBSCAN algorithm finds clusters of high density points (i.e., points with many nearby neighbors), and marks the points as outliers if they lie alone in low-density regions (whose nearest neighbors are too far away). DBSCAN algorithm can identify clusters of any shape in the data and requires two parameters that define the density of the points in the clusters: (1) ε_{nb} (radius of the neighborhood of a point), (2) m_{pts} (minimum number of points in ε_{nb}-neighborhood of a point). In general, attackers can split into formations with varied range of densities making the choice of the parameters ε_{nb} and m_{pts} challenging. There are variants of the DBSCAN algorithm, such as OPTICS (Ankerst et al., 1999), which can find clusters of varying density; however, they are more time consuming. Therefore, to keep the computational demands low, we use the DBSCAN algorithm with fixed parameters ε_{nb} and m_{pts}, which quickly yields useful clustering information about the attackers satisfying a specified connectivity constraints.

The neighborhood of an attacker is defined using weighted distance between two attackers: $d(\mathbf{x}_{ai}, \mathbf{x}_{ai'}) = \sqrt{(\mathbf{x}_{ai} - \mathbf{x}_{ai'})^T \mathbf{M} (\mathbf{x}_{ai} - \mathbf{x}_{ai'})}$, where $\mathbf{x}_{ai} = [\mathbf{r}_{ai}^T, \mathbf{v}_{ai}^T]^T$ and $\mathbf{M}$ is a weighing matrix defined as $\mathbf{M} = diag([1, 1, \varphi, \varphi])$, where φ weights relative velocity against relative position. We choose $\varphi < 1$ because relative position is more important in a spatial cluster than the velocity alignment at a given time instance. The ε_{nb}-neighborhood of an attacker $\mathcal{A}_i$ is then defined as the set of points $\mathbf{x} \in \mathbb{R}^2 \times \mathcal{B}_{\bar{v}_a}$ such that $d(\mathbf{x}_{ai}, \mathbf{x}) < \varepsilon_{nb}$.

The largest circle inscribed in the largest Closed-StringNet formation formed by the $\mathscr{R}_d(N_a)$ defenders has radius $\bar{\rho}_{ac} = \frac{\bar{R}_{sb}}{2} \cot(\frac{\pi}{\mathscr{R}_d(N_a)})$. Maximum radius of any cluster with N_a points identified by DBSCAN algorithm with parameters ε_{nb} and m_{pts} is $\frac{\varepsilon_{nb}(N_a-1)}{m_{pts}-1}$. If all of the attackers were to be a single swarm enclosed inside the region with radius $\bar{\rho}_{ac}$, then we would require ε_{nb} to be greater than $\frac{\bar{\rho}_{ac}(m_{pts}-1)}{N_a-1}$ in order to identify them as a single cluster. So we choose $\varepsilon_{nb} = \frac{\bar{\rho}_{ac}(m_{pts}-1)}{N_a-1}$, and since we want to identify clusters with as low as 3 agents, we need to choose $m_{pts} = 3$. With these parameters for DBSCAN algorithm, we have:

Lemma 1. *Let* $\mathcal{A}_c(t) = \{\mathcal{A}_{c_1}(t), \mathcal{A}_{c_2}(t), ..., \mathcal{A}_{c_{N_{ac}(t)}}(t)\}$ *be the clusters identified by DBSCAN algorithm with* $\varepsilon_{nb} = \frac{\bar{\rho}_{ac}(m_{pts}-1)}{(N_a-1)}$ *at time t, where* $N_{ac}(t)$ *is the total number of clusters at time t. If* $|\mathcal{A}_{c_k}(t)| > 3$ *and* $N_a = N_d$*, then the radius* $\rho_{ac_k}(t)$ *of the cluster* $\mathcal{A}_{c_k}(t)$ *satisfies* $\rho_{ac_k}(t) = \max_{i \in I_{ac_k}(t)} \left\| \mathbf{r}_{ai}(t) - \mathbf{r}_{ac_k}(t) \right\| \leq \frac{\bar{R}_{sb}}{2} \cot\left(\frac{\pi}{\mathscr{R}_d(|\mathcal{A}_{c_k}(t)|)}\right)$*, for all* $k \in I_{ac}(t) = \{1, 2, ..., N_{ac}(t)\}$.

As the number of attackers increases, the computational cost for DBSCAN becomes higher and looses its practical usefulness. Furthermore, the knowledge of the clusters is only required by the defenders when a swarm of attackers does not satisfy the assumed constraint on its connectivity radius so the defenders can be reconfigured and reassigned. So we continuously track the radii of the clusters and run the DBSCAN algorithm only when at some instant $t = t_{se}$ the connectivity constraint is violated by the swarms of attackers $\mathcal{A}_{c_k}(t_{se})$ for some $k \in I_{ac}(t_{se})$ i.e., when the radius $\rho_{ac_k}(t_{se})$ of the swarm of attackers $\mathcal{A}_{c_k}(t_{se})$ exceeds the value $\bar{\rho}_{ac_k}(t_{se}) = \frac{\bar{R}_{sb}}{2}\cot\left(\frac{\pi}{\mathscr{R}_d(N_a)}\right)\frac{|\mathcal{A}_{c_k}(t_{se})|-1}{N_a-1}$. The connectivity constraint violation is termed as split in this paper. The split event is defined as:

Definition 4 (Split event)**.** *An instant t_{se} when for any swarm $\mathcal{A}_{c_k}(t_{se})$, $k \in I_{ac}(t_{se})$, the radius of the swarm of attackers $\mathcal{A}_{c_k}(t_{se})$ defined as $\rho_{ac_k}(t_{se}) = \max_{i \in I_{ac_k}(t_{se})} \left\| \mathbf{r}_{ai}(t_{se}) - \mathbf{r}_{ac_k}(t_{se}) \right\|$ exceeds the value $\bar{\rho}_{ac_k}(t_{se})$.*

We also make the following assumption regarding the splitting behavior of the attackers.

Assumption 2. *Once a swarm of attackers splits, its member attackers never rejoin each other, i.e., for all $i \in I_a$, if $\exists\ t > 0$ such that $\mathcal{A}_i \notin \mathcal{A}_{c_k}(t)$ for any $k \in I_{ac}(t)$ then $\mathcal{A}_i \notin \mathcal{A}_{c_k}(t')$ for all $t \leq t'$.*

4.2. Defender Assignment to the Swarms of Attackers

The initially single-one swarm of attackers splits into smaller swarms that are being identified by the defenders. After then, the defenders must distribute themselves into smaller groups, and assign these groups to the attackers' swarms (clusters), so that subsequently they enclose the attackers' clusters and herd them to the closest safe area. Let $\mathcal{A}_c(t_{se}) = \{\mathcal{A}_{c_1}(t_{se}), \mathcal{A}_{c_2}(t_{se}), \ldots, \mathcal{A}_{c_{N_{ac}}}(t_{se})\}$ be a set of swarms of the attackers after a split event has happened at time t_{se}.

We assume that none of the swarms in $\mathcal{A}_c(t)$ is a singular one, i.e., a swarm with less than three agents, $|\mathcal{A}_{c_k}(t)| > 2$ for all $k \in I_{ac}(t),\ t \geq 0$. We formally define the defender to attackers' swarm assignment as:

Definition 5 (Defender-to-AttackSwarm Assignment)**.** *A set $\beta(t) = \{\beta_1(t,\cdot), \beta_2(t,\cdot), \ldots\beta_{N_{ac}}(t,\cdot)\}$ of mappings $\beta_k(t,\cdot): \{1,2,\ldots,\mathscr{R}_d(|\mathcal{A}_{c_k}(t)|)\} \to I_d$, where $\beta_k(t,\cdot)$ gives the indices of the defenders assigned to the swarm $\mathcal{A}_{c_k}(t)$ at time t for all $k \in I_{ac}(t)$.*

We want to find an assignment that minimizes the sum of distances of the defenders from the centers of the assigned attackers' swarms. This ensures that the collective effort needed by all the defenders is minimized when enclosing the swarms of the attackers. For successful enclosing of the newly formed attacking swarms, it is required that all the defenders that are assigned to an attackers' cluster are neighbors of each other, are already connected to each other via string barriers, and the underlying graph is an Open-StringNet.

Let $\mathcal{D}_c(t_{se}^-) = \{\mathcal{D}_{c_1}(t_{se}^-), \mathcal{D}_{c_2}(t_{se}^-), \ldots, \mathcal{D}_{c_{N_{dc}(t_{se}^-)}}(t_{se}^-)\}$ be a set of swarms of the defenders, where t_{se}^- denotes the instant immediately before $t = t_{se}$. Each of these swarms is already connected via a StringNet and was assigned to herd some cluster of attackers an instant before a split event happened at time t_{se}. Now that a split event has happened and new smaller clusters have been formed by the attackers, we seek to reassign the defenders $\mathcal{D}_c(t_{se}^-)$ to herd the newly formed clusters of the attackers.

