IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 28, 2019, accepted January 8, 2020, date of publication January 23, 2020, date of current version February 5, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2969021

Enhanced Social Spider Optimization Algorithm for Increasing Performance of Multiple Pursuer Drones in Neutralizing Attacks From Multiple Evader Drones

ARIO YUDO HUSODO[1,2] (Member, IEEE), GRAFIKA JATI[1], (Member, IEEE), AMARULLA OCTAVIAN[3], AND WISNU JATMIKO[1], (Senior Member, IEEE)

[1]Department of Computer Science, University of Indonesia, Depok 16424, Indonesia
[2]Department of Informatics Engineering, Faculty of Engineering, University of Mataram, Mataram 83125, Indonesia
[3]Indonesian Naval Command and Staff College, Jakarta 12230, Indonesia

Corresponding authors: Ario Yudo Husodo (ario.yudo@ui.ac.id) and Wisnu Jatmiko (wisnuj@cs.ui.ac.id)

This work was supported in part by the Penugasan Publikasi Artikel di Jurnal Internasional Kuartil Q1Q2 through the Universitas Indonesia under Grant NKB-0211/UN2.R3.1/HKP.05.00/2019.

ABSTRACT We propose an optimization algorithm for reducing execution time needed by multiple pursuers in solving a variant of the Multiple-Pursuer Multiple-Evader (MPME) problem where each evader tries to attack an area defended by pursuers. This problem is a variant of the Multi-Agent Pursuit Evasion problem. In our discussed problem, a group of pursuers tries to defend an area from a group of evaders' attacks. The main task given in this problem is how pursuers can capture or immobilize as soon as possible any evader trying to get closer to the defended area (evaders' target). We use Social Spider Optimization (SSO) algorithm as the basis of our proposed method. In SSO, there are female spiders, dominant-male spiders, and non-dominant-male spiders collaborating to catch their prey. In SSO, there are three main procedures usually exist: calculation of fitness value, the vibrational summons of surrounding spiders, and mating procedure. In this paper, we develop an enhanced SSO algorithm where excludes the mating procedure and propose a practical calculation process for solving our discussed problem. SSO is one of the recent optimization algorithms developed in the computer science field. Developing this algorithm for solving dynamic problem like the MPME variant surely brings a novelty in the computer science research area. We test our proposed method in a 3D simulation environment where we manifest all pursuers and evaders as drones. Based on our experiment result, our algorithm performs better than commonly used methods for solving the MPME problem.

INDEX TERMS 3D-simulation, drone, multiple-evader, multiple-pursuer, social-spider-optimization.

I. INTRODUCTION

A. PROBLEM OVERVIEW

Multiple-Pursuer Multiple-Evader (MPME) problem is a variant of pathfinding problem which consists of multiple agents from varied starting points trying to reach multiple ending points, which are usually called as targets. In MPME, the position of the targets can change dynamically along with the experiment. The focus of the problem solution is not on how a single agent finds a path to a single target; however, the focus lies on how the group of agents as a whole could reach every single target as soon as possible.

There have been some researches conducted related to MPME topic, for example [1]–[3]. Unfortunately, to the best of our knowledge, researches on MPME problem tend to focus on solving ''cops and robbers'' problem [4], wherein general, evaders (robbers) tends to avoid and run away from pursuers (cops). In our discussed problem variant, the evaders are not trying to run away from pursuers, but they move systematically to attack an area without concern about their safety. They act as *kamikaze* troops, which are programmed to attack an area without considering the path to come back to

The associate editor coordinating the review of this manuscript and approving it for publication was Fabrizio Messina.

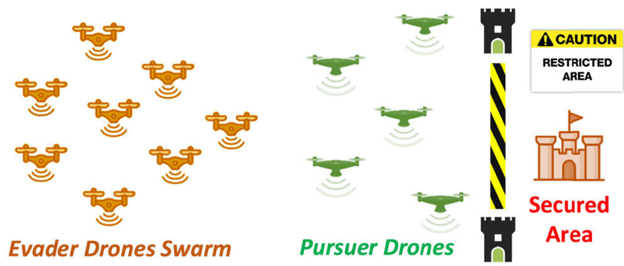

FIGURE 1. Illustration of MPME variant discussed in this research.

their origin. This topic is still being an open research issue in computer science. Technically speaking, this MPME variant has become a real practical problem that requires an effective solution. One of the real-world problems manifested from this problem that required serious attention is the Multiple Pursuer Drones (MPD) problem in Neutralizing Attacks from Multiple Evader Drones (MED). In this problem, as shown in Fig 1, there is a group of evader drones trying to attack a secured area guarded by a group of pursuer drones. Each pursuer and evader drone can move automatically without human control. MPD vs. MED is a real-world problem that needs a comprehensive research study because attacks from MED is bringing a concrete threat to society nowadays. As drone technology getting more sophisticated each day, the misuse risk of drones for attacking an area could also increase.

There are many records showings that drones have been developed as a tool in military battles. Research about the potential misused of drones can be seen on [5]. One of the most dangerous usages of drones as a military tool is its usage as a swarm drone. Swarm drone is a drone technology where several drones can coordinate and communicate with each other as a group in performing a specific task, for example, destroying an area. Some examples of current swarm drone researches can be seen on [6]–[8]. Handling such attacks by solely using human power is not effective. Although there are many anti-drone guns produced nowadays, they have limitations in handling multiple drone attacks.

One of the best alternative solutions for handling attacks from MED is by using the same technology, which is the usage of multiple drones for defending an area. These defender drones are usually called as Multiple Pursuer Drones (MPD). MPD has one main task, which is neutralizing attack from MED as soon as possible to minimize the damage caused by MED to a secured area. Although this approach has a promising result, the main challenge of MPD usage is how to coordinate the movement of each member of MPD so effectively that as a group, they can neutralize attacks from MED as soon as possible.

The term neutralizing attack in MPD vs. MED problem refers to how MPD can prevent MED from attacking an area. This process is usually conducted by chasing and capturing or immobilizing MED before the MED can attack the defended area. Any pursuer drone will perform some pathfinding and obstacle avoidance maneuvers to chase an evader drone. When the distance between a pursuer and an evader is relatively close, the pursuer can capture the evader by attacking it using any available weapon the pursuer owns. The practical implementation on how a pursuer attacks an evader depends on what technology is used by the pursuer; for example, a pursuer can use a net weapon that, when being shot to an evader, this net can capture the evader.

Although there are many methods proposed for solving the MPME problem, where some of them may be mathematically proven to catch all available evaders, some optimization is still required because, in our discussed MPME variant, the effect of the evaders' attack is the most crucial thing. In the computer science field, research about MPME is rarely discussed in detail about how a pursuer immobilizes or captures an evader because it depends on the technological specification each pursuer uses. These types of researches usually use a simplification that whenever an evader is located near to a pursuer, which is in pursuer limited capture range, the evader will be automatically captured. This research will also use the same approach, so we will not discuss in detail how the capture process is conducted.

B. REASON FOR CHOOSING PROPOSED METHOD

We propose an optimization algorithm for pursuer drones to conduct chase and capture movement to neutralize attacks from multiple evader drones. The basis of our proposed method is Social Spider Optimization (SSO) algorithm. SSO is an optimization algorithm that is introduced in 2013 [9]. SSO is a variant of the Particle Swarm Optimization (PSO) algorithm. In SSO, there is a group of spiders that collaborate to catch some prey on their communal web. There is no competition among the spiders. All spiders work together as a team to support their group survival. SSO has been used for many problems solving approaches and performs noticeable results.

In typical SSO, the population is divided into 3 main categories, which are female spiders, male-dominant spiders, and male-non-dominant spiders. Each type of spider runs a different role for the population. In SSO, there are 3 main possible procedures performed by each spider, which are the calculation of fitness value, the vibrational summons of surrounding spider, and mating procedure. This SSO characteristic inspired this paper's authors to conduct some research about the possible enhancement of SSO in solving the MPD vs. MED problem. We analyze that the principle of SSO, where there are some different roles among the population, can solve MPD vs. MED problem more efficiently because each pursuer drones can conduct a more efficient movement. As a consequence, we analyze that the development of this method will bring some novelty and perform better than currently available MPME algorithms for solving the discussed problem in this paper.

Other than SSO, some possible methods can potentially be used as alternatives for solving our discussed problem. For example, the principle of the artificial bee colony, as explored in [10], or locust search algorithm as improved in [11]. Both methods are based on the swarm-optimization algorithm. Although both methods might bring some promising results for solving our discussed problem, we choose SSO as the basis of our proposed method because we analyze that the individuals' variation among the population in SSO is more suitable to be developed as the defender algorithm for our MPD. As described earlier in our problem domain, our pursuer drones' main objective is to defend an area from the group of MPD's attack. Thus, we need to develop our MPD algorithm with a good defender algorithm.

To the best of our knowledge, there is still limited publication towards SSO direct usage for solving the MPME problem. According to state-of-the-art research analysis, because SSO is a new and promising approach for solving the optimization problem, enhancement about this approach toward open research issues such as the discussed MPME variant problem could bring a novelty aspect to computer science research area.

II. PROBLEM FORMULATION

In our discussed problem variant, there are exist a group of pursuer drones P containing n_p pursuers and a group of evader drones E containing n_e evaders. Both P and E are moving in a 3D space S, where $S \subset \mathbb{R}^N$. We denote any pursuer as p_i, where the position of each pursuer as x_p^i for $i \in \{1, 2, \ldots, n_p\}$. Meanwhile, we denote any evader as e_j, where the position of each evader as x_e^j for $j \in \{1, 2, \ldots, n_e\}$. In general, we use the term agent to describe any drone, whether it is a pursuer or an evader. Thus, the total number of agents in our environment denotes as $n = n_p + n_e$.

