A UAV Optimal Path Planning Using Particle Swarm Optimization Algorithm

Institute of Aeronautics and Space Studies

HAIK Walid
HAMRANI Said

15 July 2021

Abstract

Optimization is one of the important categories of engineering problems, concerned with the search for better solutions of maximum or minimum ways in a certain areas. Because of its complexity, the meta-heuristique algorithms are usually used and applied because these algorithms don't require prior knowledge of the search space and because they also depend on theories based on randomness. A large group of this algorithms (optimization techniques) compete to find a better solution, the most prominent of which is particle swarm optimization (PSO), and given its good performance in many optimization problems, it is considered a modern method and is relatively close to experimenting with swarms. This thesis aims to provide a review and discussion of the application of the PSO algorithm and its projection to plan the path of drone swarms and we analyze its present situation of research and parameter selection, topology structure, Basic PSO algorithm and multi-objective optimization PSO and its simulations. Finally, current problems are analyzed and future research directions are presented.

Resumé

L'optimisation est l'une des catégories importantes de problèmes d'ingénierie, concernés par la recherche de meilleures solutions de manières maximales ou minimales dans certains domaines. En raison de sa complexité, les algorithmes méta-heuristiques sont généralement utilisés et appliqués car ces algorithmes ne nécessitent pas de préconnaissance de l'espace de recherche et parce qu'ils dépendent également de théories basées sur l'aléatoire. Un grand groupe de ces algorithmes (techniques d'optimisation) rivalisent pour trouver une meilleure solution, dont la plus importante est l'optimisation par essaim de particules (PSO), et compte tenu de ses bonnes performances dans de nombreux problèmes d'optimisation, elle est considérée comme une méthode moderne et est relativement proche à expérimenter avec des essaims. Cette thèse vise à fournir une revue et une discussion de l'application de l'algorithme PSO et de sa projection pour planifier le chemin des essaims de drones et nous analysons sa situation actuelle de recherche et de sélection des paramètres, la structure topologique, l'algorithme PSO de base et l'optimisation multi-objectifs PSO et ses simulations. Enfin, les problèmes actuels sont analysés et les directions de recherche futures sont présentées.

Dedication

I dedicate this work to My mother and my father, source of my happiness and success in life. May Allah bless them.

To my brother Samy, and my sister Ghania for their unconditional support and encouragement to purse my interests.

To my uncles TAHI Ali and TAHI Massinissa, the ones who tought me how to have life with a good purpose.

To my cousin, HAMAZ Zahia, my inspiration and whom i consider as my elder sister.

To my extended family and the many friends, particularly AIT DRIS Massinissa, who have been so supportive and encouraged the fulfillment of this work.

To my best thesis partner HAIK Walid.

To all my friends and colleagues of the class: Avionics

To anyone who encouraged or helped me during my studies. To all those who believed in me and pried for my success.

HAMRANISaid

Dedication

Acknowledgement

In the name of Allah, the most beneficent and most merciful. We thank Allah for all his blessing and strength that he gives us in completing this work.

We would like to express our deepest gratitude to our master thesis supervisor, Dr CHEGGAGA Nawal for her invaluable advices and guidance throughout this dissertation. Her profound knowledge and ideas to give us all for this work.

We wish to thank all jury members, our friends and those who directly or indirectly guided and helped us in this project. The knowledge and support that they shared with us will always be remembered.

Thank you all. And we will not stop here, but will continue our research until we achieve our ambitions in this project.

Contents

List of Figures

General Introduction

Due to the high complexity of real-world optimization problems, often it is not easy to solve them using traditional or deterministic optimization methods. There are many real-world optimization problems for which one can afford near-optimal solution rather than an exact solution. Therefore, a class of robust algorithms is required, which does not depend upon the particular characteristics of the problems and hence can be applied to a wide variety of problems. Evolutionary computation and swarm intelligence-based optimization algorithms serve the purpose. Swarm and evolutionary algorithms are probabilistic algorithms, which are often very effective with problems that are not easy to deal with classical optimization methods. However, we want to emphasize that it is not our intention to say that these families provide a set of all-cure solutions. In fact, because of the stochastic nature of the search process, reproducibility may become a challenging issue unless one is careful about the experiments. Often, the computational overhead could be very high also. If a problem can be tackled with a classical optimization method for which the characteristics of the solutions can be analyzed easily, our prescription is not to use swarm or evolutionary algorithms for such a problem. This thesis provides a detailed study and working procedure of one of the algorithms in the area of swarm intelligence which will be applied in the field of Aeronautics.