This assignment problem is closely related to generalized assignment problem (GAP) (Öncan, 2007), in which n objects are to be filled in m knapsacks ($n \geq m$). This problem is modeled as a GAP with additional constraints on the objects (defenders) that are assigned to a given knapsack (attackers' swarm). The additional constraint on the defenders is to ensure their connectivity to each other within a newly formed swarm of defenders. So, we call this constrained assignment problem as connectivity constrained generalized assignment problem (C2GAP), and provide a mixed integer quadratically constrained program (MIQCP) to find the optimal assignment centrally as:

$$\boldsymbol{\delta}^*(t_{se}) = \underset{\boldsymbol{\delta}(t_{se})}{\arg\min} \quad \sum_{k=1}^{N_{ac}(t_{se})} \sum_{j=1}^{\mathscr{R}_d(N_a)} \left\| \mathbf{r}_{ac_k}(t_{se}) - \mathbf{r}_{dj}(t_{se}) \right\| \delta_{jk}(t_{se}) \tag{6a}$$

Subject to

$$\sum_{k \in I_{ac}(t_{se})} \delta_{jk}(t_{se}) = 1, \quad \forall j \in I_{dc_0}; \tag{6b}$$

$$\sum_{j \in I_{dc_0}} \delta_{jk}(t_{se}) = \mathscr{R}_d(|\mathcal{A}_{c_k}(t_{se})|), \quad \forall k \in I_{ac}(t_{se}); \tag{6c}$$

$$\sum_{k'=1}^{N_{dc}(t_{se}^-)} \sum_{l \in \tilde{I}_{dc_{k'}}(t_{se}^-)} \delta_{\beta_{k'}^-(l)k}\,\delta_{\beta_{k'}^-(l+1)k} \geq \mathscr{R}_d(|\mathcal{A}_{c_k}(t_{se})|) - 1, \quad \forall k \in I_{ac}(t_{se}); \tag{6d}$$

$$\sum_{k \in I_{ac}(t_{se})} \sum_{j \in I_{dc_0}} \delta_{jk}(t_{se}) = \mathscr{R}_d(N_a); \tag{6e}$$

$$\delta_{jk}(t_{se}) \in \{0,1\}, \quad \forall j \in I_{dc_0},\ k \in I_{ac}(t_{se}); \tag{6f}$$

where $I_{dc_0} = \{1, 2, \ldots, \mathscr{R}_d(N_a)\}$; $\tilde{I}_{dc_{k'}}(t_{se}^-) = \{1, 2, \ldots, |\mathcal{D}_{c_{k'}}(t_{se}^-)| - 1\}$, where $\mathcal{D}_{c_{k'}}(t_{se}^-)$ is the k'^{th} swarm before the reassignment; $\boldsymbol{\delta}(t_{se}) \in \{0,1\}^{\mathscr{R}_d(N_a)N_{ac}}$ is the binary decision vector defined as $\boldsymbol{\delta}(t_{se}) = \{\delta_{jk}(t_{se}) | j \in I_{dc_0}, k \in I_{ac}(t_{se})\}$, where $\delta_{jk}(t_{se})$ is a decision variable which is equal to 1 when the defender $\mathcal{D}_j$ is assigned to the swarm $\mathcal{A}_{c_k}(t_{se})$ and 0 otherwise; and $\beta_{k'}^-(\cdot) = \beta_{k'}(t_{se}^-, \cdot)$ is the assignment of the defenders to a cluster $\mathcal{A}_{c_{k'}}(t_{se}^-)$ prior to the split event. The constraints (6b) ensure that each defender is assigned to exactly one swarm of the attackers. The capacity constraints (6c) ensure that for all $k \in I_{ac}(t_{se})$ swarm $\mathcal{A}_{c_k}(t_{se})$ has exactly $\mathscr{R}_d(|\mathcal{A}_{c_k}(t_{se})|)$ defenders assigned to it. The quadratic constraints (6d) ensure that all the defenders assigned to swarm $\mathcal{A}_{c_k}(t_{se})$ are connected together with an underlying Open-StringNet for all $k \in I_{ac}(t_{se})$. Consider any sub-group of $\mathscr{R}_d(|\mathcal{A}_{c_k}(t_{se})|)$ defenders out of the defenders that are already connected via Open-StringNet at the time of assignment. The intuition behind the constraints (6d) is that, for these $\mathscr{R}_d(|\mathcal{A}_{c_k}(t_{se})|)$ defenders, to be connected to each other via an Open-StringNet, each defender needs to be connected to its immediate neighbors and the total number of such connections should be equal $\mathscr{R}_d(|\mathcal{A}_{c_k}(t_{se})|) - 1$. If the total number of connections is less than $\mathscr{R}_d(|\mathcal{A}_{c_k}(t_{se})|) - 1$ then it is evident that there are at least two disconnected components in the given subgroup of defenders and they do not form a single Open-StringNet, which is not desired in our case. For example, in the scenario shown in **Figure 3**, if we were to assign 5 defenders to a cluster of attackers then the only choices under this quadratic constraint would be to choose the defenders at $\boldsymbol{\xi}_l^g$ for $l \in \{1, 2, \ldots, 5\}$ or the defenders at $\boldsymbol{\xi}_l^g$ for $l \in \{6, 7, \ldots, 10\}$ and not the defenders at $\boldsymbol{\xi}_l^g$ for $l \in \{1, 2, 7, 8, 9\}$ or any such combination of defenders that violates the Open-StringNet connectivity. The constraint (6e) ensures that all the $\mathscr{R}_d(N_a)$ defenders are assigned to the attackers' swarms.

The aforementioned MIQCP can be solved using a MIP solver Gurobi (Gurobi Optimization, 2018). After solving (6), one can find the mapping $\beta_k(t,\cdot)$, for all $k \in I_{ac}(t)$, as follows:

$$\beta_k(t,l) = \beta^-_{k^*}(l_0 + l), \qquad \text{for } t \in [t_{se} + t_{comp}, \min(t^{next}_{se}, \infty)], \tag{7}$$

where $k^* = \arg\max_{k'} \sum_{l'}^{|\mathcal{D}_{c_{k'}}(t^-_{se})|} \delta_{\beta^-_{k'}(l')k}(t_{se})$ and l_0 is the smallest integer for which $\delta_{\beta^-_{k^*}(l_0+1)k}(t_{se}) = 1$; t_{comp} is the computation time to solve (6); and t^{next}_{se} is an unknown future time at which a split happens. In other words, the assignment obtained using the states at t_{se} continues to be a valid assignment until the next split event happens at some unknown time t^{next}_{se} in the future. As shown in an instance of the defender-swarm assignment in **Figure 3**, the defenders at $\boldsymbol{\xi}^g_l$ for $l \in \{1, 2, ..., 5\}$ are assigned to swarm $\mathcal{A}_{c_2}(t_{se})$ and those at $\boldsymbol{\xi}^g_l$ for $l \in \{6, 7, ..., 10\}$ are assigned to swarm $\mathcal{A}_{c_1}(t_{se})$.

4.3. Decentralized Algorithm for Defender to AttackSwarm Assignment

The MIQCP in (6) is solved centrally, i.e., a single agent has access to the information pertaining to all the agents and the MIQCP is solved by a single agent or computer and the assignment result is communicated to other agents. To make the proposed assignment approach robust toward failure, we provide a decentralized algorithm to solve the assignment problem.

When a swarm of attackers $\mathcal{A}_{c_k}$ splits into smaller swarms at $t = t_{se}$. The newly identified swarms of the attackers by the DBSCAN algorithm are assigned new indices. Namely, one of the swarm is assigned the index k, i.e., the index of the parent swarm $\mathcal{A}_{c_k}$ and the rest swarms are assigned integers greater than $N_{ac}(t_{se})$ as their indices. Let $C^{(k)}(t_{se})$ denote the indices of the clusters of the attackers that are newly formed out of the parent cluster $\mathcal{A}_{c_k}(t^-_{se})$, when the cluster $\mathcal{A}_{c_k}$ splits at $t = t_{se}$, as identified by the DBSCAN algorithm. Then, we can assign the defenders in $\mathcal{D}_{c_k}(t^-_{se})$ only to the clusters $\mathcal{A}_{c'_k}(t_{se})$, for all $k' \in C^{(k)}(t_{se})$, and not consider other clusters of attackers, that did not split, or the other defenders during the reassignment process. We do so by solving the following modified MIQCP, which is solved only for the defenders in $\mathcal{D}_{c_k}(t^-_{se})$ and the attackers in the cluster $\mathcal{A}_{c_k}(t^-_{se})$.

$$\boldsymbol{\delta}^{(k)*}(t_{se}) = \underset{\boldsymbol{\delta}^{(k)}(t_{se})}{\arg\min} \sum_{k' \in C^{(k)}(t_{se})} \sum_{j \in D_{c_k}(t^-_{se})} \left\| \mathbf{r}_{ac_{k'}}(t_{se}) - \mathbf{r}_{dj}(t_{se}) \right\| \delta_{jk'}(t_{se}) \tag{8a}$$

Subject to
$$\textstyle\sum_{k' \in C^{(k)}(t_{se})} \delta_{jk'}(t_{se}) = 1, \quad \forall j \in D_{c_k}(t^-_{se}); \tag{8b}$$
$$\textstyle\sum_{j \in D_{c_k}(t^-_{se})} \delta_{jk'} = \mathscr{R}_d(|\mathcal{A}_{c_{k'}}(t_{se})|), \quad \forall k' \in C^{(k)}(t_{se}); \tag{8c}$$
$$\textstyle\sum_{l \in \tilde{I}_{dc_k}(t^-_{se})} \delta_{\beta^-_k(l)k'} \delta_{\beta^-_k(l+1)k'} \geq \mathscr{R}_d(|\mathcal{A}_{c_{k'}}(t_{se})|) - 1, \quad \forall k' \in C^{(k)}(t_{se}); \tag{8d}$$
$$\textstyle\sum_{j \in D_{c_k}(t^-_{se})} \sum_{k' \in C^{(k)}(t_{se})} \delta_{jk'} = |\mathcal{D}_{c_k}(t^-_{se})|; \tag{8e}$$
$$\delta_{jk}(t_{se}) \in \{0,1\}, \quad \forall j \in D_{c_k}(t^-_{se}),\ k \in C^{(k)}(t_{se}); \tag{8f}$$

where $\boldsymbol{\delta}^{(k)}(t_{se}) \in \{0,1\}^{|\mathcal{D}_{c_k}(t^-_{se})||C^{(k)}(t_{se})|}$ defined as $\boldsymbol{\delta}^{(k)}(t_{se}) = [\delta_{jk'}(t_{se}) | j \in \mathcal{D}_{c_k}(t^-_{se}), k' \in C^{(k)}(t_{se})]$, $D_{c_k}(t^-_{se}) = \{\beta_k(t^-_{se}, j) | j \in \{1, 2, ..., |\mathcal{D}_{c_k}(t^-_{se})|\}\}$ is the set of indices of the defenders in $\mathcal{D}_{c_k}$ prior to the reassignment. As one can see, the dimension of the decision vector $\boldsymbol{\delta}^{(k)}(t_{se})$ is going to be smaller than that of the decision vector in (6) and hence (8) can be solved relatively quicker than (6).