Every evader e_j has a goal to attack a global target that is guarded by a group of pursuers. We denote the global target as G, where its position is denoted as x_G. Every evader's attack produces a damage D at a time t_c towards G if and only if the position x_e^j is in the damage radius of G. We denote the damage radius as r_{damage}. Every evader e_j that is inside r_{damage} radius in time t_c produces damage to G according to the evader's distance from x_G. We use evader's distance as the divisor in (1) because the closer an evader from G, the higher the risk it can produce from its attack. Meanwhile, if an evader is outside r_{damage} radius in time t_c, then e_j produces 0 damage to G in time t_c. Equation (1) resumes the damage function produced by any evader e_j at a time t_c.

$$D(e_j)_{t_c} \begin{cases} \dfrac{100}{\left\| x_e^j - x_G \right\|^2}, & if \left\| x_e^j - x_G \right\| \le r_{damage} \\ 0, & if \left\| x_e^j - x_G \right\| > r_{damage} \end{cases} \tag{1}$$

onsider $\widehat{E}$ as a subset of evaders group E which having positions inside damage radius r_{damage} of global target G. In another word, $\widehat{E} \subset E$ and if $e_j \in \widehat{E}$ then $\left\| x_e^j - x_G \right\| \le r_{damage}$. By using this term, if $\left|\widehat{E}\right|$ denotes the number of $\widehat{E}$ member, then total damage that is given to global target G by a group of evaders at a time t_c is shown by (2). Please notice that (2) can also be calculated as (3) because according to (1), the damage $D(e_j)_{t_c}$ will be 0 if $\left\| x_e^j - x_G \right\| > r_{damage}$. In general, the damage accumulation of G during the execution time ($Damage_{accum}$) is a sum up result of total damage at each time t as shown in (4). In (4), $\mathrm{t_{end}}$ represents the last time when the experiment is held.

$$Total\_Damage_{t_c} = \sum_{j=1}^{\left|\widehat{E}\right|} D(e_j)_{t_c}, \quad where\ e_j \in \widehat{E} \tag{2}$$

$$Total\_Damage_{t_c} = \sum_{j=1}^{n_e} D(e_j)_{t_c}, \quad where\ e_j \in \mathrm{E} \tag{3}$$

$$Damage_{accum} = \sum_{t=1}^{t_{end}} Total\_Damage_t \tag{4}$$

The main objective of evaders E is to create as many damages as possible to G. In other words, E tries to maximize the amount of $Damage_{accum}$. Meanwhile, the main objective of pursuers P is to prevent attacks from E towards G. P tries to minimize $Damage_{accum}$ as low as possible. The lower $Damage_{accum}$ produced from P vs. E interaction, the better P performance is. Thus, in general, the best performance could be produced in this MPME problem variant is gained when $Damage_{accum}$ is 0. P tries to minimize $Damage_{accum}$ by capturing all evaders E as fast as possible to prevent E from conducting attack to G. In general, P handles E by chasing them, which means moving to get closer to E, then if e_j is located inside capture radius r_{capture} of p_i, then e_j is captured by p_i. If a pursuer p_i captures an evader e_j at a time $\mathrm{t_c}$, then e_j remains captured for all time $\mathrm{t} > \mathrm{t_c}$. As shown in (5), an evader e_j is captured by a pursuer p_i at a time $\mathrm{t_c}$ if the distance between them is less than capture radius r_{capture} of a pursuer p_i. When an evader has been captured, it is immobilized and cannot create any damage to G.

$$if \left\| x_p^i - x_e^j \right\| \le r_{capture} \quad then\ e_j\ is\ captured\ by\ p_i \tag{5}$$

Another parameter that can be used as performance evaluation parameter beside $Damage_{accum}$ is execution time. Execution time refers to the amount of $\mathrm{t_{end}}$ needed for pursuers P to capture all evaders E. As shown in (4), $\mathrm{t_{end}}$ represents the last time for the experiment to be conducted. Experimentally speaking, $\mathrm{t_{end}}$ is assigned as time t when all evaders E have been captured. In the practical world, $\mathrm{t_{end}}$ cannot be measured because information about how many evaders E left is unknown. However, in an experimental environment, the number of evaders E is always known. Thus, $\mathrm{t_{end}}$ can be used as a supporting parameter to measure the performance of the proposed method in solving this MPME problem variant. The less the value of $\mathrm{t_{end}}$ has, the better the performance of the algorithm in solving the discussed problem is. Please notice that although there are two parameters used as the performance measurement indicator, the $Damage_{accum}$ is the main parameter used as the performance indicator because it represents the main objective of the proposed method, which is solving MPME problem variant in preventing attacks from a group of evaders E.

III. RESEARCH SCOPE

We develop a coordination algorithm for improving the performance of multiple pursuers P in capturing multiple evaders E in the Multiple-Pursuer Multiple-Evader (MPME) problem variant. The problem variant we discussed in this paper is every evader in E does not try to run away from P. However, a group of evaders E tries to attack a specific area (target) guarded by P. E act as a *kamikaze* troop where they do not consider the way back home maneuver. What they do care is conducting attack as much as possible to its target.

This work presents a theoretical work instead of a real-world application. However, the result of this work can be used as a fundamental-guidance in developing methods to overcome the related real-world problems. Because this work scope is in theoretical work, some complex factors existing in the real-world might be ignored. We do not discuss errors that can occur in pursuer sensors or any environmental challenges that may be faced by pursuers in a real-world scenario. We also do not explain in details about what kind of sensors needed by pursuers to chase and capture an evader, or technology needed by pursuers to communicate with each other, or even weapon used by evaders to attack their global target G. There are many kinds of research explaining technology or algorithm nowadays that drones obtain to track or follow an object, or even conducting some obstacle avoidance algorithm, for example [12]–[16]. The communication technology among autonomous drones is also well developed nowadays, as shown in [17], [18]. Thus, we will not discuss in detail about the above aspects. What we focus on is developing a coordination algorithm among pursuers P to capture multiple evaders E as effectively as possible.

Please notice that there have been some studies related to the MPME problem focusing on the theoretical approach, for example, [19]–[21]. In theoretical discussion, the practical implementation on how a pursuer capture an evader is rarely explained in details because it depends on technology implementation that a pursuer own, for example, a pursuer can shoot a net weapon to an evader so accurately that the evader is immobilized then drop down to the ground. See [22] for more details info about the nowadays anti-drone system. In theoretical discussion, there is no limit to the amount of weapon ammunition that an agent has. So, every pursuer can capture an unlimited number of evaders if the evader meets (5). On another side, any evader can conduct an unlimited attack on global target G as long as it has not been captured.

Agents in the MPME problem can be manifested as many possible entities, for example, ground robots or unmanned vehicles. In this research, we use drones as our agent manifestation. We set up a 3D simulation environment where each agent can move freely by conducting possible movements of a drone in a real-world environment. The reason why we choose drone as our agent manifestation is that a drone can move freely in any direction in a 3D-environment; besides, it can stay still in a position without having to move like a standard fixed-wing aerial vehicle that should always move forward to gain lifting force. This drone characteristic makes the simulation more dynamics, and experiment results could be more comprehensive. Another main reason for this is because, in real-world circumstances, drones swarm have been used as a real weapon to attack an area. Thus, we hope by using drones as our agent manifestation, the result of this experiment in the future can be used as a foundation to solve real-world MPD vs. MED problem.

Because we test our proposed method in a simulation environment, we use the term *iteration* to refer time t needed during our experiment. One iteration is defined as a duration needed by each agent to change its position from one point to another point. In our experiment, each agent has the same maximum speed, so, in each iteration, every agent can move with the same distance. If there are n agents during an experiment, then in one iteration, there can be n total movement performed by all agents where each agent only performs 1 movement. We use the term *iteration* to measure t_{end} because iteration is a parameter that can be measured objectively without relying on hardware specification used to run the simulation. If we use clock time duration to measure t_{end}, the result can be varied depending on simulator hardware specification. Thus, in this experiment, time t is measured as the number of iterations.

IV. PAPER CONTRIBUTION

A. SSO BASIC CONCEPT

Social Spider Optimization (SSO) is an optimization algorithm that is inspired by the behavior of spiders in nature. It was introduced in 2013 by [9]. In nature, there are 2 types of spider groups, namely solitary spider groups and spider groups that live in a colony. In the SSO algorithm, the basic inspiration for this algorithm is the behavior of spiders who live in a colony. The paradigm used in SSO is the coordination mechanism carried out by a group of spiders who work together in finding prey and mates. In SSO, the search space is called a communal web. Each spider interacts with each other through the communal web. Every spider works together as a team to capture preys on their communal web. There is no competition among the spiders.

Each spider in SSO can communicate with other spiders by vibrating the communal web. From the results of these vibrations, the spider population can determine the direction of motion of the group, whether to approach the source of vibration or away from the source of vibration. This algorithm is a variation of Particle Swarm Optimization (PSO). At SSO, there is a spider population consisting of a group of individuals of different sexes. Each sex has a different tendency of movement characteristics.

The basic principle of SSO is similar to PSO, where each individual can communicate with other individuals so that the entire population can obtain optimum results efficiently. The main characteristic of individuals in SSO, in general, is that the individuals are divided into a female (**F**) and male types. The male group was divided into dominant male (**D**) and non-dominant male (**ND**). Each **F** can call individual **D** to come closer toward **F**. Meanwhile; every individual **D** tends to be near **F** spiders to perform the mating procedure. Like the behavior in nature, **ND** groups tend to gather in areas where there are no individual **D**. The proportion of **D** and **ND** in a population is around 50%: 50%.

In the SSO concept, each individual has a fitness value (FV) that is informed to the entire population through the communal web. FV is the value of the effectiveness of an individual solution. In the computer world, FV can be calculated based on how close an individual is to a target/solution. One of the interesting things in SSO is that there is a unique paradigm of mating between individual **F** and individual **D**. Two individuals can mate and produce a new individual with FV resulting from a combination of individual **F** and individual **D**. If the FV results from new individuals are better than with the weakest individual's FV in the population, the weakest individual will be destroyed and replaced with a new individual. However, if a new individual does not have a better FV than the weakest individual's FV in the population, then this new individual will not survive (disappear).

In nature, the aim of one individual spider to call another individual is to mate or together attack food target (prey). A female spider can attract a dominant male spider based on the female spider's FV condition and its distance towards the male spider. In computer science, the aim of calling other individuals is to produce new individuals (alternative solutions) or to explore a possible solution jointly. Because individual **F** can call individual **D**, and each male tends to keep his distance to each other, then the possibility of trapped populations in local optimum solutions can be minimized. If there is no vibration of the call made by any individual, then each individual will be silent or move freely according to their instincts. The instinct of each individual is looking for food, where in terms of computer science, food is a parable for the target/solution. When an individual gets closer to a food source, then the individual's FV value gets higher.

The complexity of SSO arises when, in one moment, there is a group of individuals who vibrate the communal web to call other individuals. In SSO, the number of individual **F** is far more than individual **D**, where an individual **D** can come and mate more than 1 individual **F**. When there is more than 1 call in the population, the population movement algorithm needs to be designed so systematically that any wrong area neglection can be prevented. In the concept of SSO, each **ND** can change to **D**, and vice versa depending on the condition of each's FV.

SSO characteristics that can minimize the occurrence of local optimum solutions makes this algorithm is often used to solve various problems such as Artificial Neural Network Training [23], [24], Support-Vector-Machine Parameter Tuning [25], [26], Controller Design [27]–[29], Frequency Controller [30], Image Processing [31], [32], Distributed Renewable Energy [33], Congestion Management [34], or Anti-Islanding Protection [35]. [36] shows several international publications that have been indexed by JCR (Journal Citation Report) from 2014 to 2017. From [36], it is shown that SSO is an algorithm that can be used to solve various problems.

B. NOVELTY AND CONTRIBUTION

In general, there are 3 main procedures usually conducted in SSO: calculation of fitness value, the vibrational summons of surrounding spiders, and mating procedure. Detail variety of each process depends on the type of problem being solved. In our proposed method, we enhance the basic concept of SSO as introduced on [9] and develop some novel setup and calculation to optimize SSO performance in solving the MPME problem variant discussed in this paper. The details about the enhancement will be explained in the next section. We try to emphasize the enhancement in each subsection.

Since it was first published, SSO has been used by various domains. Although SSO has now been used as an algorithm to solve a variety of problems, unfortunately, there is no comprehensive research related to the use of SSO in solving the MPME problem variant as the issue raised in this study. The MPME problem variant discussed in this study is also still being an open research issue. The enhancement of SSO to handle the topic of problems in this study certainly has its novelty aspect because a relatively new method such as SSO is developed to deal with an open research issue.

According to researches conducted using the SSO approach, there are some tuning or modifications required to produce an optimal result of the SSO algorithm for solving a specific problem. That is why although the principle of SSO is promising for solving our discussed problem, we still propose some enhancement method. In the following section, we will discuss in detail what parts of SSO we enhance and the reason behind the enhancement.

This research contributes an alternative solution to the open problem of counteracting attacks from a group of evader drones. To the best of our knowledge, there have not been any international journal publications that discuss in detail the strategies to deter evader drones' attacks against a region. Many MPME problem solution tends to deal with the ''cops and robbers'' problem. Our research to offer a new method dealing with discussed MPME problem variant surely bring novelty and contribution to the computer science field.

V. PROPOSED METHOD

Before we jump to the mathematical issue of the proposed method, we would like to highlight that the proposed method we explain here is a coordination algorithm among a group of pursuer drones P to prevent attack from a group of evader drones E toward a global target G. Thus, when we explain further about the SSO as our basic method, it means that every spider in the SSO refers to one single pursuer drone p_i. We manage to explain our proposed method with the same structure of processes needed in the basic SSO approach. Please notice that in each following subsection, we will emphasize the enhancement we do to improve the SSO performance in dealing with our discussed MPME problem variant.

Although in the SSO approach, there are 3 variants of spiders: female, dominant-male, and non-dominant-male; the implementation of each variant in our pursuer drones only affects their movement behavior. Any specification of equipment, speed, or capability of each our pursuer drone is set to be the same. We do not differentiate purser drones' performance capability. We only differentiate their movement behavior.

A. SSO POPULATION INITIALIZATION

In the conventional SSO, the population of spider colonies is dominated by female spiders, which account for 65-90% of the population. For this reason, at the stage of population initialization, it is necessary to determine the number of N_f that indicates the number of female spiders and the N_m, which indicates the number of male spiders. The total number of spiders in the population is N. In the initialization stage, a randomization function needs to be used so that the SSO can approach the natural characteristics of spider colonies in the wild.

$$N_f = floor[\text{random}\,(0.65, 0.90)\, x\, N] \tag{6}$$

$$N_m = N - N_f \tag{7}$$

Equation (6) shows the basic SSO initialization for determining the number of female individuals in a population. In (6), it appears that there is a random function that will produce a decimal number between 0.65 to 0.9, which will then be multiplied by the total number of population (N). The floor function in (6) is used to round down the result if the result of multiplying the percentage of females and the number of individuals in the population is not an integer. Next, in (7), the number of male individuals is calculated by reducing the entire population by the number of female spiders.

To formalized the notation, the spider population in an SSO is symbolized as **S** (spider), with a total of N individuals, where the **S** population consists of two sets, namely the female group **F** (female) and the male group **M** (male). To facilitate indexing, if individuals in $\mathbf{S} = \{s_1, s_2, \ldots, s_N\}$, individuals in $\mathbf{F} = \{f_1, f_2, \ldots, f_{Nf}\}$, and individuals in $\mathbf{M} = \{m_1, m_2, \ldots, m_{Nm}\}$, then the first Nf individual in **S** is female spiders and individuals in the index greater than Nf in **S** are male spiders. Simply put $\mathbf{S} = \mathbf{F} \cup \mathbf{M}$, where $\mathbf{S} = \{s_1 = f_1, s_2 = f_2, \ldots, s_{Nf} = f_{Nf}, s_{Nf+1} = m_1, s_{Nf+2} = m_2, \ldots, s_N = m_{Nm}\}$.

The proposed enhancement that we do in this SSO aspect is determining the fixed percentage of female numbers in the population. In the basic SSO, as shown in (6), the percentage of Nf compared to N can be between 65% to 90%. The role of **F** in population is to attract dominant male spiders to move toward a female spider. Thus, if the percentage of **F** is low, then there will be a lot of dominant-male (**D**) spiders in the population that should be summoned by **F**. If this happens, then spider population tends to be gathered in some narrow spot around **F**. For the discussed MPME variant, this situation could make many areas unguarded, thus it could bring high damage potential for global target G. Meanwhile, if the proportion of **F** is high, there will be few **D** spiders that can be summoned when **F** is calling. If this happens, then SSO coordination will fall into the conventional MPME algorithm because the essential part of the SSO algorithm is the summoning process. Understanding the tradeoff caused by Nf, we come to an analysis that the proper proportion of Nf should be 85%. We change the basic SSO equation, as shown in (6) into (8). We have done some preliminary experiments showing that the best performance of SSO for solving the discussed MPME problem variant is produced when the proportion of Nf compared to total population number N is 85%.

As mentioned in the basic SSO concept, male spiders are classified into two categories: dominant and non-dominant. A male spider m_i is categorized as a dominant spider if the weight value of the W_i spider is greater than the median weight value of the overall male spiders. If not, then male spider m_i is categorized as a non-dominant male. The formulation of the decision whether a male spider m_i is a dominant male spider or not can be seen in (9) and (10). Please notice that because the weight Wi of a male spider is related to its FV, and FV value can dynamically change according to the surrounding evaders' position, then the median value of male spiders can so dynamically change too that the category of each male spider can dynamically change along with the experiment.

$$N_f = floor[0.85\text{x}\, N] \tag{8}$$

$$m_i \in \boldsymbol{D}, \quad if\ FV\,(m_i) > \widetilde{FV}(\boldsymbol{M}) \tag{9}$$

$$m_i \in \boldsymbol{ND}, \quad if\ FV\,(m_i) \le \widetilde{FV}(\boldsymbol{M}) \tag{10}$$

After the number of females in a population is determined, the next step is to place existing individuals in the search area (communal web) so that the initial position of individuals can be spread evenly and proportionally. Because **F** spiders have a role as spiders that can summon other **D** spiders, we set the **F** spiders' initial position in the outermost perimeter of the pursuer group. Meanwhile, we locate the male spiders in the inner perimeter. Because at the initial position, there should no evaders detected, so the median of FV for all males spiders should be 0. Thus there is no dominant male at the beginning of the initialization state. Fig 2 shows an initial position example when there are 25 spiders in the population. Please notice that Fig 2 is just a 2D simplification of the 3D simulation environment used in our experiment.

In Fig 2, the center of image represents the global target G. According to (8) and (7), because N is 25, then Nf is 21, and Nm is 4. In Fig 2, the pink number represents female spiders **F**, while the blue number represents non-dominant male spider **ND**. In this paper, **D** is represented by using a green color. However, because there is no dominant spider in the initialization state, there is no green number in Fig 2. The initial position might be slightly varied according to the number of SSO population. However, in general, we put the

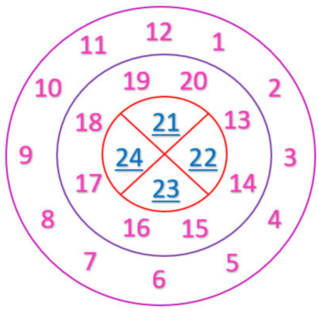

FIGURE 2. Initial spiders position example in the proposed method.

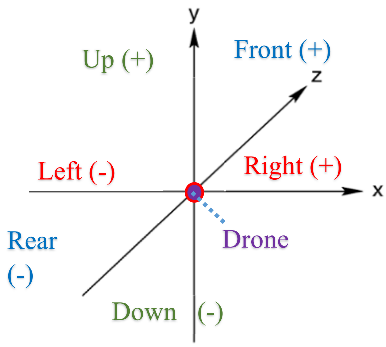

FIGURE 3. Axis orientation in 3D search space.

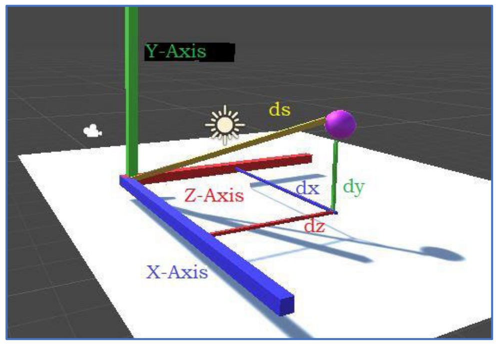

FIGURE 4. Visualization of distance in 3D space.

male spiders in the inner perimeter; while we put the female spiders in the outer perimeter.

B. SSO INDIVIDUAL WEIGHTING

Each individual in SSO has a fitness value (FV), which indicates the quality of the solution obtained by an individual at a particular time. The FV value in SSO is very dependent on the domain of the problem being handled, for example, the individual's FV in a route searching problem is better when an individual gets closer to the intended target. We use the term *weight*(W) as an FV normalization value for each spider. The *weight* term is also used in the basic SSO algorithm. The FV normalization process to produce weight is shown in (11). Each weights *Wi* is notated in a decimal number between 0 to 1. In (11), FV(s_i) shows the fitness value of a spider s_i from population **S**. Meanwhile, $best_s$ and $worst_s$ show the best and worst FV values of spiders in population **S**. As a mathematical equation, the $best_s$ and $worst_s$ can be notated through (12). In SSO, the greater the *Wi* indicates the better the quality of the solution owned by an individual s_i.

We do not modify the calculation process of each spider's weight. The equation (11) and (12) are equations used in the basic SSO method. What we do develop is the calculation method to measure FV of a spider, as will be discussed in the following passage. In the case of capturing an evader drone, a pursuer drone has a higher weight when it is closer to an evader. Therefore, the FV value will be higher when the distance between a pursuer and an evader is getting smaller. In 3D space, the distance between two objects (*ds*) is the resultant vector of the distance between the x-axis (*dx*), y-axis (*dy*), and z-axis (*dz*). The direction of the three axes in this research can be seen in Fig 3. In Fig 4, you can see an illustration of how to measure *ds* of a purple circle object from the center of the axis.

$$W_i = \frac{\mathrm{FV}(\boldsymbol{s}_i) - worst_s}{best_s - worst_s}, \quad i \in \{1, 2, \ldots, N\} \tag{11}$$

$$best_s = \max_{\mathrm{k}\in\{1,2,\ldots,N\}} (\mathrm{FV}(\boldsymbol{s}_\mathrm{k})) \,|\, worst_s = \min_{\mathrm{k}\in\{1,2,\ldots,N\}} (\mathrm{FV}(\boldsymbol{s}_\mathrm{k})) \tag{12}$$

From Fig 4, if the center of the axis is the location of a spider, then the distance of an evader from the spider (*ds*) can be calculated using the resultant vector formula in (13).