Although the clustering information is acquired and tracked centrally, the problem in (8) is solved and the assignment solution is communicated to other teammates by the lead defender in $\mathcal{D}_{c_k}(t^-_{se})$, where the lead defender is identified to be the one in the middle of the Open-StringNet formation, i.e., the defender $\mathcal{D}_{\beta_k(t^-_{se}, l_i})$ where $l_i = \lfloor \frac{|\mathcal{D}_{c_k}(t^-_{se})|}{2} \rfloor$, for all k for which the $\mathcal{A}_{c_k}$ have split.

This helps the defenders find the Defender-to-AttackSwarm assignment quickly, and without having to consider all the agents in the assignment formulation, i.e., in a decentralized way.

If only one swarm of the attackers splits at a given time instant t_{se} then indeed the optimal cost obtained in (6) and (8) are same. For other cases, the cost obtained by the decentralized algorithm would be suboptimal. A detailed analysis of performance of the centralized and the decentralized algorithms will be studied in our future work.

4.4. Heuristic to Find Defender-To-AttackSwarm Assignment

Finding the optimal defender-swarm assignment by solving the MIQCPs discussed above may not be real-time implementable for a large number of agents (> 100). In this section, we develop a computationally-efficient heuristic, called hierarchical approach, to find defender-swarm assignment. A large dimensional assignment problem is split into smaller, lower-dimensional assignment problems that can be solved optimally and quickly using the MIQCP formulation discussed earlier. Algorithm 1 provides the steps to reduce the problem of size N_{ac} to smaller problems of size smaller than or equal $\underline{N}_{ac} (\leq N_{ac})$. In Algorithm 1, $\mathscr{A}$ is a data structure, at the time of assignment $t = t_{se}$, with fields that store the information of: (i) centers of the attackers' swarms $\mathbf{R}_a(t_{se}) = [\mathbf{r}_{ac_{k'}}(t_{se}) | k' \in C^{(k)}(t_{se})]$, (ii) numbers of the attackers in each newly formed swarm $\mathbf{n}_a(t_{se}) = [|\mathcal{A}_{c_{k'}}(t_{se})| | k' \in C^{(k)}(t_{se})]$, (iii) total number of attackers $N^-_a = N_{ac_k}(t^-_{se})$. Similarly, $\mathscr{D}$ is a data structure that stores the information of: (i) defenders' positions $\mathbf{R}_d(t^-_{se}) = \{\mathbf{r}_{dj}(t^-_{se}) | j \in D_{c_k}(t^-_{se})\}$, and (ii) the goal assignment $\beta^- = \beta_k(t^-_{se}, \cdot)$. `splitApproxEqual` function splits the attackers into two groups $\mathscr{A}^l$ and $\mathscr{A}^r$ of roughly equal number of attackers, and the defenders into two groups $\mathscr{D}^l$ and $\mathscr{D}^r$. The split is performed based on the angles $\psi_{k'}$ made by relative vectors $\mathbf{r}_{ac_{k'}}(t_{se}) - \mathbf{r}_{dc}(t^-_{se})$, for all $k' \in C^{(k)}(t_{se})$, with the vector $\mathbf{r}_{d\beta^-(|D_{c_k}(t^-_{se})|)}(t_{se}) - \mathbf{r}_{dc}(t^-_{se})$ where $\mathbf{r}_{dc}(t^-_{se})$ is the center of $\mathbf{R}_d(t^-_{se})$ as shown in **Figure 5**.

We first arrange these angles $\psi_{k'}$ in descending order. The first few clusters in the arranged list with roughly half the total number of attackers become the left group $\mathscr{A}^l$ and the rest become the right group $\mathscr{A}^r$. Similarly, the left group $\mathscr{D}^l$ is formed by the first $\mathscr{A}^l.N^-_a$ defenders as per the assignment β^- and the rest defenders form the right group $\mathscr{D}^r$. For example, as shown in

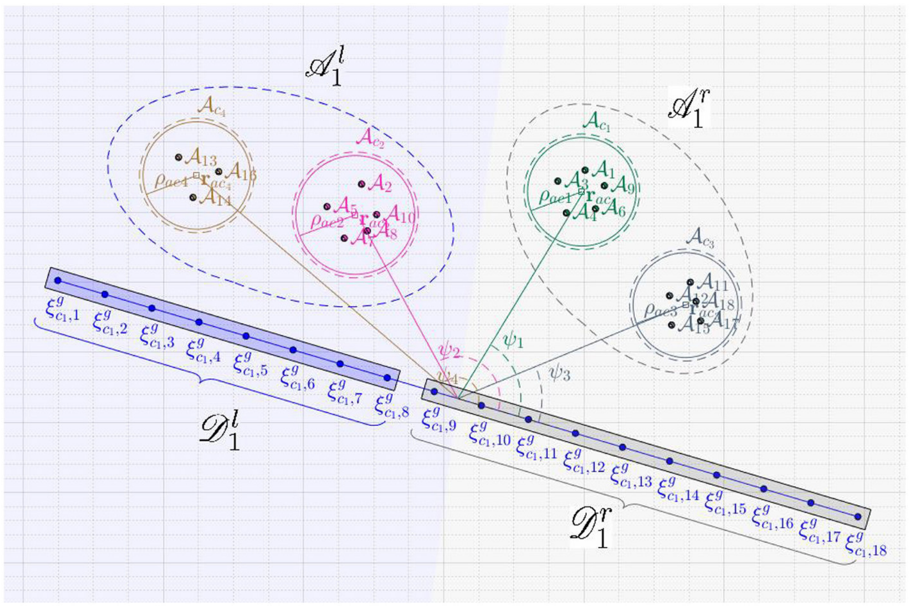

FIGURE 5 | Splitting for the hierarchical algorithm.

Algorithm 1: Defender-Swarm Assignment

```
1  Function assignHierarchical($\mathscr{A}$, $\mathscr{D}$):
2      if $\mathscr{A}.N_{ac} > \underline{N}_{ac}$ then
3          [$\mathscr{A}^l, \mathscr{D}^l, \mathscr{A}^r, \mathscr{D}^r$]=splitApproxEqual($\mathscr{A}, \mathscr{D}$);
4          if $\mathscr{A}^l.N_{ac} > \underline{N}_{ac}$ then
5              $\beta$=assignHierarchical($\mathscr{A}^l, \mathscr{D}^l$);
6          else
7              $\beta^l$ =assignMIQCP($\mathscr{A}^l, \mathscr{D}^l$);
8          if $\mathscr{A}^r.N_{ac} > \underline{N}_{ac}$ then
9              $\beta^r$ =assignHierarchical($\mathscr{A}^r, \mathscr{D}^r$);
10         else
11             $\beta^r$ =assignMIQCP($\mathscr{A}^r, \mathscr{D}^r$);
12         $\beta = \{\beta^l, \beta^r\}$;
13     else
14         $\beta$=assignMIQCP($\mathscr{A}$,$\mathscr{D}$);
15     return $\beta = \{\beta_1, \beta_2, ..., \beta_{N_{ac}}\}$
```

Figure 5, after the very first split event, the clusters $\mathcal{A}_{c_1}$, $\mathcal{A}_{c_3}$ become part of $\mathscr{A}_1^r$ and $\mathcal{A}_{c_2}$, $\mathcal{A}_{c_4}$ are a part of $\mathscr{A}_1^l$. Similarly, the defenders at $\boldsymbol{\xi}^g_{c_1,j}$ for $j \in \{1, 2, ..., 8\}$ become part of $\mathscr{D}_1^l$ while the defenders at $\boldsymbol{\xi}^g_{c_1,j}$ for $j \in \{9, 10, ..., 18\}$ become part of $\mathscr{D}_1^r$. We assign the defenders in $\mathscr{D}^l$ only to the swarms in $\mathscr{A}^l$ and those in $\mathscr{D}^r$ only to the swarms in $\mathscr{A}^r$. By doing so we may or may not obtain an assignment that minimizes the cost in (6a) but we reduce the computation time significantly and obtain a reasonably good assignment quickly. As in Algorithm 1, the process of splitting is done recursively until the number of attackers' swarms is smaller than a pre-specified number $\underline{N}_{ac}$. The function `assignMIQCP` finds the defender-swarm assignment by solving (6).

In **Figure 6**, we show the average computation time for a number of cluster configurations and random initial conditions. The computation cost first increases with the number of clusters, reaches a maximum point, and then decreases. This is because the computation cost is proportional to the number of choices available, i.e., number of ways N_{ac} groups of given sizes can be formed out of N players ($^N C_{N_{ac}}$). More importantly, as shown in **Figure 6**, the average computation time for the hierarchical approach (heuristic) to assignment is significantly smaller than that of the MIQCP formulation. We also compare the cost of MIQCP and the heuristic. **Figure 6** shows the percentage error between the cost from heuristic and the optimal cost from MIQCP, defined as % error $= \frac{100|cost_{MIQCP} - cost_{Heuristic}|}{cost_{MIQCP}}$, where $cost_{MIQCP}$ and $cost_{Heuristic}$ are the costs obtained by the MIQCP and the Heuristic, respectively. The cost of the hierarchical algorithm is very close to the optimal cost (MIQCP), see **Figure 6**. In summary, the hierarchical heuristic exploits the geometry to provide a reasonable assignment solution within a fraction of time that the optimal MIQCP formulation could have taken to find.