Because the weight value of a spider is inversely proportional to the distance between the spider and the target, then to calculate the FV of an individual, we propose the formula stated in (14) to be used. In (14), the FV value is calculated as the inverse distance between individuals and targets. This formula is quite simple and represents the characteristics of the problem being studied.

$$ds^2 = dx^2 + dy^2 + dz^2 \tag{13}$$

$$FV(\boldsymbol{s}_i) = \frac{1}{ds(s_i, target)} \tag{14}$$

$$FV(\boldsymbol{s}_i) = \sum_{t=1}^{k} \left(\frac{1}{ds\left(s_i, target_t\right)}\right) \tag{15}$$

If a spider detects more than 1 target around it, then, of course, the FV should also increase because this situation indicates that there are many preys around a spider. Thus if there are k targets around the individual s_i, then we propose total FV (s_i) in (14) to be modified into (15). By using (15), the calculation formula FV (s_i) can be more proportionate to the number of targets faced by an individual s_i.

C. SSO VIBRATION MODELLING

As explained earlier, each individual in SSO can communicate using vibrations propagated through the communal web. In SSO, there are 3 types of vibrations that affect the movements of an individual s_i, which are:

1) Vibration from the **closest** individual with a weight greater than s_i ($Vibc_i$: vibration that is closest to s_i).

This vibration is felt by s_i and is produced by another individual s_c, no matter whether male or female, where s_c is the closest individual to the s_i who has weight $W_c > W_i$.

2) Vibration from the individual with the **best** FV ($Vibb_i$: vibration from the best spider). This vibration is felt by s_i and is produced by another individual, no matter whether male or female, where $W_b = best_s$.
3) Vibration from the **closest female** ($Vibf_i$: vibration from the closest female to s_i). This vibration is only produced by female spiders to attract male spiders to approach themselves. This vibration only affects the male spider. This vibration is felt by s_i and is produced by female spider s_f, who is the closest female to s_i.

Like the characteristics of the intensity of the waves, the farther the center of vibration from an individual, the smaller the vibration will be felt. In SSO, to reduce the vibration felt by the individual s_i due to the vibration of the individual s_j, the weight of the s_j will be reduced by using some division scaling factor powered by Squared Straight Line Distance (SSLD) between the position of s_i and s_j. Equation (16) shows the SSLD formula between two individual s_i and s_j. Please notice that in (16), each individual is placed in a 3D space of communal web with coordinates in the form of (x, y, z) axes, where respectively $s_{i.x}$, $s_{i.y}$, $s_{i.z}$ denotes the coordinate points of s_i on the x, y, and z axis.

$$SSLD_{i,j} = \left\|s_i - s_j\right\|^2 = \left(s_{i.x} - s_{j.x}\right)^2 + \left(s_{i.y} - s_{j.y}\right)^2 + \left(s_{i.z} - s_{j.z}\right)^2 \quad (16)$$

In SSO, for the same distance, the greater the vibration value produced by s_j towards s_i, the greater the impact of the vibration for the s_i. The formulas of vibrations found in basic SSO can be seen in (17), (18), (19), which respectively formulate various vibrations that have been previously described. Please notice that in (17), (18), (19), for the same magnitude of vibration value, the farther the distance of s_j vibration sources from an individual s_i, the smaller the vibrations felt by s_i due to the $e^{SSLDi,j}$ scaling factor.

$$\mathrm{Vibc}_i = \frac{W_c}{e^{SSLD_{i,c}}} \quad (17)$$

$$\mathrm{Vibb}_i = \frac{W_b}{e^{SSLD_{i,b}}} \quad (18)$$

$$\mathrm{Vibf}_i = \frac{W_f}{e^{SSLD_{i,f}}} \quad (19)$$

Although the $e^{SSLDi,j}$ scaling factor seems proportional in reducing the vibrational effect from an individual, we analyze that the scaling factor tends to make the vibrational factor become 0 even when the distance between a spider toward a vibration source is near. If this happens, then the vibrational factor seems useless. Thus, to optimize this factor for utilizing the SSO algorithm in solving the discussed MPME problem, we propose some different scaling factors to enhance the computational resource to measure vibrational factor impact from individuals. We remove the use of exponential factor *e*.

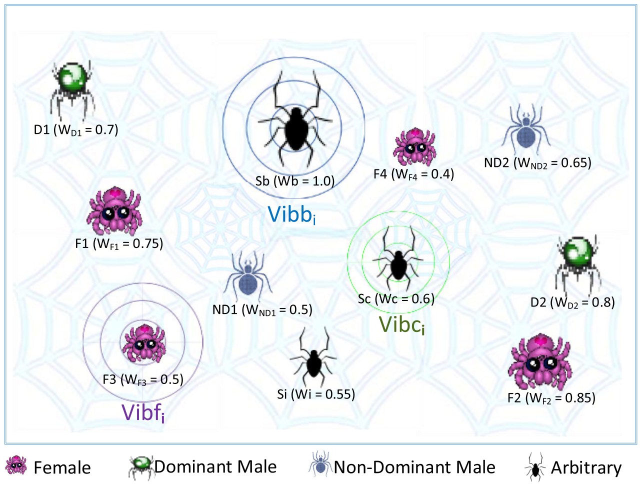

FIGURE 5. Various vibrations that affect individual movement in SSO.

We propose (20), (21), and (22), respectively, to enhance the (17), (18), (19). The reason behind this decision is because we analyze that the use of power function towards *e* brings unnecessary computational resources. We consider that using SSLD itself as a scaling factor is quiet represents the proportionality of distance factor towards the vibrational effect of a spider. In general, we propose the vibrational factor of a spider s_j towards spider s_i as a division of weight s_j (W_j) by the SSLD between the two spiders.

$$\mathrm{Vibc}_i = \frac{W_c}{SSLD_{i,c}} \quad (20)$$

$$\mathrm{Vibb}_i = \frac{W_b}{SSLD_{i,b}} \quad (21)$$

$$\mathrm{Vibf}_i = \frac{W_f}{SSLD_{i,f}} \quad (22)$$

Fig 5 shows an illustration of various vibrations that possibly exist in SSO. In Fig 5, please focus the observation on the black spider s_i. In Fig 5, the black spider indicates the sex of the spider does not need to be considered because it does not affect the computation of SSO algorithm. The spider has a *Wi* weight of 0.55. To find Vibc_i felt by s_i, it is necessary to determine in advance the closest spider from s_i that has a weight greater than the weight of the s_i. Notice in Fig 5 that the closest spider to s_i is ND1. However, because W_{ND1} is no greater than W_i, the vibration of ND1 is not considered by s_i as Vib_c. Thus, we move to the next closest spider from s_i, i.e. s_c, where W_c is 0.6, which means it is greater than W_i. Because s_c meets the criteria, Vibc_i for the case in Fig 5 comes from s_c regardless of the sex of s_c.

The next vibration that affects s_i is Vibb_i. Vibb_i is produced by individuals with the largest W weights in the population. Based on Fig 5, it is clear that the individual with the greatest weight is s_b, with a W_b of 1. Therefore, regardless of sex s_b, Vibb_i vibration for the case of Fig 5 is produced by s_b. The last vibration that affects s_i is Vibf_i vibration coming from the nearest female spider. Based on Fig 5, it is clear that the

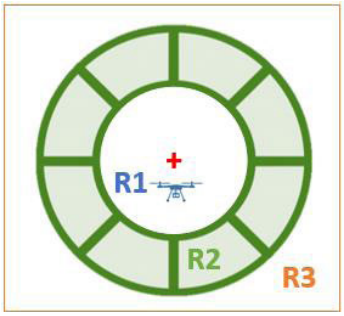

FIGURE 6. Various regions of a pursuer drone.

closest female spider to s_i is f_3. Therefore, the $Vibf_i$ vibration felt by s_i in Fig 5 is produced by f_3.

D. SSO INDIVIDUAL MOVEMENT DECISION

In order not to be easily trapped in the local optimum solution, every individual in the SSO has different movement characteristics depending on the sex of a spider. In general SSO, female spiders are not affected by Vibf. The female spider f_i is only affected by the vibrations of $Vibc_i$ and $Vibb_i$. On another hand, dominant male spiders are affected only by Vibf; while non-dominant male spiders are not affected by any vibrational factor. SSO individual movement is the most crucial factor in SSO algorithm performance. The main enhancement we propose lies in this aspect. For each spider sex in SSO, we propose some enhancement to their movement algorithm.

As a common assumption used on MPME theoretical research, each pursuer can detect the precise position (coordinate) of evaders when the evaders are in the pursuer's detection range. As shown in Fig 6, each pursuer has 3 different ranges around its location. Assume the center of Fig 6 represents the position of a pursuer. R2 represents the detection range of a pursuer. Any evaders' that are located inside this region can be accurately detected. Meanwhile, R1 refers to the capture range of a pursuer. It is similar to $r_{capture}$ in (5). The pursuer will automatically capture every evader located inside the R1 region. The last region of a pursuer is R3. R3 represents the limit of a pursuer's detection range. A pursuer cannot detect any evader in the R3 region because the distance is too far. We also use this common approach for our proposed method.

1) FEMALE SPIDER MOVEMENT

In the basic SSO algorithm, female spiders move according to (23), where f_i^k denotes the position of female spider f_i in k^{th} iteration, while α, β, and δ are random numbers between (0,1). In (23), besides $Vibb_i$ and $Vibc_i$, a female spider movement is affected by some internal factors. First is the tendency towards stimulus factors. This factor determines whether a female spider tends to move closer to a stimulus, or she moves away from a stimulus. This factor is denoted as different "+" (plus) and "-" (minus) signs in (23). In the basic SSO, this factor is determined randomly with some probability factor PF. The basic SSO algorithm will generate a random value between 0 and 1; then, if the value is lower than PF, the female spider is designed to move towards a stimulus. Otherwise, the female spider moves away from a stimulus. This factor is designed to imitate the hormonal-like factor of a female spider. The second factor is designed to imitate the instinct-like factor of a female spider. This factor is notated as a random factor in (23). According to (23), the female spider movement is affected by some random factor. In the basic SSO algorithm, this random factor is designed to prevent a female spider to be trapped in some local optimum solution.

We propose some enhancement to the basic SSO female spider movement in (23). We analyze that the hormonal-like factor seems to make the performance of female spiders might be lower because female spiders are sometimes allowed to move away from a stimulus. Whereas in the discussed MPME problem variant, when vibration from a spider is high, it indicates that there are many evaders around the spider. Thus, other spiders should move toward the vibration source to capture more evaders. By using this consideration, in our proposed method, we drop the use of the hormonal-like factor and set each spider should move toward stimuli she feels.

$$f_i^{k+1} = \begin{cases} f_i^k + \alpha.Vibc_i.\left(\boldsymbol{S}_c - f_i^k\right) + \beta.Vibb_i.\left(\boldsymbol{S}_b - f_i^k\right) \\ +\delta.random(-0.5, 0.5) \\ \quad \text{with probability PF} \\ f_i^k - \alpha.Vibc_i.\left(\boldsymbol{S}_c - f_i^k\right) - \beta.Vibb_i.\left(\boldsymbol{S}_b - f_i^k\right) \\ +\delta.random(-0.5, 0.5) \\ \quad \text{with probability } (1 - \text{PF}) \end{cases} \tag{23}$$

The next enhancement we propose in SSO female spider movement is by replacing the random instinct-like factor of a female spider with a calculated external factor based on detected preys' location. We introduce the term of $UnVec_{prey}$ as a parameter to be considered by a female spider when making a move. $UnVec_{prey}$ represents the **unit vector** towards a free-and-nearest prey from a spider. It is related to the position of the closest target around an individual. We propose a mechanism where an individual tends to get closer to its nearest target. It is similar to spider's nature, which is more likely to move toward the nearest food or prey trapped in the communal web.

To prevent the wasteful resource allocation of available spiders, we manage to avoid sieging maneuvers conducted by the spiders. We tend to prevent different spiders from being attracted by the same $UnVec_{prey}$ source. Thus, we use the term $e_{cl(i)}$ as the evader producing $UnVec_{prey}$ to a spider s_i. We use e_{cl} to notate the closest evader that is not being chased by any other pursuer. Because each pursuer can know the exact position of an evader (when the evader is inside R2 region of Fig 6), every pursuer can communicate with each other to mark which evader is being chased by which pursuer. After determining the e_{cl} of a spider s_i, we calculate $UnVec_{prey}$ by using (24). $UnVec_{prey}$ is calculated as the vector difference between the location of a spider (s_i) and its e_{cl} location.

$$UnVec_{prey} = \frac{\overrightarrow{e_{cl(i)}} - \overrightarrow{s_i}}{\left\| \overrightarrow{e_{cl(i)}} - \overrightarrow{s_i} \right\|} \tag{24}$$

The last factor we enhance for female spider movement is the values of α, β, and δ in (23). In basic SSO, these values are set to be a random number between (0,1). We analyze that making these values random creates inconsistent behavior of female spiders' movement. Although setting these values as random numbers might bring female spiders not being trapped in local optimum, we consider this value should be some **constant** numbers because it will make female spider movement performance more stable. However, because we remove the use of random instinct factor, we do not use any δ variable. As a summary of our enhanced female spider movement, we formulate our proposed enhancement method as (25) to replace (23), where the values of α and β are constant and will be mentioned in the experiment results section.

Please notice that we use a different paradigm to define the female spiders' movement formula. Instead of defining the exact position of a spider in the next iteration, we define the next position of a spider as a vector resultant. Every spider s_i has a speed v_i. Thus, the position of a female spider in the next iteration is determined by the unit vector direction it wants to conduct multiplied by the speed it has. In this research, we use "<>" notation to inform the reader that we need to calculate the unit vector of the vector resultant inside the symbol.

$$f_i^{k+1} = \begin{cases} f_i^k + UnVec_{prey} * v_i, \\ \quad if\ e_{cl(i)} \neq \emptyset \\ f_i^k + \langle [\alpha.Vibc_i.\langle \boldsymbol{S}_c - f_i^k \rangle + \beta.Vibb_i \langle \boldsymbol{S}_b - f_i^k \rangle] \rangle .v_i, \\ \quad if\ e_{cl(i)} = \emptyset \end{cases} \tag{25}$$

2) DOMINANT MALE SPIDER MOVEMENT

In the basic SSO algorithm, a Dominant-Male (**D**) spider m_i's movement is affected only by $Vibf_i$ as shown in (26), where α and δ are random numbers between (0, 1). In (26), m_i^k represents the position of dominant male spider m_i in the k^{th} iteration. In the basic SSO, a dominant-male spider tends to come closer to a female spider. To optimize the SSO performance, we replace this characteristic by managing a dominant male spider m_i to come to the female spider with the highest vibrational effect to m_i. We call this female spider as f_i^{best}. Please notice that f_i^{best} might be different from the best spider s_b because in calculating vibration, the distance between two spiders holds a big impact.

Similar to female spiders' characteristics, in basic SSO, **D** spiders' movement is also affected by some random instinct-like factor. We consider this **D** random instinct-like factor as an unnecessary factor. So, with a similar analysis for the enhancement of female spiders' movement, we replace the instinct-like factor with $UnVec_{prey}$ factor, as shown in (24). As a result, we change the basic SSO dominant male formula (26) to (27), where we remove the use of α and δ.

$$m_i^{k+1} = m_i^k + \alpha.Vibf_i.\left(s_f - m_i^k\right) + \delta.random(-0.5, 0.5) \tag{26}$$

$$m_i^{k+1} = \begin{cases} m_i^k + UnVec_{prey} * v_i, & if\ e_{cl(i)} \neq \emptyset \\ m_i^k + \langle f_i^{best} - m_i^k \rangle * v_i, & if\ e_{cl(i)} = \emptyset \end{cases} \tag{27}$$

3) NON-DOMINANT MALE SPIDER MOVEMENT

The role of Non-Dominant male (**ND**) spiders in the basic SSO is to guard the region in the communal web, which is not occupied by any female or dominant male spider.**ND** spiders work as an anticipation plan to catch solitary preys which are not gathered with their hordes. The general movement of **ND** spiders is shown in (28), where α is a random number between (0, 1). Generally speaking, each **ND** spider tends to gather in the central position of every **ND** spider by also considering the weight of each **ND** spider.

$$m_i^{k+1} = m_i^k + \alpha.\left(\frac{\sum_{h=1}^{N_{ND}} m_h^k.W_h}{\sum_{h=1}^{N_{ND}} W_h} - m_i^k\right)$$
$$where\ m\ is\ a\ NonDominant\ spider \tag{28}$$

$$m_i^{k+1} = \begin{cases} m_i^k + \langle X_{cl}^{nd(i)} - m_i^k \rangle .v_i, & if\ e_{cl}^{nd(i)} \neq \emptyset \\ m_i^k + \langle \boldsymbol{X}_G - m_i^k \rangle .v_i, & if\ e_{cl}^{nd(i)} = \emptyset \end{cases} \tag{29}$$

According to (28), there is no other factor affecting the movement of ND spiders besides the parameters of each ND spiders. In the basic SSO algorithm, this ND movement is managed not to be affected by other types of spiders. There is a benefit by using this approach, which is ND movements tend to be independent; thus, ND spiders are hoped to be occupied in the necessary area. Nevertheless, different from the basic SSO principle, we analyze that making ND spiders independent potentially brings a critical drawback, which is they cannot adapt their position to what is happening in the communal web.

As an enhancement, in this research, we propose a new behavior of ND as shown in (29) where m_i^k represents the position of a non-dominant male spider m_i in the k^{th} iteration. Because we try to solve the MPME problem variant where the evaders manage to attack a global target G, we propose the use of non-dominant spiders as the last-perimeter of G defenders. We set the stand by location of **ND** spiders near to G. If there are some evaders successfully manage to come near to G, the **ND** spiders will try to chase the evaders, starting from the closest one. We use the term of $e_{cl}^{nd(i)}$ as the term of closest evader to an **ND** spider m_i. **ND** spider ignores whether $e_{cl}^{nd(i)}$ is being chased by another pursuer. As long as there is an evader near to G, **ND** will chase it. The position of $e_{cl}^{nd(i)}$ is notated as $X_{cl}^{nd(i)}$ in (29). If by any chance there are many evaders are located near to G, there is a possibility where **ND** spiders' FV increase significantly and raise above **FV** median of male spiders. If this happens, the **ND** spiders will change the role to be **D** spiders and use the dominant spider movement algorithm.

E. SSO MATING PROCESS

→ One interesting procedure in SSO is the mating process between dominant male (**D**) and female (**F**) spiders. In the basic SSO, 1 spider **D** can mate more than 1 spider **F**. When the mating stage is carried out, the characteristics of the new generation are a mixed combination of the parameters held by both parents. Let's say a new individual s_{new} is produced from the mating of D_{parent} and F_{parent}. If each spider has n parameters p_1, p_2, ..., p_n, where $p_k(s_i)$ represents the k^{th}-parameter value of the individual s_i, then the value of each parameter in the new individual s_{new} can be written according to equation (30). These p_k parameters could be a spider position on the x-axis, y-axis, z-axis, a spider speed, etc. Note that the new individual parameter is a mixed collaboration of the parameters of the female parent or dominant male parent.

$$p_k(\boldsymbol{s}_{new}) = \min_{k\in\{1,2,\ldots,n\}} \text{between } \left(p_k\left(\mathrm{D_{parent}}\right) \quad \text{and } p_k\left(F_{\mathrm{parent}}\right)\right) \tag{30}$$

When all new individual s_{new} parameters have been produced, the fitness function is run to calculate the FV of the new individual. If this FV is higher than the weakest individual FV in the population, then the weakest individual will be destroyed and the new individual will have the same sex as the weakest individual who has just been destroyed. However, if the FV of this new individual is no better than the weakest individual FV, then this new individual will be immediately destroyed when it has just been produced.

For some researches using SSO as its solution algorithm, the SSO mating process might bring some good benefits because this procedure can replace the weakest individual with the better one. However, this approach cannot be conducted in this research MPME problem variant because it is impossible to destroy a pursuer drone when chasing evaders. One approach that seems appropriate to replace the destruction step of the weakest individual is by moving the weakest individual to the position of "new-born" individual. However, this step might consume time. Besides, we analyze that this step can be substituted with the summoning process produced by vibration factors. Thus, in this enhancement method, we ignore and exclude the SSO mating process.

VI. RESULT AND ANALYSIS

A. BASIC CONFIGURATION

We implement our proposed method in a 3D-simulator using Unity Engine, where the basic environment module can be downloaded from https://unitylist.com/p/hcm/. Fig 7 shows the general interface of our simulator environment. In our simulator, the physical appearance of each drone agent is similar, regardless of whether it is a pursuer or an evader. The only difference between pursuers and evaders is located in their color. Fig 8 shows the sample difference of the agents we use. The physical body of an evader is colored with brown color. Meanwhile, the pursuer drones have a pink color for females, green color for dominant males, and blue color for non-dominant males. As shown in Fig 7, there is a small mini-map on the top right corner of our simulator screen to visualize the position of each drone in our simulator. The pink circle in Fig 7 mini-map indicates the border of the defended area. Please note that each pursuer drone cannot access the position of all evaders. Each pursuer can only know the position of evaders inside its R2 region, as shown in Fig 6.