5. DOMINANCE REGION FOR THE DEFENDERS

In this and the following section, we provide conditions under which the defenders can successfully herd the attackers. The defenders succeed in herding the attackers if they manage to achieve the open-StringNet with line formation $\mathscr{F}^g_d$ centered at

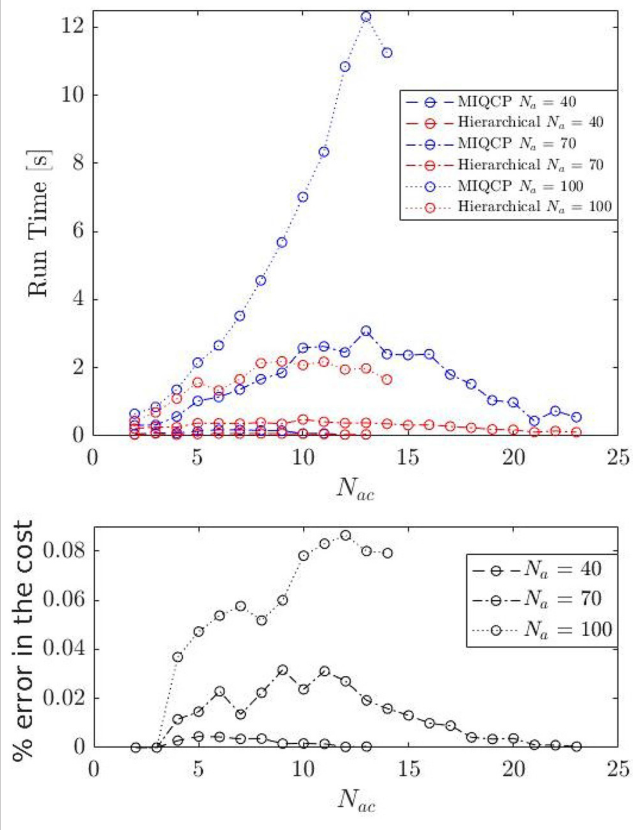

FIGURE 6 | Run-time for assignment algorithms.

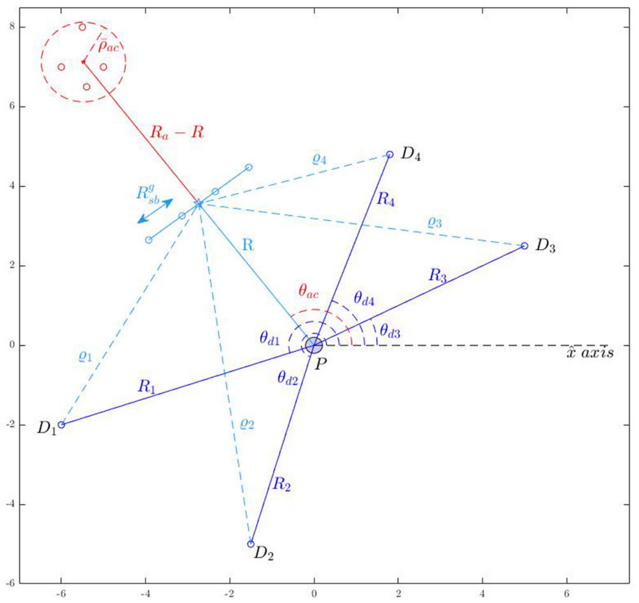

FIGURE 7 | Abstraction for estimate of dominance region.

a gathering center on the expected path of the attackers, well before the attackers reach the gathering center. If the successful gathering is possible by the defenders, then they could proceed to use the proposed StringNet herding approach to herd the attackers to safe areas, otherwise, the defenders would have to use some other approach to defense, such as direct physical capturing of the attackers.

For given initial conditions of all the agents, the defenders require to solve the problem of finding the best gathering center $\mathbf{r}_{dcg}$ and the corresponding defender-goal assignment β_o using the iterative MIQP formulation discussed in Algorithm 1 in Chipade and Panagou (2020b). One needs to check if this problem is feasible for given initial conditions to conclude whether defenders can succeed in gathering. This formulation, however, becomes computationally demanding as the number of agents becomes larger. In order to quickly decide if the successful gathering is feasible or not, we provide the following approximate problem formulation that can be solved relatively quickly.

Let $T_a(\mathbf{r}_a, \mathbf{r}, \rho_a)$ be the minimum time required by an attacker at $\mathbf{r}_a$ to reach within ρ_a distance from the point $\mathbf{r} \in \mathbb{R}^2$. Let $\mathbf{R}_d = [\mathbf{r}_{d1}, \mathbf{r}_{d2}, ..., \mathbf{r}_{dN_d}]$ denote the positions of the defenders $\mathcal{D}_j$ for all $j \in I_d$. Let $T_d(\mathbf{R}_d, \mathscr{F}^g_d(\mathbf{r}, \theta))$ be the maximum time required by all the defenders to achieve the gathering formation $\mathscr{F}^g_d(\mathbf{r}, \theta)$ centered at $\mathbf{r}$ with orientation vector making an angle θ with x-axis.

Consider N_d defenders and N_a attackers located at given positions as shown in **Figure 7**. Consider the protected area located at the origin ($\mathbf{r}_p = [0, 0]^T$). Let the largest radius of the attackers' formation be $\bar{\rho}_{ac}$. Let the position vector of the center of mass of the attackers make an angle θ_{ac} with x-axis. Let the center of the desired formation be located at a distance R from the protected area along the direction θ_{ac}. The distance of the defender $\mathcal{D}_j$ from the center of the desired formation is:

$$\varrho_j = \sqrt{R^2 + R_j^2 - 2RR_j \cos(|\theta_{dj} - \theta_{ac}|)}, \qquad (9)$$

for all $j \in I_d$. The maximum value among ϱ_j, for all $j \in I_d$, can be bounded as: $\bar{\varrho} = \max_{j \in I_d} \varrho_j \le \tilde{\varrho}_\delta = \left(\sum_{j \in I_d} \varrho_j^\delta\right)^{\frac{1}{\delta}}$ (Stipanović et al., 2012). For which we also have $\lim_{\delta \to \infty} \tilde{\varrho}_\delta = \bar{\varrho}$.

The maximum distance any defender would have to travel in the best defender-goal assignment can be upper bounded by $\bar{\varrho}_d = \tilde{\varrho}_\delta + 0.5(N_d - 1)R^g_{sb}$. The maximum time for any defender to reach the gathering location assigned to it as per the best defender-goal assignment under time-optimal control (Chipade and Panagou, 2020a) can be bounded from above by:

$$\bar{T}_d = \tau(\bar{\varrho}_d, 0, 0) = \frac{1}{\lambda_0}\left(\tanh^{-1}\left(\frac{v_{sw}}{\bar{v}_d}\right) + \tan^{-1}\left(\frac{v_{sw}}{\bar{v}_d}\right)\right) \qquad (10)$$

where $\lambda_0 = \sqrt{\bar{u}_d C_D}$, $v_{sw} = \sqrt{\frac{(\lambda-1)\bar{u}_d}{(\lambda+1)C_D}}$, $\lambda = e^{2C_D\bar{\varrho}_d}$. Similarly, the minimum time that the attackers require to reach the gathering location is when the attackers move toward the protected with the maximum possible speed. Then, the difference between the time needed for the attackers to reach the gathering location and the time required by the defenders to reach there can be bounded from below by:

$$\Delta T = \frac{R_{ac} - \bar{\rho}_{ac} - R}{\bar{v}_a} - \bar{T}_d(R) \quad (11)$$

where R_{ac} is the distance of the center of mass of the attackers from the center of the protected area and $\bar{\rho}_{ac}$ is the maximum radius of the attackers' formation under consideration. Defenders want $\Delta T \geq \Delta T_d^g$ to be able to gather well before the attackers reach the gathering center. Here ΔT_d^g is a user-defined time to account for the size of the attackers' swarm and the time required by the defenders to get connected by string barriers once arrived at the desired gathering formation. Given initial states of the attackers, one can find ΔT using (11) to assess, at least conservatively, whether the defenders can gather in the attackers' path before the attackers, without solving the actual, computationally heavy iterative MIQP formulation (Chipade and Panagou, 2020b).

Furthermore, using the above approximate analysis, for given initial conditions of the defenders, we characterize sufficient conditions on the initial positions of the attackers for which the defenders successfully gather on the shortest path of the attackers to the protected area, before the attackers can reach there. We call this set of initial conditions of the attackers as the dominance region for the given initial positions of the defenders. The dominance region is formally defined as:

Definition 6 (Defenders' Dominance Region)**.** *$Dom(\mathbf{R}_d, \bar{\rho}_{ac}, \Delta T_d^g) = \{\mathbf{r} \in \mathbb{R}^2 | \exists \upsilon \in (\frac{\rho_p}{\|\mathbf{r}\|}, 1 - \frac{\bar{\rho}_{ac}}{\|\mathbf{r}\|})$ such that $T_a(\mathbf{r}, \mathbf{r}_{dc^g}, \bar{\rho}_{ac}) - T_d(\mathbf{R}_d, \mathscr{F}_d^g(\mathbf{r}_{dc^g}, \theta_{dc^g})) \geq \Delta T_d^g$ where $\mathbf{r}_{dc^g} = \upsilon \mathbf{r}\}$.*

We use (11) to find an estimate Dom_{est} of the dominance region Dom that is completely contained inside Dom. We are interested in the limiting condition when $\Delta T = \Delta T_d^g$, that corresponds to the boundary of Dom_{est}, for which we have:

$$R_{ac} = f(R) = \bar{\rho}_{ac} + R + \bar{v}_a(\bar{T}_d(R) + \Delta T_d^g). \quad (12)$$

We want to find the smallest value $\underline{R}_{ac}(> \rho_p)$ of R_{ac} for which $\Delta T = \Delta T_d^g$, i.e.,

$$\underline{R}_{ac} = \min_{R > \rho_p} f(R). \quad (13)$$

Lemma 2. *Given that no two defenders are co-located, i.e., $\|\mathbf{r}_{dj} - \mathbf{r}_{dj'}\| > 0$ for all $j \neq j' \in I_d$, $f(R)$ as given in Equation (12) is a locally convex function of R.*

Proof: The proof is provided in the **Appendix**. □

One can find $\underline{R}_{ac}$ by solving the convex optimization (13) with $R = R^*$, the minimizer of $\tilde{\varrho}_\delta(R)$, as an initial guess to a gradient descent algorithm with sufficiently small step size. For different directions from which the attackers can approach the protected area, we solve the convex optimization (13) to find the corresponding point on the boundary Dom_{est}. **Figure 8** shows the boundaries ∂Dom_{est} and ∂Dom of the estimate Dom_{est} and the dominance region Dom, respectively. Here ∂Dom is obtained by numerically evaluating the iterative MIQP for each direction. The regions outside of the closed boundaries ∂Dom_{est} and ∂Dom are, respectively, Dom_{est} and Dom, computed for the case where the defenders are at given locations (blue circles). On the other hand, the set inside the boundaries ∂Dom_{est} and ∂Dom are the complement sets $Dom_{est}^c = \mathbb{R}^2 \backslash Dom_{est}$ and $Dom^c = \mathbb{R}^2 \backslash Dom$, respectively. The set Dom^c is essentially the dominance region of the attackers, i.e., the attackers can reach the protected area before the defenders can gather on their path if the attackers start inside Dom^c. Note that the estimate Dom_{est} is completely contained in the dominance region Dom. The region Dom is larger on the side where the density of the defenders is larger. This is intuitive because many defenders have to travel less when the attackers approach from this side and hence allow defenders to gather on the expected path of the attackers in time even if the attackers start more closer to the protected area on this side. We have the following result based on the above analysis.