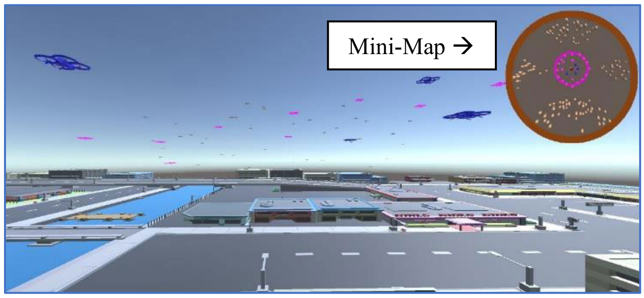

FIGURE 7. General interface of simulation environment.

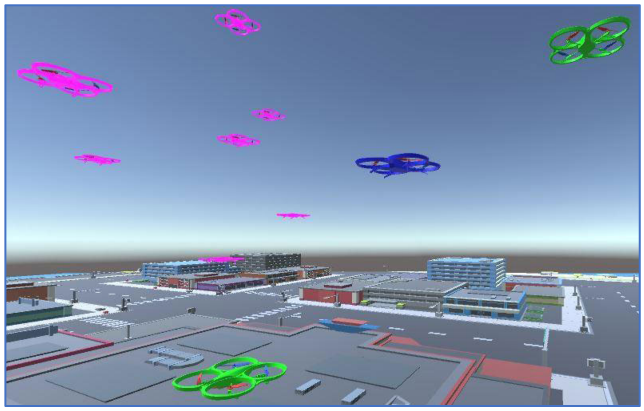

FIGURE 8. Color variety in pursuer drones.

The colors in our simulator mini-map also represent the agents in our simulation environment: brown for evaders, pink for female spiders, green for dominant male spiders, and blue for non-dominant male spiders. In our mini-map, there is a little red-white-black circle at the center of mini-map. This circle represents the coordinate of evaders' global target G. In our simulator, for each iteration, every agent can move up to 0.5 distance units. Meanwhile, evaders' damage range r_{damage} in (1) is set to be 10 units. From the pursuers side, every pursuer has r_{capture} (R1) set to be 2 units. Meanwhile, the pursuer detection range (R2) is set to be the same as r_{damage}, which is 10 units. Our mini-map is taken using a sky view approach; it means the bigger circle of an agent in our mini-map, the higher its position from the ground is.

To prevent the situation where pursuer drones lured to move too far from global target G, we determine the maximum range of search space for each pursuer. We set the maximum search range for female and dominant male spiders as 2 times of r_{damage}, while the maximum range of search space for non-dominant males is set to be twice of r_{capture}.

FIGURE 9. The huge amount of swarm evader drones in experiment.

If by any chance, a pursuer is located outside its maximum range, the pursuer will move towards G position until the pursuer gets inside its maximum range. In our experiment, G is only represented as a coordinate. It is represented as a 3-color ball (red-white-black) in the sky. Because the R2 region of each pursuer is set to be 10 distance units, the height variety of every agent in our experiment is set around $\pm$ 5 units (above and below) from our global target G's height.

B. BEHAVIOR OF EVADER DRONES

The behavior of evader drones surely influences the performance of pursuer drones in handling attacks from evaders. There can be a lot of variants for this movement. The more complicated the evaders' movement strategy, the more $Damage_{accum}$ they probably can produce. Evaders drone may conduct some runaway maneuvers from pursuers. However, this cannot bring any benefit for the evaders because the evaders tend to get farther from global target G. The farther an evader from G, the less its probability to conduct damage towards G.

Because the problem variant discussed in this paper is about evader drones conducting a *kamikaze-like* attack to a global target G, we remove evaders' capability to run away from pursuers. Every evader moves directly to a global target G without considering the location of pursuer drones. Although this movement seems straightforward, handling this maneuver is quite difficult, especially when the number of evaders is high. As shown in Fig 9, there are many brown evaders in front of some pursuers. Handling Fig 9 situation effectively surely requires a sophisticated approach.

The formula of evader drones' movement is shown in (31), where $x^j_{e(t)}$ represents the position of evader e_{j} at iteration t, meanwhile v^j_e represents the speed of evader e_{j}. In each iteration, every evader e_{j} moves directly in the constant height towards the global target G position. When evader e_{j} reaches a position where it is near to G, it moves directly to G. Every pursuer doesn't know evaders' maneuver. Each pursuer can only detect the existence of evaders in the pursuer's limited detection range. Every evader cannot collide to any pursuer and the evaders are not programmed to attack any pursuer. The only objective evaders have is getting closer to G as soon as possible then perform attacks to G.

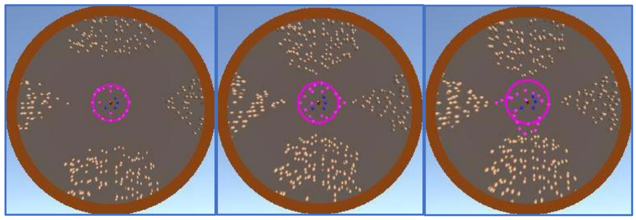

FIGURE 10. Evader formation variants in conducting attacks.

$$x^j_{e(t+1)} = \begin{cases} x^j_{e(t)} + \langle \left(x_{G.x} - x^j_{e(t).x}\right)\hat{x} + \left(x_{G.z} - x^j_{e(t).z}\right)\hat{z}\rangle * v^j_e \\ \quad if \ \left\| x^j_e - x_G \right\| > \dfrac{r_{damage}}{2} \\ x^j_{e(t)} + \langle x_G - x^j_{e(t)}\rangle * v^j_e, \\ \quad if \ \left\| x^j_e - x_G \right\| \le \dfrac{r_{damage}}{2} \end{cases} \tag{31}$$

There are two main formation evaders conduct in this research, as shown in Fig 10. First is an arrow-like formation that imitates the shape of the triangle group. This formation is shown on the left side and right side of Fig 10. The second formation is a rectangle-like formation, as shown in the upside and downside of Fig 10. In each formation, every evader has varied altitudes. In the initialization step, the nearest evader is located in 40 distance units from global target G. We use these two formations because this formation can represent the possible basic formation of swarm evader drones when attacking a global target. We do not use an arbitrary-formation because the consistency for this formation handling is hard to be measured.

C. EXPERIMENT PARAMETER

In this research, we propose an optimization algorithm for solving a problem variant of MPME. Thus, to measure the effectiveness of our proposed method, we compare our proposed method with other methods for solving the discussed MPME problem variant. We use 3 methods for comparison. Table 1 describes the maneuver conducted in each of our comparison methods. To the best of our knowledge, there is no open-source method that comprehensively discusses the code to handle the MPME problem variant raised in this research. For that case, we develop our comparison method as objective as possible.

Because we use comparison-method as our basis measurement approach, the behavior of evader drones does not affect the comparison results. If the evader drones' movement gets more complicated, the performance result of our proposed method and the basis-comparison method will become lower. Thus, by using only one evader drones' movement behavior, as explained previously, the performance analysis of the proposed method is sufficient.

As described in the problem formulation and research scope sections, we use two parameters to measure our

TABLE 1. Methods variety used for performance comparison.

ID	Method Name	Behavior
WD	Waiting Defender	All pursuers conduct standby maneuver around global target *G*. The pursuers surround *G* from the distance of r_{damage}. The pursuers try to capture evaders if the evaders already have near or inside r_{damage} radius. There is no coordination algorithm among pursuers. Every pursuer tries to capture the nearest evader from the pursuer's coordinate.
BF	Brute Force	All pursuers try to capture the nearest evader detected. There is no coordination algorithm among pursuers. The maximum searching space range for each pursuer is set to be the same as the maximum search space range of our proposed method.
ST	Switching Target	There is some communication strategy among pursuers. Each pursuer tries to capture the nearest evader that is not being chased by another pursuer. The maximum search space range for each pursuer is set to be the same as the maximum search space range of our proposed method.
Prop	Proposed Method	SSO enhancement method proposed in this paper.

TABLE 2. Values used for experiment.

No.	Variable	Value
1	Female Population Percentage	85%
2	Pursuer and Evader Drones movement speed	0.5 units per iteration
3	Pursuers' R1 capture radius ($r_{capture}$)	2 units
4	Maximum height margin between global target G position and agents	10 units (5 x 2 → above-below)
5	Pursuers' R2 detection range	10 units
6	Evaders' damage range, or Global Target G damage range (r_{damage})	10 units
7	Nearest evaders distance to global target G position	40 units
8	Maximum Search Range of Female and Dominant Male Spiders from global target G position	2 * 10 units = 20 units
9	Maximum Search Range of Non-Dominant Male Spiders from global target G position	4 units
10	Experiment Repetition	10 times
11	β	3α

TABLE 3. Mean damage accumulation measurement result.

No	P	E	WD	BF	ST	Prop
1	20	20	53	5	45	40
2	20	100	500	859	421	324
3	20	200	2,369	4,115	1,333	1,737
4	40	40	3	0	0	0
5	40	200	400	440	123	45
6	40	400	585	1,348	274	187
7	60	600	676	607	150	67
8	60	900	1,229	993	137	87
9	60	1,200	3,203	1,149	270	177
10	60	1,800	5,207	3,101	275	243

Abbreviation:
No = Case Number, P = Number of Pursuers, E = Number of Evaders, WD = Waiting Defender, BF = Brute Force, ST = Switching Target, Prop = Proposed Method.

proposed method performance: $Damage_{accum}$ and iteration time. Iteration time starts at time $t = 1$ where every agent is located in its initial positions as described earlier. Iteration time stops at t_{end} when there is no evader in the simulation environment. If two methods perform the same $Damage_{accum}$, then a method with less t_{end} indicates that the method performs better than another. Because of this parameter, we do not program each evader to perform a hiding maneuver because it could bring bias to our performance measurement.

We do not use the success rate as our parameter because, at the end of each experiment, every method should capture all the evaders because the evaders do not try to run away from any pursuer. The success rate in each of our experiments is always 100%. According to our problem domain, the success rate is not quite relevant to be used as a measurement parameter because what this problem domain needs is how to minimize the damage caused by evaders. We do not use travel-distance either as our measurement parameter because of this problem domain concern.

The main parameter used as our performance measurement is $Damage_{accum}$. It directly indicates how much damage a global target *G* receives from evaders' attack. To gain comprehensive results about this parameter, we conduct varied cases for the number of pursuers and evaders involved in each experiment. The following section will describe the variety of experiments we conduct. Because the MPME problem is a dynamic problem, where each agent involved can conduct different movements in every experiment, we repeat simulation for each case up to 10 times then calculate the average results as our measurement value.

Because in (25) we use α and β as multiplying factors for vibrational effect, where the final multiplication result should be in the form of a unit vector; thus, the exact value of α and β does not matter. The important thing about this parameter is the comparison value between the two of them. Because we consider that many evaders should surround the best spider, we set the value of β with a 3 times value of α. If the distance of best spider s_b is too far from a spider s_i, of course, the impact of β is neglectable. Table 2 contains the summary of values we use in our simulation. The result of our simulation is discussed in the next section.

D. EXPERIMENT RESULT

Table 3 and Table 4 describe the experiment results we obtain from our simulation. Please remember that the values written in both tables are calculated from the mean values of each aspect after experimenting with each case for 10 repetitions. The more detail value for data in Table 3 can be observed in Table 5 and Table 6. Meanwhile, the more detail value for data in Table 4 can be observed in Table 7 and Table 8.

Please notice that the number of pursuers and evaders in Table 5 – Table 8 is similar to the number of pursuers

TABLE 4. Mean iteration time measurement result.

No	P	E	WD	BF	ST	Prop
1	20	20	86	76	83	86
2	20	100	108	112	116	113
3	20	200	129	131	133	128
4	40	40	90	81	80	80
5	40	200	124	116	119	112
6	40	400	155	150	156	144
7	60	600	175	170	172	164
8	60	900	207	194	190	192
9	60	1,200	232	223	217	215
10	60	1,800	291	273	271	271

Abbreviation:
No = Case Number, P = Number of Pursuers, E = Number of Evaders, WD = Waiting Defender, BF = Brute Force, ST = Switching Target, Prop = Proposed Method.

TABLE 5. Damage accumulation measurement result detail 1.

No	WD			BF		
	Min	Max	Std	Min	Max	Std
1	0	380	110.8	0	28	9.0
2	136	1,400	355.6	310	1,625	346.9
3	1,769	3,398	586.1	2,704	5,888	1,041.2
4	0	17	5.1	0	0	0.0
5	12	1,025	359.3	37	898	264.0
6	55	1,310	406.2	335	2,516	794.8
7	1	2,630	958.6	150	1,470	440.5
8	28	3,849	1,401.6	343	1,785	459.4
9	7	6,376	2,491.4	289	3,530	923.2
10	1,590	10,418	2,597.5	714	5,884	1,418.4

Abbreviation:
No = Case Number, Min = Minimum Value, Max = Maximum Value, Std = Standard Deviation, WD = Waiting Defender, BF = Brute Force.

and evaders in Table 3 and Table 4 for each case number. For readability easiness, we round the values in Table 3 – Table 8 except for the value of standard deviation. In our analysis section, we focus our discussion on Table 3 and Table 4 data. Fig 11 and Fig 12 show the plot of each method's performance measured from the mean of $Damage_{accum}$ aspect and *iteration time* parameters. For each parameter, the less value produced by a method, the better performance the method has. According to our experiment, in general, the proposed method performs better than the other compared methods.

E. PERFORMANCE ANALYSIS

According to data shown in the previous section, there are two main parameters we can analyze for performance measurement. First is damage accumulation ($Damage_{accum}$), and second is iteration time (t). For $Damage_{accum}$ parameter, we can analyze that our proposed method mostly performs better than the other algorithms. Our proposed method is only outperformed by Switching Target (ST) algorithm in case 3. Meanwhile, our proposed method is also outperformed by Brute Force (BF) algorithm in case 1. We analyze the reason behind this is because the number of pursuers is still low, which is only 20.

TABLE 6. Damage accumulation measurement result detail 2.

No	ST			Prop		
	Min	Max	Std	Min	Max	Std
1	0	151	61.8	22	89	21.5
2	177	720	184.6	132	535	139.6
3	808	1,852	310.0	1,061	2,943	645.5
4	0	5	1.5	0	1	0.3
5	0	388	135.8	0	148	40.4
6	52	683	185.7	25	430	115.8
7	0	300	113.7	6	239	64.7
8	7	282	96.5	17	246	66.8
9	72	467	109.9	20	354	102.1
10	144	723	164.1	71	551	134.7

Abbreviation:
No = Case Number, Min = Minimum Value, Max = Maximum Value, Std = Standard Deviation, ST = Switching Target, Prop = Proposed Method.

TABLE 7. Iteration time measurement result detail 1.

No	WD			BF		
	Min	Max	Std	Min	Max	Std
1	81	96	5.08	70	83	4.22
2	104	114	2.79	100	118	5.73
3	126	134	2.47	126	140	4.45
4	89	92	1.00	77	86	2.50
5	123	126	0.90	110	125	4.67
6	153	157	1.02	142	155	4.56
7	174	176	0.83	160	174	4.72
8	207	208	0.46	187	200	4.14
9	229	234	1.73	211	233	7.24
10	290	292	0.64	270	285	4.27

Abbreviation:
No = Case Number, Min = Minimum Value, Max = Maximum Value, Std = Standard Deviation, WD = Waiting Defender, BF = Brute Force.

TABLE 8. Iteration time measurement result detail 2.

No	ST			Prop		
	Min	Max	Std	Min	Max	Std
1	73	100	10.83	81	91	3.07
2	95	126	10.37	104	121	4.57
3	124	144	7.69	124	136	3.54
4	77	91	3.95	76	88	3.08
5	107	138	11.52	103	120	5.40
6	143	173	9.40	134	155	6.95
7	155	184	11.32	157	173	4.15
8	187	203	4.65	187	203	4.76
9	209	239	9.81	209	225	5.99
10	270	272	0.54	270	273	0.83

Abbreviation:
No = Case Number, Min = Minimum Value, Max = Maximum Value, Std = Standard Deviation, ST = Switching Target, Prop = Proposed Method.

According to our experiment, when the number of pursuers increases, the proposed method's performance result becomes higher. As explained earlier, the basic approach of our proposed method relies on the coordination among pursuers (spiders). Thus, when the number of pursuers is not so high, the coordination effectiveness of the proposed method is not so high either. When the number of the pursuers is high, the performance of our proposed method is good.

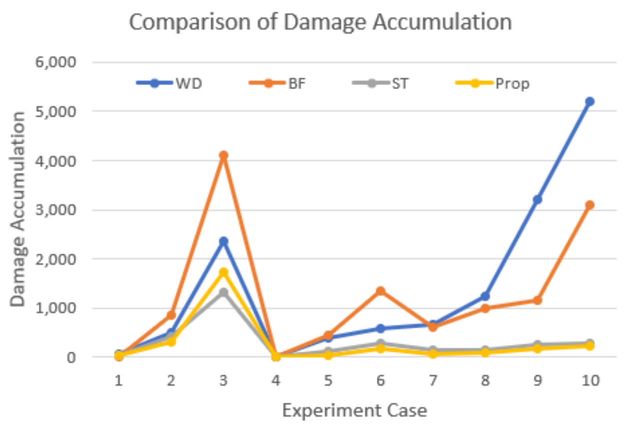

FIGURE 11. Comparison of mean damage accumulation chart.

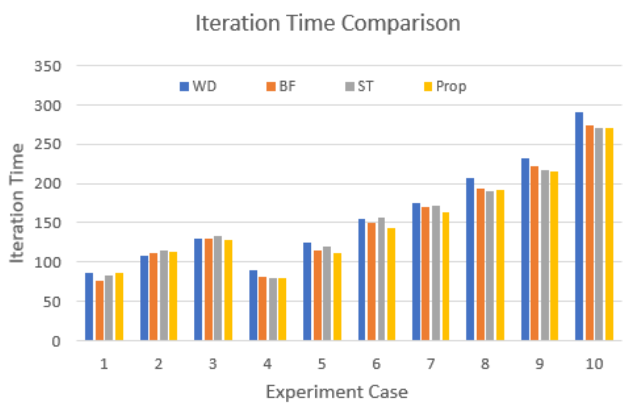

FIGURE 12. Mean iteration time comparison chart.

We realize that we have not compared our proposed method with any other methods beyond our authors' design. However, the compared methods used in the authors' experiment can generally reflect the foundation methods available to handle the MPME problem variant discussed in this paper. The WD, BF, and ST algorithms are the basic principle of any possible algorithms used for solving the discussed MPME problem. Thus, comparing our proposed method with these algorithms is technically sufficient as an objective performance measurement approach.

The use of **ND** spiders as the last perimeter defender contributes significantly to our proposed method. When there are some evaders successfully infiltrate to the near global target G, these **ND** spiders can be used to capture the evaders. Enhancement on female and dominant-male spiders in holistic also contributes to our good performance. Mathematically speaking, we introduce some new equations to be used as SSO enhancement for solving the discussed MPME problem variant. Thus, in general, we can state that the proposed method brings novelty to the computer science research field, where it produces a magnificent result.

From the iteration time parameter, our experiment results show that there are no significant differences among all compared methods for each case. It means that the proposed method does not significantly affect the time needed for pursuers to catch evaders. Because the pursuers in the proposed method conduct a distributed algorithm, the computational load among the pursuers is evenly shared. If we relate the iteration time with the energy used for capturing evaders, we can conclude that the proposed method does not increase energy consumption significantly to capture all evaders. In some cases, the proposed method even decreases the iteration time. We understand that there are some other factors related to energy consumption calculation. Nevertheless, we provide iteration time as a basic parameter to predict the energy raising needed by the proposed method to perform the proposed algorithm.

Because the main parameter used for the performance evaluation is $Damage_{accum}$, we want to highlight the experiment results for this parameter. According to our experiment in Table 3, some data show our proposed method outperforms the other algorithms. Here, we ignore the result of Case 4 in Table 3 because the value of each method is around 0. Case 10 shows that our proposed method reduces the $Damage_{accum}$ of WD algorithm up to 95%. Case 10 also shows that our proposed method reduces $Damage_{accum}$ of BF algorithm up to 92%. Meanwhile, Case 5 shows that our proposed method can reduce $Damage_{accum}$ of ST algorithm up to 63%. The reduced percentage of $\mathrm{Damage}_{\mathrm{accum}}$ produced by our method depends on the number of pursuers and evaders involved in the experiment.

Although the problem domain exposed in this paper is related to a dynamic environment, we realize that the experiment result may be varied in each repetition for the same case. That is why we repeat the experiment in each case for 10 times of repetition. Because our experiment results have created a pattern showing that our proposed method has a stable improvement when the number of pursuers has been high, we can conclude that our proposed method brings a significant novelty to the computer science research field.

VII. CONCLUSION

This research develops a coordination algorithm for improving the performance of multiple pursuers P in capturing multiple evaders E on the Multiple-Pursuer Multiple-Evader (MPME) problem variant. The problem variant we discussed in this paper is every evader in E does not try to run away from P; however, a group of evaders E tries to attack a specific area (target) guarded by P. The basis of our proposed algorithm is Social Spider Optimization (SSO) algorithm. We enhance some aspects of the SSO approach to fit in our problem domain.

We compare our proposed method with three other algorithms, which are Waiting Defender (WD), Brute Force (BF), and Switching Target (ST) algorithms. According to our experiment, the best performance of our proposed method can reduce the damage of WD up to 95%, damage of BF up to 92%, and damage of ST up to 63%. Meanwhile, the iteration time for our proposed method is relatively the same as the iteration time consumed by other compared methods. From computer science theoretical knowledge, we can conclude that our proposed method brings a novelty to the computer science research field.

For future work, we would like to consider the development of some SSO variants to improve our results. We also plan to conduct some research related to this topic where the evaders can perform some sophisticated maneuvers. The most important thing in our research roadmap is implementing our proposed algorithm in the real device. For now, we hope our current research can be useful as a foundation for future development in MPME research topic area.