Theorem 3. *Consider a group of defenders $\mathcal{D}_c = \{\mathcal{D}_1, \mathcal{D}_2, ...\mathcal{D}_{N_{dc}}\}$ starting at given locations $\mathbf{R}_{dc} = [\mathbf{r}_{d1}, \mathbf{r}_{d2}, ..., \mathbf{r}_{dN_d}]$ and a swarm of Attackers $\mathcal{A}_c$ with footprint of maximum radius $\bar{\rho}_{ac}$. If the attackers start inside $Dom_{est}(\mathbf{R}_{dc}, \bar{\rho}_{ac}, \Delta T_d^g)$, then the defenders in $\mathcal{D}_c$ are guaranteed to achieve a formation $\mathscr{F}_d^g(\mathbf{r}_{dc^g}, \theta_{dc^g})$ on the shortest path from the center of mass of the attackers in $\mathcal{A}_c$ to the protected area $\mathcal{P}$ before the attackers could reach there.*

Proof: By construction, $Dom_{est}(\mathbf{R}_{dc}, \bar{\rho}_{ac}, \Delta T_d^g) \subseteq Dom(\mathbf{R}_{dc}, \bar{\rho}_{ac}, \Delta T_d^g)$. The proof follows from the definition of the dominance region $Dom(\mathbf{R}_{dc}, \bar{\rho}_{ac}, \Delta T_d^g)$. □

In other words, Theorem 3 states that for the attackers starting in $Dom_{est}(\mathbf{R}_d, \bar{\rho}_{ac}, \Delta T_d^g)$, the defenders are guaranteed to gather in their shortest path to the protected area in time. However, if the attackers do not start in $Dom_{est}(\mathbf{R}_d, \bar{\rho}_{ac}, \Delta T_d^g)$ nothing can be concretely said about the gathering of the defenders based on the above approximate analysis.

6. RESULTS

Based on the DBSCAN clustering based assignment algorithm discussed earlier and the conditions on the initial states of the agents for successful gathering, we have the following result on the successful herding of the attackers.

Theorem 4. *Consider a group of defenders $\mathcal{D}_c = \{\mathcal{D}_1, \mathcal{D}_2, ...\mathcal{D}_{\mathscr{R}_d(N_a)}\}$ starting at given locations $\mathbf{R}_{dc} = [\mathbf{r}_{d1}, \mathbf{r}_{d2}, ..., \mathbf{r}_{d\mathscr{R}_d(N_a)}]$. Let attackers' initial positions be such that $\rho_{ac}(0) = \max_{i \in I_a} \|\mathbf{r}_{ai}(0) - \mathbf{r}_{ac}(0)\| \leq \bar{\rho}_{ac}$, where $\mathbf{r}_{ac}(0) = \sum_{i \in I_a} \frac{\mathbf{r}_{ai}(0)}{N_a}$, and that they belong to $Dom_{est}(\mathbf{R}_{dc}, \bar{\rho}_{ac})$. Furthermore, suppose that for all times $t > 0$, each attacker belongs to one of the clusters $\{\mathcal{A}_{c_1}, \mathcal{A}_{c_2}, ..., \mathcal{A}_{c_{N_{ac}}}\}$, for some N_{ac}, that are identified by DBSCAN algorithm with $\varepsilon_{nb} = \frac{\bar{\rho}_{ac}(m_{pts}-1)}{N_a - 1}$ and $m_{pts} = 3$ whenever a swarm splits into smaller swarms. Then,*

(i) the defenders in $\mathcal{D}_c$ are guaranteed to enclose every attacker inside a StringNet; and

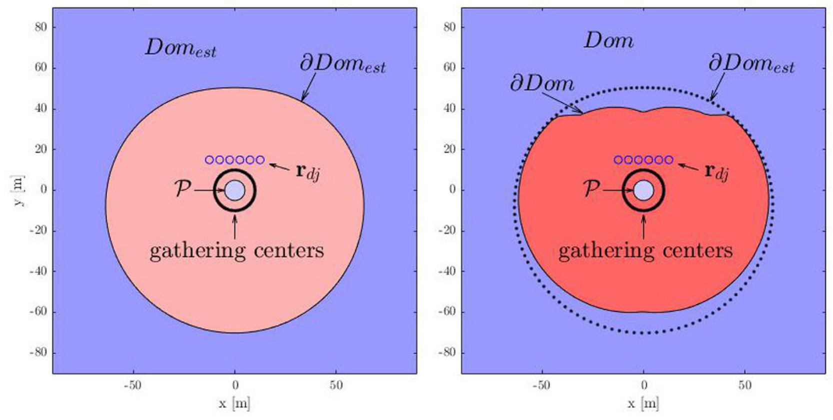

FIGURE 8 | Dominance regions of the players.

(ii) if a swarm of attackers has not reached the protected area $\mathcal{P}$ by the time it is enclosed within a StringNet then the swarm is guaranteed to be herded to one of the safe areas.

Proof: (i) The attackers start inside $Dom_{est}(\mathbf{R}_{dc}, \bar{\rho}_{ac}) \subset Dom(\mathbf{R}_{dc}, \bar{\rho}_{ac})$, so as per Theorem 3 the defenders are able to gather and get connected by string barriers on the shortest path of the attackers to the protected area. Since each attacker belongs to one of the swarms of attackers $\{\mathcal{A}_{c_1}, \mathcal{A}_{c_2}, ..., \mathcal{A}_{c_{N_{ac}}}\}$ that is identified by DBSCAN algorithm, we have from the Lemma 1 that each cluster satisfies $\rho_{ac_k} = \max_{i \in I_{ac_k}} \|\mathbf{r}_{ai} - \mathbf{r}_{ac_k}\| \leq \frac{\bar{R}_{sb}}{2} \cot\left(\frac{\pi}{\mathscr{R}_d(|\mathcal{A}_{c_k}|)}\right)$, as long as $|\mathcal{A}_{c_k}| > 3$. This implies that a group of $\mathscr{R}_d(|\mathcal{A}_{c_k}|)$ defenders $\mathcal{D}_{c_k}$ connected via StringNet are capable of enclosing the attackers in $\mathcal{A}_{c_k}$ for all $k \in I_{ac}$. Under the control laws as described in section 3 (and in Chipade and Panagou (2020b)), it is proved in Theorem 5 and 6 in Chipade and Panagou (2020b) that the defenders in $\mathcal{D}_{c_k}$ form a closed-StringNet around the attackers in $\mathcal{A}_{c_k}$, i.e., the defenders enclose the attackers, for all $k \in I_{ac}$.

(ii) Once the attackers in a cluster $\mathcal{A}_{c_k}$ are enclosed by the defenders in $\mathcal{D}_{c_k}$ and it is true that the attackers have not reached the protected area, then the defenders in $\mathcal{D}_{c_k}$, under the control actions described in section 3.4 (and in Chipade and Panagou 2020b), herd the attackers in $\mathcal{A}_{c_k}$ to one of the safe areas as proved in Theorem 7 in Chipade and Panagou (2020b), for all $k \in I_{ac}$.

$\square$

Remark 1. *We do provide strong guarantees on the completion of the gathering phase. We also proved in Chipade and Panagou (2020b) that the attackers' clusters will be enclosed within some finite time under the state-feedback control laws as discussed in Chipade and Panagou (2020b). However, finding upper bounds on this finite time with many agents and formation interacting with each other is not a trivial task and hence providing strong conditions under which seeking and enclosing phases are also completed well before attackers reach the protected areas for each cluster of attackers is left open for future research.*

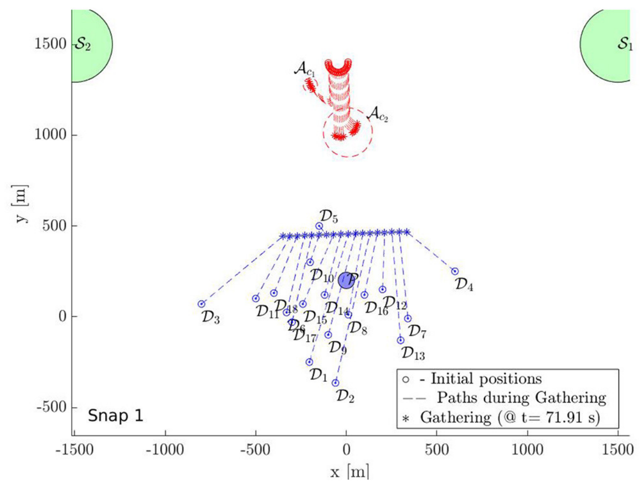

FIGURE 9 | Multi-Swarm StringNet Herding: Snapshot 1.

7. SIMULATIONS

In this section, we provide a simulation of 18 defenders herding 18 attackers to $\mathcal{S}$ with bounded control inputs. **Figures 9–12** show the snapshots of the paths taken by all agents. The positions and paths of the defenders are shown in blue color, and that of the attackers in red. The string-barriers between the defenders are shown as wide solid blue lines with white dashes in them.

Snapshot 1 shows the paths during the gathering phase. As observed the defenders are able to gather at a location on the shortest path of the attackers to the protected area before the attacker reach there. Five attackers are already separated from the rest 13 in reaction to the incoming defenders in their path. The defenders have identified two swarms of the attackers $\mathcal{A}_{c_1}$ and

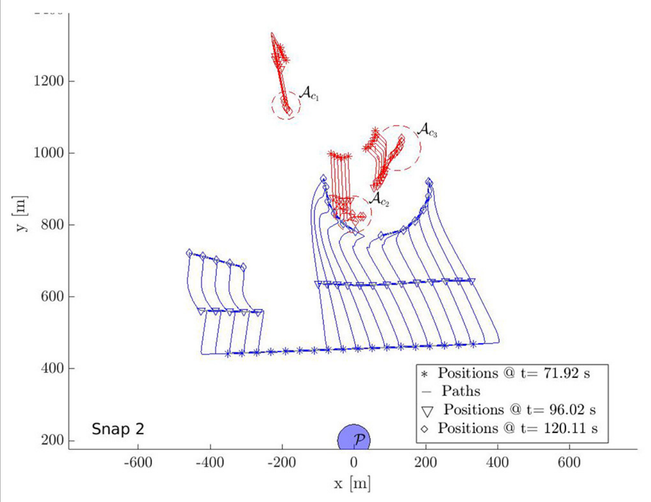

FIGURE 10 | Multi-Swarm StringNet Herding: Snapshot 2.

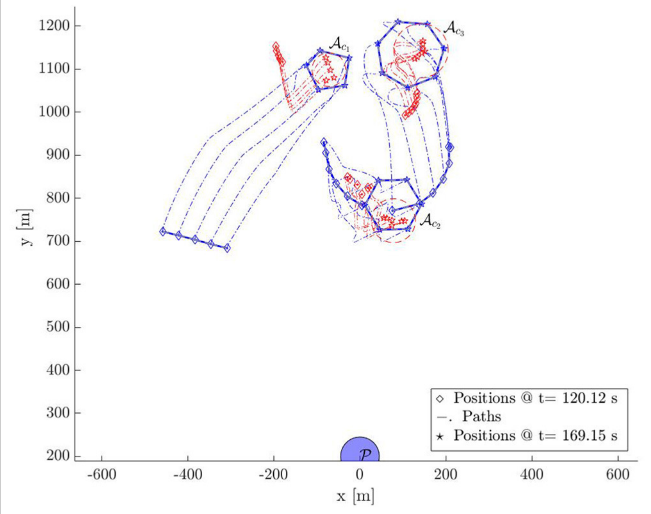

FIGURE 11 | Multi-Swarm StringNet Herding: Snapshot 3.

$\mathcal{A}_{c_2}$ at the end of the gathering phase and assign two subgroups $\mathcal{D}_{c_1}$ and $\mathcal{D}_{c_2}$ of the defenders to $\mathcal{A}_{c_1}$ and $\mathcal{A}_{c_2}$ using Algorithm 1. As shown in snapshot 2, $\mathcal{D}_{c_1}$ and $\mathcal{D}_{c_2}$ seek $\mathcal{A}_{c_1}$ and $\mathcal{A}_{c_2}$, but the attackers in swarm $\mathcal{A}_{c_2}$ further start splitting and the defenders identify this newly formed $\mathcal{A}_{c_2}$ and $\mathcal{A}_{c_3}$ at time $t = 120.11$ s. The group $\mathcal{D}_{c_2}$ is then split into two subgroups $\mathcal{D}_{c_2}$ and $\mathcal{D}_{c_3}$ of appropriate sizes and assigned to the new swarms $\mathcal{A}_{c_2}$ and $\mathcal{A}_{c_3}$ using Algorithm 1.

Snapshot 3 shows how the 3 subgroups of the defenders are able to enclose the identified 3 swarms of the attackers by forming Closed-StringNets around them. Snapshot 4 shows how all the three enclosed swarms of the attackers are taken to the respective closest safe areas while each defenders' group ensures collision avoidance from other defenders' groups. Additional simulations can be found at https://tinyurl.com/yypb2yv9.

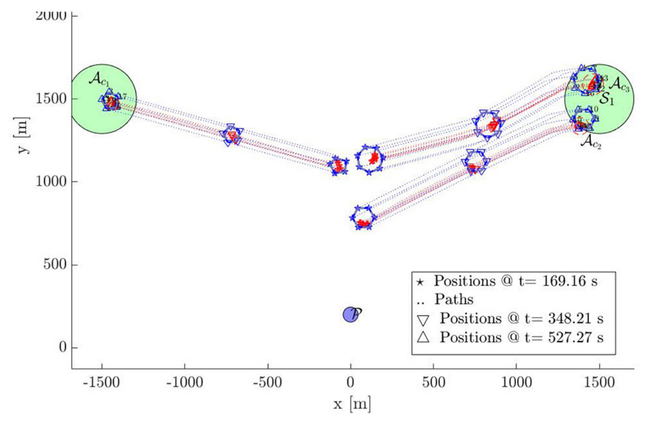

FIGURE 12 | Multi-Swarm StringNet Herding: Snapshot 4.

8. EXPERIMENTAL RESULTS

In this section, we provide hardware demonstrations of the herding approach. For this purpose, we use a fleet of in-house built 4 quadrotors each of which uses PixHawk cube 2.1 autopilot board for autonomous control. Each quadrotor is also fitted with Real-Time Kinetic (RTK) supported *here+* GPS module and a ESP8266 Wi-Fi module. **Figure 13** shows the overall experimental setup used for the demonstrations. The experimental setup consists of the fleet of quadrotors, a ground station computer, a wifi router, a RTK GPS base module. The ground station computer and the quadrotors are connected to a common wifi network created by the wifi router. The wifi modules on the quadrotors are used for the communication between the quadrotors and the ground station computer. The RTK GPS base module is used to provide corrections to the on-board GPS modules to provide centimeter level position accuracy. We use robot operating system (ROS) as an underlying framework to exchange and manipulate different signals used across the system. In particular, we use MAVROS package, based on MAVLink communication protocol, to exchange information between the ground station and the quadrotors.

As a proof of concept, we only demonstrate the "StringNet Herding" for single attacking swarm case in a centralized setting. In this setup, the ground station receives position and velocity commands from the vehicles and sends next reference commands, obtained through the MATLAB simulation running in the background based on the StringNet Herding formulation discussed earlier, to the quadrotors. The decentralized version for multi-swarm case can be tested similarly by having sufficient computational power available on the quadrotors.

We perform our experiments in an outdoor netted facility named M-Air at the University of Michigan. M-Air is a cuboid shaped netted facility as shown in **Figure 13**. We consider a scenario with 3 defenders ($\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$) and 1 attacker ($\mathcal{A}_1$) to demonstrate the proposed herding approach due to the limited space available in M-Air. The location of the protected area, the

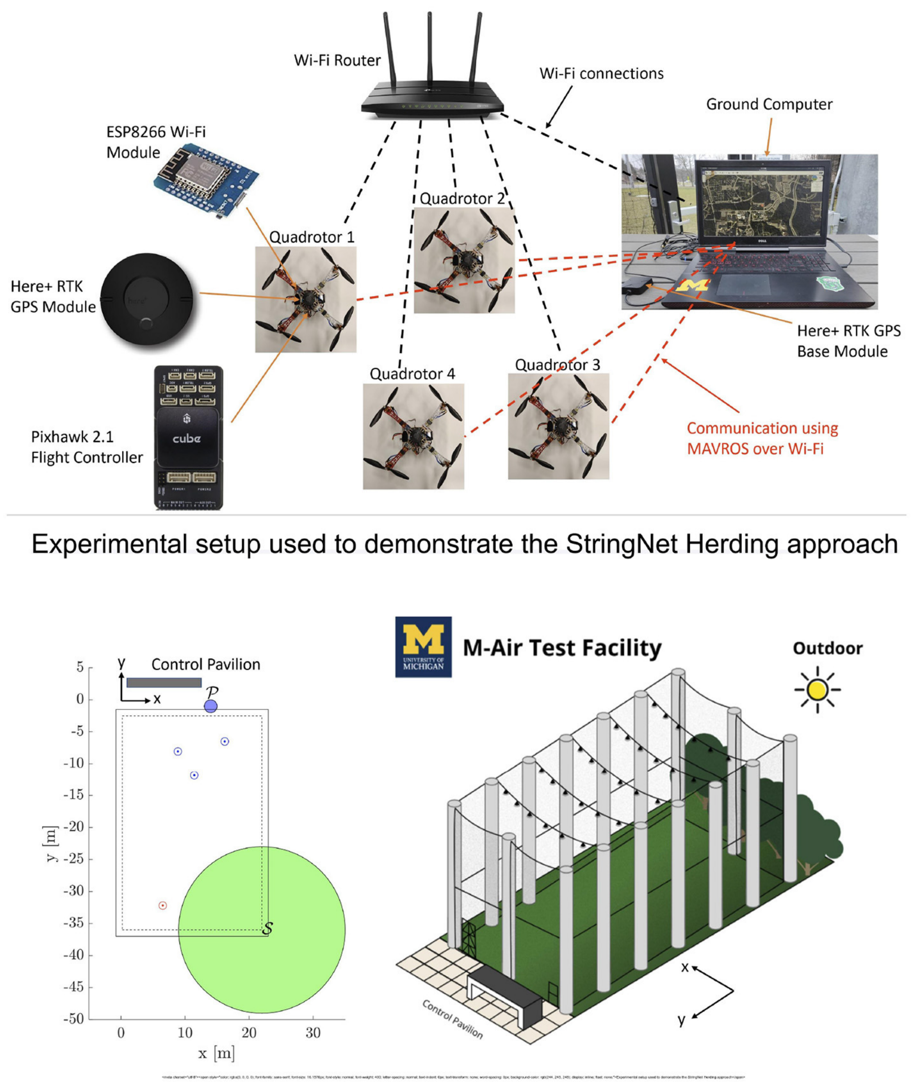

FIGURE 13 | Experimental setup, M-Air facility and the initial configuration (The dotted line in the figure on the left shows the safe region of operation inside M-Air facility).

safe area and the initial locations of the quadrotors in M-Air are shown in **Figure 13**. We chose the protected area outside the M-Air so that we have large area available for quadrotors' motion. The safe area is chosen to be centered at one corner of M-Air with radius of 13 m so that the entire formation of the defenders after enclosing the attacker is able to reach inside the safe area.