REFERENCES

[1] A. Pierson, Z. Wang, and M. Schwager, "Intercepting rogue robots: An algorithm for capturing multiple evaders with multiple pursuers," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 530–537, Apr. 2017.

[2] Z. Zhou, W. Zhang, J. Ding, H. Huang, D. M. Stipanović, and C. J. Tomlin, "Cooperative pursuit with Voronoi partitions," *Automatica*, vol. 72, pp. 64–72, Oct. 2016.

[3] W. Li, "A dynamics perspective of pursuit-evasion: Capturing and escaping when the pursuer runs faster than the agile evader," *IEEE Trans. Autom. Control.*, vol. 62, no. 1, pp. 451–457, Jan. 2017.

[4] M. Aigner and M. Fromme, "A game of cops and robbers," *Discrete Appl. Math.*, vol. 8, no. 1, pp. 1–12, Apr. 1984.

[5] A. Vacca and H. Onishi, "Drones: Military weapons, surveillance or mapping tools for environmental monitoring? The need for legal framework is required," *Transp. Res. Procedia*, vol. 25, pp. 51–62, Jul. 2017.

[6] P. Chandhar, D. Danev, and E. G. Larsson, "Massive MIMO for communications with drone swarms," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 1604–1629, Mar. 2018.

[7] A. Majd, A. Ashraf, E. Troubitsyna, and M. Daneshtalab, "Integrating learning, optimization, and prediction for efficient navigation of swarms of drones," in *Proc. 26th Eur. Int. Conf. Parallel, Distrib. Netw.-Based Process. (PDP)*, Cambridge, U.K., Mar. 2018, pp. 101–108.

[8] D. Wu, D. I. Arkhipov, M. Kim, C. L. Talcott, A. C. Regan, J. A. Mccann, and N. Venkatasubramanian, "ADDSEN: Adaptive data processing and dissemination for drone swarms in urban sensing," *IEEE Trans. Comput.*, vol. 66, no. 2, pp. 183–198, Feb. 2017.

[9] E. Cuevas, M. Cienfuegos, D. Zaldívar, and M. Pérez-Cisneros, "A swarm optimization algorithm inspired in the behavior of the social-spider," *Expert Syst. Appl.*, vol. 40, no. 16, pp. 6374–6384, Nov. 2013.

[10] W. Gao, L. Huang, Y. Luo, Z. Wei, and S. Liu, "Constrained optimization by artificial bee colony framework," *IEEE Access*, vol. 6, pp. 73829–73845, 2018.

[11] O. Camarena, E. Cuevas, M. Pérez-Cisneros, F. Fausto, A. González, and A. Valdivia, "Ls-II: An improved locust search algorithm for solving optimization problems," *Math. Problems Eng.*, vol. 2018, pp. 1–15, Oct. 2018.

[12] A. Y. Husodo, H. A. Wisesa, and W. Jatmiko, "The use of direction matrix to determine autonomous quad-copter drone path when facing obstacle," in *Proc. 4th Asia–Pacific Conf. Intell. Robot Syst. (ACIRS)*, Japan, Jul. 2019, pp. 89–94.

[13] A. Y. Husodo, G. Jati, N. Alfiany, and W. Jatmiko, "Intruder drone localization based on 2D image and area expansion principle for supporting military defence system," in *Proc. IEEE Int. Conf. Commun., Netw. Satell. (Comnetsat)*, Makassar, Indonesia, Aug. 2019, pp. 35–40.

[14] A. Bahabry, X. Wan, H. Ghazzai, H. Menouar, G. Vesonder, and Y. Massoud, "Low-altitude navigation for multi-rotor drones in urban areas," *IEEE Access*, vol. 7, pp. 87716–87731, 2019.

[15] A. Rohan, M. Rabah, and S.-H. Kim, "Convolutional neural network-based real-time object detection and tracking for parrot AR drone 2," *IEEE Access*, vol. 7, pp. 69575–69584, 2019.

[16] P. Chen, Y. Dang, R. Liang, W. Zhu, and X. He, "Real-time object tracking on a drone with multi-inertial sensing data," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 131–139, Jan. 2018.

[17] L. Shan, R. Miura, T. Kagawa, F. Ono, H.-B. Li, and F. Kojima, "Machine learning-based field data analysis and modeling for drone communications," *IEEE Access*, vol. 7, pp. 79127–79135, 2019.

[18] W. Shi, J. Li, W. Xu, H. Zhou, N. Zhang, S. Zhang, and X. Shen, "Multiple drone-cell deployment analyses and optimization in drone assisted radio access networks," *IEEE Access*, vol. 6, pp. 12518–12529, 2018.

[19] M. Chen and D. Zhu, "A novel cooperative hunting algorithm for inhomogeneous multiple autonomous underwater vehicles," *IEEE Access*, vol. 6, pp. 7818–7828, 2018.

[20] A. Von Moll, D. W. Casbeer, E. Garcia, and D. Milutinovic, "Pursuit-evasion of an evader by multiple pursuers," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Dallas, TX, USA, Jun. 2018, pp. 133–142.

[21] C. Wang, W. Shi, P. Zhang, J. Wang, and J. Shan, "Evader cooperative capture by multiple pursuers with area-minimization policy," in *Proc. IEEE 15th Int. Conf. Control Autom. (ICCA)*, Edinburgh, U.K., Jul. 2019, pp. 875–880.

[22] X. Shi, C. Yang, W. Xie, C. Liang, Z. Shi, and J. Chen, "Anti-drone system with multiple surveillance technologies: Architecture, implementation, and challenges," *IEEE Commun. Mag.*, vol. 56, no. 4, pp. 68–74, Apr. 2018.

[23] L. A. M. Pereira, D. Rodrigues, P. B. Ribeiro, J. P. Papa, and S. A. T. Weber, "Social-spider optimization-based artificial neural networks training and its applications for Parkinson's disease identification," in *Proc. 27th IEEE Int. Symp. Comput.-Based Med. Syst. (CBMS)*, New York, NY, USA, May 2014, pp. 14–17.

[24] S. Z. Mirjalili, S. Saremi, and S. M. Mirjalili, "Designing evolutionary feedforward neural networks using social spider optimization algorithm," *Neural Comput. Appl.*, vol. 26, no. 8, pp. 1919–1928, Nov. 2015.

[25] D. R. Pereira, M. A. Pazoti, L. A. M. Pereira, and J. P. Papa, "A social-spider optimization approach for support vector machines parameters tuning," in *Proc. IEEE Symp. Swarm Intell.*, Orlando, FL, USA, Dec. 2014, pp. 1–6.

[26] D. R. Pereira, M. A. Pazoti, and L. A. Pereira, "Social-spider optimization-based support vector machines applied for energy theft detection," *Comput. Electr. Eng.*, vol. 49, pp. 25–38, Jan. 2016.

[27] E. Cuevas, A. Luque, D. Zaldívar, and M. Pérez-Cisneros, "Evolutionary calibration of fractional fuzzy controllers," *Appl. Intell.*, vol. 47, no. 2, pp. 291–303, Sep. 2017.

[28] H. Shayeghi, A. Molaee, and A. Ghasemi, "Optimal design of fopid controller for LFC in an interconnected multi-source power system," *Int. J. Tech. Phys. Problems Eng.*, vol. 8, no. 1, pp. 36–44, 2016.

[29] H. Shayeghi, A. Molaee, K. Valipour, and A. Ghasemi, "Multisource power system FOPID based load frequency control with high-penetration of distributed generations," in *Proc. 21st Electr. Power Distrib. Netw. Conf. (EPDC)*, Karaj, Iran, Apr. 2016, pp. 131–136.

[30] A. A. El-Fergany and M. A. El-Hameed, "Efficient frequency controllers for autonomous two-area hybrid microgrid system using social-spider optimiser," *IET Gener., Transmiss. Distrib.*, vol. 11, no. 3, pp. 637–648, Feb. 2017.

[31] S. Ouadfel and A. Taleb-Ahmed, "Social spiders optimization and flower pollination algorithm for multilevel image thresholding: A performance study," *Expert Syst. Appl.*, vol. 55, pp. 566–584, Aug. 2016.

[32] L. Maurya, P. K. Mahapatra, and A. Kumar, "A social spider optimized image fusion approach for contrast enhancement and brightness preservation," *Appl. Soft Comput.*, vol. 52, pp. 575–592, Mar. 2017.

[33] A. Laddha, A. Hazra, and M. Basu, "Optimal operation of distributed renewable energy resources based micro-grid by using social spider optimization," in *Proc. IEEE Power, Commun. Inf. Technol. Conf. (PCITC)*, Bhubaneswar, India, Oct. 2015, pp. 756–761.

[34] Z. Hejrati, S. Fattahi, and I. Faraji, "Optimal congestion management using the social spider optimization algorithm," in *Proc. 29th Int. Power Syst. Conf.*, Tehran, Iran, 2014, pp. 1–6.

[35] D. Vijay and V. Priya, "Anti-islanding protection of distributed generation based on social spider optimization technique," *Int. J. Adv. Eng. Res. Sci.*, vol. 4, no. 6, pp. 32–40, 2017.

[36] A. Luque-Chang, E. Cuevas, F. Fausto, D. Zaldívar, and M. Pérez, "Social spider optimization algorithm: Modifications, applications, and perspectives," *Math. Problems Eng.*, vol. 2018, Dec. 2018, Art. no. 6843923.

ARIO YUDO HUSODO (Member, IEEE) received the bachelor's and master's degrees in informatics from the Institut Teknologi Bandung, Indonesia, and the Ph.D. degree from the Faculty of Computer Science, University of Indonesia. He is currently a Lecturer with the University of Mataram, Lombok, Indonesia. His research interests are computer vision and intelligent systems.

GRAFIKA JATI (Member, IEEE) received the B.S. and M.Sc. degrees in computer science from Universitas Indonesia, in 2014 and 2016, respectively. He currently works as a Research Assistant with the Faculty of Computer Science, Universitas Indonesia. His research interests include visual object tracking and autonomous robot.

AMARULLA OCTAVIAN graduated from Naval Academy, in 1988, and received the Ph.D. degree from the Faculty of Social and Political Sciences, University of Indonesia, in 2013. He was the Dean of the Faculty of Defense Management, Indonesia Defense University, in 2016. He is currently the Commander of the Indonesian Naval Command and Staff College, and a Professor of Naval Operation Research and Modern Warfare. He is also a Flag Officer of Indonesian Navy.

WISNU JATMIKO (Senior Member, IEEE) received the B.S. degree in electrical engineering and the M.Sc. degree in computer science from Universitas Indonesia, in 1997 and 2000, respectively, and the Dr.Eng. degree from Nagoya University, Japan, in 2007. He is currently a Full Professor with the Faculty of Computer Science, Universitas Indonesia. His research interests are autonomous robot, optimization, and real-time traffic monitoring systems.