Again due to limited space, we only demonstrate the enclosing and the herding phases of the StringNet Herding approach. During the experiment all the quadrotors are commanded to fly at an altitude of 2.5 m above the local ground. The paths traversed by the quadrotors starting at the initial positions as shown in **Figure 13** during the experiment are shown in **Figure 14**. The visuals of the quadrotors at different time instances during the experiment are shown in **Figure 15**. In **Figure 15**, the defenders $\mathcal{D}_1$, $\mathcal{D}_2$, and $\mathcal{D}_3$ are denoted by D1, D2, and D3, respectively, with blue oval drawn around them to highlight where they are located in the figure. Similarly, the attacker is denoted by A1 and a red oval is drawn around it to highlight it.

As one can observe in **Figure 14** the attacker starts moving toward the protected area in the beginning. Once it detects the defenders on its path, it starts to move away from them in order to protect itself. But, the defenders are able to enclose the attacker successfully at around $t = 77$ s despite attacker's initial attempt at escaping from them, see **Figure 15** for the visual of the quadrotors at this instance. After the attacker is enclosed it is herded to the safe area located at the corner of M-Air at $t = 149$ s as shown in **Figures 14**, **15** and thus the protected area is protected from the attack by the attacker. The video of the experiment can be found at https://tinyurl.com/yyd3qfty.

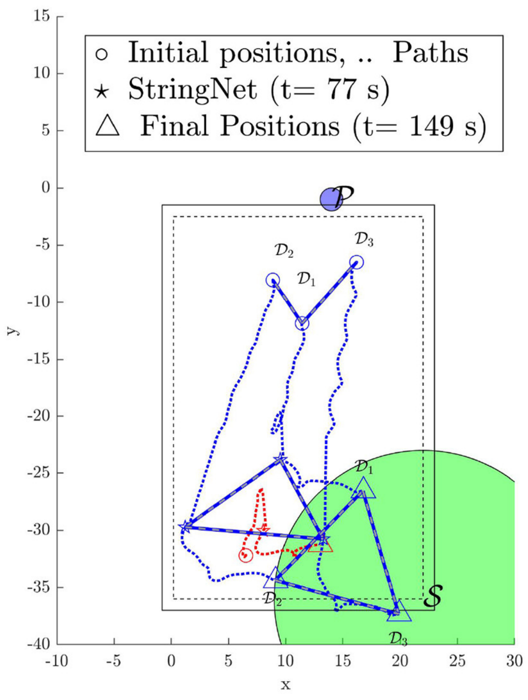

FIGURE 14 | Paths traversed by the quadrotors during StringNet Herding (The dotted blue lines denote the paths of the defenders while the red denote that of the attackers. The solid blue lines with white dashes in them denote the string barrier assumed between the quadrotors).

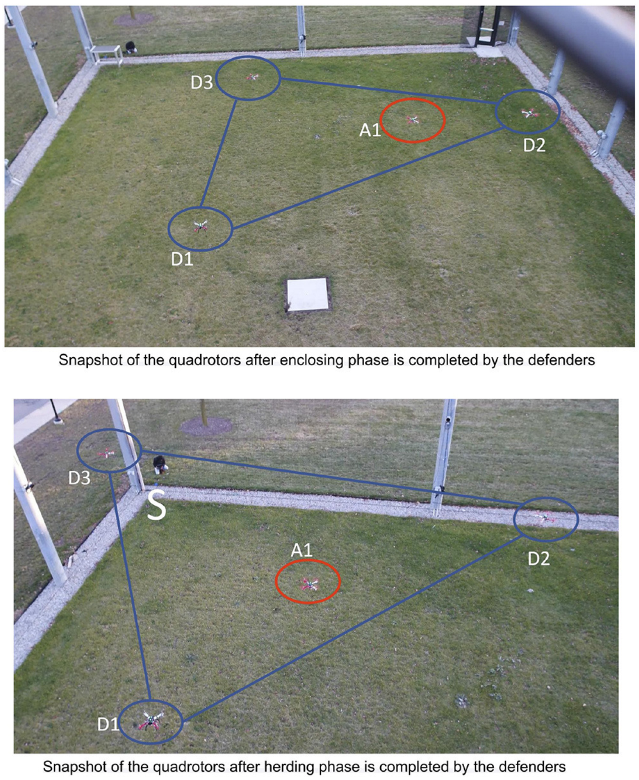
D3
A1
D2
D1
Snapshot of the quadrotors after enclosing phase is completed by the defenders
D3
S
A1
D2
D1
Snapshot of the quadrotors after herding phase is completed by the defenders

FIGURE 15 | Snapshot of the quadrotors after enclosing and herding phases are completed by the defenders.

9. OUR THOUGHTS ON THREE DIMENSIONAL (3D) CASE AND NON-CIRCULAR GEOMETRIES

The idea of StringNet can also be applied to 3D case. In our recent work Zhang et al. (2020), we extended the idea of "StringNet Herding" to the 3D case. The StringNet in 3D case, 3D-StringNet, is a single component, orientable triangle mesh with zero genus (holes) made of triangular net-like barrier faces. Similar to 2D-Stringnet herding, 3D-StringNet herding also consists of four phases: (1) gathering, (2) seeking, (3) enclosing and (4) herding. In Zhang et al. (2020), we design three 3D-StringNet formations of the defenders namely planar, hemispherical, spherical that are required to be achieved in the phases discussed above in order to effectively enclose the attackers and herd them to a safe area.

Although we assumed that the areas and the agents are circular, the proposed algorithm can be easily extended to non-circular geometries by considering appropriate distance metric in the algorithmic formulation.

10. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a clustering-based, connectivity-constrained, assignment algorithm that distributes and assigns groups of defenders against swarms of the attackers, to herd them to the closest safe area using "StringNet Herding" approach. We provide two algorithms to solve this assignment problem: centralized and decentralized, and also a heuristic based on the optimal MIQCP that finds this assignment quickly. Furthermore, we provide conditions under which the defenders can successfully herd the attackers to safe areas.

Simulations show how this proposed multi-swarm herding method improves the original "StringNet Herding" method and enables the defenders herd all the attackers to safe areas even though the attackers start splitting into smaller swarms in reaction to the defenders. Hardware experiments demonstrate the success of the approach in real applications.

In our future work, we want to study a defense approach that combines herding and interception approach together in order to defend against wide range of attacks by the attackers.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

AUTHOR CONTRIBUTIONS

VC and DP contributed to the conception and design of the study. VC formulated and simulated the theoretical algorithms and simulation case studies. VC and VM contributed to the development of experimental platform and experimental demonstrations of the herding algorithm. VC wrote major sections of the manuscript. DP is the principal investigator associated with this project. All authors contributed to manuscript revision, read, and approved the submitted version.

FUNDING

This work has been funded by the Center for Unmanned Aircraft Systems (C-UAS), a National Science Foundation Industry/University Cooperative Research Center (I/UCRC) under NSF Award No. 1738714 along with significant contributions from C-UAS industry members.

REFERENCES

Ankerst, M., Breunig, M. M., Kriegel, H.-P., and Sander, J. (1999). Optics: ordering points to identify the clustering structure. *ACM Sigmod Rec.* 28, 49–60. doi: 10.1145/304181.304187

Boyd, S., and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge: Cambridge University Press.

Burkard, R., Dell'Amico, M., and Martello, S. (2012). *Assignment Problems, Revised Reprint*, Vol. 106. Philadelphia, PA: Siam.

Cai, N., Diao, C., and Khan, M. J. (2017). A novel clustering method based on quasi-consensus motions of dynamical multiagent systems. *Complexity* 2017:4978613. doi: 10.1155/2017/4978613

Chen, M., Zhou, Z., and Tomlin, C. J. (2017). Multiplayer reach-avoid games via pairwise outcomes. *IEEE Trans. Autom. Control* 62, 1451–1457. doi: 10.1109/TAC.2016.2577619

Chipade, V. S., and Panagou, D. (2019). "Herding an adversarial swarm in an obstacle environment," in *2019 IEEE 58th Conference on Decision and Control (CDC)* (Nice: IEEE), 3685–3690.

Chipade, V. S., and Panagou, D. (2020a). Approximate time-optimal trajectories for damped double integrator in 2d obstacle environments under bounded inputs. *arXiv [Preprints] arXiv:*2007.

Chipade, V. S., and Panagou, D. (2020b). Multi-agent planning and control for swarm herding in 2d obstacle environments under bounded inputs. *IEEE Trans. Robot.* Available online at: https://tinyurl.com/yy5k6943

Chipade, V. S., and Panagou, D. (2020c). "Multi-swarm herding: Protecting against adversarial swarms," in *2020 59th IEEE Conference on Decision and Control (CDC)* (IEEE), 5374–5379. doi: 10.1109/CDC42340.2020.9303837

Coon, M., and Panagou, D. (2017). "Control strategies for multiplayer target-attacker-defender differential games with double integrator dynamics," in *Conference on Decision and Control* (Melbourne, VIC: IEEE), 1496–1502.

Dai, B., and Li, W. (2014). "Flocking of -agents with arbitrary shape obstacle," in *Proceedings of the 33rd Chinese Control Conference* (Nanjing: IEEE), 1311–1316.

Deptula, P., Bell, Z. I., Zegers, F. M., Licitra, R. A., and Dixon, W. E. (2018). "Single agent indirect herding via approximate dynamic programming," in *2018 IEEE Conference on Decision and Control (CDC)* (Miami Beach, FL: IEEE), 7136–7141.

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Kdd* (Portland, OR), Vol. 96-34, 226–231.

Goel, R., Lewis, J., Goodrich, M., and Sujit, P. (2019). "Leader and16 predator based swarm steering for multiple tasks," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)* (Bari: IEEE), 3791–3798.

Gurobi Optimization, L. (2018). *Gurobi Optimizer Reference Manual.*

Haque, M. A., Rahmani, A. R., and Egerstedt, M. B. (2011). Biologically inspired confinement of multi-robot systems. *Int. J. Bio Inspir. Comput.* 3, 213–224. doi: 10.1504/IJBIC.2011.041145

Kline, A., Ahner, D., and Hill, R. (2019). The weapon-target assignment problem. *Comput. Operat. Res.* 105, 226–236. doi: 10.1016/j.cor.2018.10.015

Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval Res. Log. Q.* 2, 83–97. doi: 10.1002/nav.3800020109

Licitra, R. A., Bell, Z. I., Doucette, E. A., and Dixon, W. E. (2018). Single agent indirect herding of multiple targets: a switched adaptive control approach. *IEEE Control Syst. Lett.* 2, 127–132. doi: 10.1109/LCSYS.2017.2763968

Licitra, R. A., Hutcheson, Z. D., Doucette, E. A., and Dixon, W. E. (2017). Single agent herding of n-agents: a switched systems approach. *IFAC PapersOnLine* 50, 14374–14379. doi: 10.1016/j.ifacol.2017.08.2020

MacQueen, J., Le Cam, L. M., and Neyman, J. (1967). "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1–14 (Oakland, CA), 281–297.

Mirjan, A., Federico, A., Raffaello, D., Fabio, G., and Matthias, K. (2016). "Building a bridge with flying robots," in *Robotic Fabrication in Architecture, Art and Design 2016* (Cham: Springer), 34–47.

Nardi, S., Mazzitelli, F., and Pallottino, L. (2018). A game theoretic robotic team coordination protocol for intruder herding. *IEEE Robot. Autom. Lett.* 3, 4124–4131. doi: 10.1109/LRA.2018.2857004

O'Callaghan, L., Mishra, N., Meyerson, A., Guha, S., and Motwani, R. (2002). "Streaming-data algorithms for high-quality clustering," in *Proceedings 18th International Conference on Data Engineering* (San Jose, CA: IEEE), 685–694.

Öncan, T. (2007). A survey of the generalized assignment problem and its applications. *Inform. Syst. Oper. Res.* 45, 123–141. doi: 10.3138/infor.45.3.123

Paranjape, A. A., Chung, S.-J., Kim, K., and Shim, D. H. (2018). Robotic herding of a flock of birds using an unmanned aerial vehicle. *IEEE Trans. Robot.* 34, 901–915. doi: 10.1109/TRO.2018.2853610

Pierson, A., and Schwager, M. (2018). Controlling noncooperative herds with robotic herders. *IEEE Trans. Robot.* 34, 517–525. doi: 10.1109/TRO.2017.2776308

Raghuwaiya, K., Vanualailai, J., and Sharma, B. (2016). "Formation splitting and merging," in *International Conference on Swarm Intelligence* (Bali: Springer), 461–469.

Rezende, M. D., De Lima, B. S. P., and Guimarães, S. (2018). A greedy ant colony system for defensive resource assignment problems. *Appl. Artif. Intell.* 32, 138–152. doi: 10.1080/08839514.2018.1451137

Sharan, R., and Shamir, R. (2000). Click: a clustering algorithm with applications to gene expression analysis. *Proc. Int. Conf. Intell. Syst. Mol. Biol.* 8, 307–316.

Shishika, D., Paulos, J., and Kumar, V. (2020). Cooperative team strategies for multi-player perimeter-defense games. *IEEE Robot. Autom. Lett.* 5, 2738–2745. doi: 10.1109/LRA.2020.2972818

Stipanović, D. M., Tomlin, C. J., and Leitmann, G. (2012). Monotone approximations of minimum and maximum functions and multi-objective problems. *Appl. Math. Optimiz.* 66, 455–473. doi: 10.1007/s00245-012-9179-8

Varava, A., Hang, K., Kragic, D., and Pokorny, F. T. (2017). "Herding by caging: a topological approach towards guiding moving agents via mobile robots," in *Proceedings of Robotics: Science and Systems* (Cambridge, MA). doi: 10.15607/RSS.2017.XIII.074

Xu, D., and Tian, Y. (2015). A comprehensive survey of clustering algorithms. *Ann. Data Sci.* 2, 165–193. doi: 10.1007/s40745-015-0040-1

Yan, R., Shi, Z., and Zhong, Y. (2019). Task assignment for multiplayer reach–avoid games in convex domains via analytical barriers. *IEEE Trans. Robot.* 36, 107–124. doi: 10.1109/TRO.2019.2935345

Zhang, T., Ramakrishnan, R., and Livny, M. (1996). Birch: an efficient data clustering method for very large databases. *ACM Sigmod Rec.* 25, 103–114. doi: 10.1145/235968.233324

Zhang, W., Chipade, V. S., and Panagou, D. (2020). Herding an adversarial swarm in three-dimensional spaces. *arXiv preprint arXiv:2007.04406.*

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

11. APPENDIX

11.1. Proof of Lemma 2

Sum of two convex functions is always a convex function (Boyd and Vandenberghe, 2004), so it is sufficient to show that $\bar{T}_d(R)$ is a locally convex function to show that $f(R)$ is a locally convex function. Let $g(R) = \bar{T}_d(\bar{\varrho}_d(R))$. The double derivative of g is:

$$\frac{\partial^2 g}{\partial R^2} = \frac{\partial^2 \bar{T}_d}{\partial \bar{\varrho}_d^2} \left(\frac{\partial \bar{\varrho}_d}{\partial R}\right)^2 + \frac{\partial \bar{T}_d}{\partial \bar{\varrho}_d} \frac{\partial^2 \bar{\varrho}_d}{\partial R^2}. \tag{14}$$

We have

$$\frac{\partial \bar{T}_d}{\partial \bar{\varrho}_d} = \frac{C_d}{\lambda_0}\sqrt{\frac{\lambda+1}{\lambda-1}} \geq 0; \tag{15a}$$

$$\frac{\partial^2 \bar{T}_d}{\partial \bar{\varrho}_d^2} = \frac{1}{\lambda_0}\left(\frac{(2C_d)^2\lambda}{1-\lambda^2}\sqrt{\frac{\lambda+1}{\lambda-1}}\right) \leq 0; \tag{15b}$$

$$\frac{\partial \bar{\varrho}_d}{\partial R} = \sum_{j=1}^{N_d} \varrho_j^{\delta-2} \, (\tilde{\varrho}_\delta)^{\frac{1}{\delta}-1} \, (R - R_{\theta j}); \tag{15c}$$

$$\frac{\partial^2 \bar{\varrho}_d}{\partial R^2} = \sum_{j=1}^{N_d} (\tilde{\varrho}_\delta)^{\frac{1}{\delta}-1} \, \varrho_j^{\delta-2}\Big\{\big(\tfrac{1}{\delta}-1\big)\frac{(R-R_{\theta j})}{(\tilde{\varrho}_\delta)}\frac{\partial \bar{\varrho}_d}{\partial R} \\ 1 + (\delta-2)\varrho_j^{-2}(R-R_{\theta j})^2\Big\}, \tag{15d}$$

where $R_{\theta j} = R_j cos(|\theta_{dj} - \theta_{ac}|)$. Let R^* be such that $\frac{\partial \bar{\varrho}_d}{\partial R}|_{R=R^*} = 0$. We have that ϱ_j is a convex function of R which implies that its ℓ_δ-norm, $\tilde{\varrho}_\delta$, is also a convex function (Boyd and Vandenberghe, 2004). This means $\tilde{\varrho}_\delta(R^*)$ is the minimum value of $\tilde{\varrho}_\delta$, i.e., $\tilde{\varrho}_\delta \geq \tilde{\varrho}_\delta(R^*)$. Since not all defenders are co-located $\tilde{\varrho}_\delta(R^*) > 0$ implying $\tilde{\varrho}_\delta > 0$ and $\lambda > 1$. From Equation (15d), we have $\frac{\partial^2 \bar{\varrho}_d}{\partial R^2}|_{R=R^*} > 0$. Then from Equation (14), we get $\frac{\partial^2 g}{\partial R^2}|_{R=R^*} > 0$. We know that ϱ_j is a twice continuously differentiable function of R for $R > 0$ and if we choose $\delta \geq 2$ then we can show that both $\frac{\partial \bar{\varrho}_d}{\partial R}$ and $\frac{\partial^2 \bar{\varrho}_d}{\partial R^2}$ are continuous functions of R. From Equations (15a) and (15b), we have that $\frac{\partial \bar{T}_d}{\partial \bar{\varrho}_d}$ and $\frac{\partial^2 \bar{T}_d}{\partial \bar{\varrho}_d^2}$ are continuous functions of R. This implies that $\frac{\partial^2 g}{\partial R^2}$ is continuous at $R = R^*$.

Combining the two results that $\frac{\partial^2 g}{\partial R^2}$ is continuous and greater than 0 at $R = R^*$ implies that there exists $\epsilon > 0$ such that $\frac{\partial^2 g}{\partial R^2} > 0$ for all R satisfying $|R - R^*| < \epsilon$, i.e., $g(R)$ is locally convex in the neighborhood of $R = R^*$. Hence $f(R)$ is also a locally convex function in the neighborhood of $R = R^*$.