Chapter 1

Drones and Swarm Intelligence

1.1 Introduction

Moving from a place to another is considered a simple task for humans, they decide how to go from a place to another in a split second. For air-crafts, such a task has been a challenging problem for decades since the era of air-crafts. However scientists made a lot of efforts and developed many ways so that an air-craft can navigate from a starting position to reach it's target by following an optimal path that fits the conditions of the trip. Also with the technology advance that our world reached today, researches reached a high level so that they started imitating the behaviour of animals...etc to make intelligent machines that makes things easier. In aeronautics, an air-craft is invented by imitating birds, it's form, it's shape, even some of its characteristics are token from birds, so scientists concluded that the best way to solve a problem in this field or to invent something new, we better return to the source, which is birds or any other related source, especially when talking about Drones or UAVs.

1.2 Definition of Drones

The Meriam-Webb Dictionary describes drones as "an unmanned aircraft or ship guided by remote control or onboard computers". Drones are characterized by their remote and automatic nature. Drones are generally abbreviated as Unmanned Aerial Vehicle (UAV), emphasizing that there is no man on board and there is a remote pilot on the ground. It is unclear why the UAV

was called a drone. The typical aircraft, such as airplanes, helicopters, balloons and gliders, requires a pilot or a crew controlling the aircraft in the sky. Unmanned aircraft does not need a human pilot to operate the aircraft in the sky. These drones are either fully automatic (i.e. controlled by computer onboard) or controlled remotely from the ground. There are various expressions of drones in English, and the official names are not unified. Due to the wide variety of design and performance of unmanned aerial vehicles, legal definition is not easy. The drone covers the meanings of various types of aircraft such as UAV, RPA (Remotely Piloted Aircraft), RPV (Remotely Piloted Vehicle), UAS (Unmanned Aircraft Systems) which are divided according to each purpose [1].

1.3 History of Drones

As the proverb says, 'war accelerates human development', The first pilot-less vehicles were developed in Britain and the USA during the First World War. Britain's Aerial Target, a small radio-controlled aircraft, was first tested in March 1917 while the American aerial torpedo known as the Kettering Bug first flew in October 1918. Although both showed promise in flight tests, neither were used operationally during the war. During the inter-war period the development and testing of unmanned aircraft continued. In 1935 the British produced a number of radio-controlled aircraft to be used as targets for training purposes. It's thought the term 'drone' started to be used at this time, inspired by the name of one of these models, the DH.82B Queen Bee. Radio-controlled drones were also manufactured in the United States and used for target practice and training. Reconnaissance UAVs were first deployed on a large scale in the Vietnam War. Drones also began to be used in a range of new roles, such as acting as decoys in combat, launching missiles against fixed targets and dropping leaflets for psychological operations. Following the Vietnam War other countries outside of Britain and the United States began to explore unmanned aerial technology. New models became more sophisticated, with improved endurance and the ability to maintain greater height. In recent years models have been developed that use technology such as solar power to tackle the problem of fuelling longer flights. Drones now have many functions, ranging from monitoring climate change to carrying out search operations after natural disasters, photography, filming, and delivering goods. But their most well-known and controversial use is

by the military for reconnaissance, surveillance and targeted attacks. Since the 9/11 terrorist attacks, the United States in particular has significantly increased its use of drones. They are mostly used for surveillance in areas and terrains where troops are unable to safely go. But they are also used as weapons and have been credited with killing suspected militants. Their use in current conflicts and over some countries has raised questions about the ethics of this kind of weaponry, especially when it results in civilian deaths, either due to inaccurate data or because of their proximity to a 'target'.[7].

1.4 Uses of Drones

1.4.1 Military

Probably the oldest, most well-known and controversial use of drones is in the military. The British and U.S. militaries started using very basic forms of drones in the early 1940's to spy on the Axis powers. Today's drones are much more advanced than the UAVs of yesteryear, equipped with thermal imaging, laser range finders and even tools to perform airstrikes.

1.4.2 Delivery

Delivery drones are usually autonomous UAVs that are used to transport food, packages or goods to your front doorstep. These flying vehicles are known as "last mile" delivery drones because they are used to make deliveries from stores or warehouses close by. Retailers and grocery chains all over the country are turning to drones as more efficient delivery alternative, instead of relying on delivery drivers with inefficient trucks.

1.4.3 Emergency rescue

Sometimes it's just not safe enough to send humans into a rescue situation due to the scope or severity of the disaster. That's where drones come in. In the case of a capsized boat or drowning individual, officials can throw an Autonomous Underwater Vehicle (AUV) into the water to assist in the rescue. If there's an avalanche, drones are deployed to look for those caught in the snow.

1.4.4 Photography

Drones have been a boon for photographers, who use the UAVs to take expansive aerial photos. Ever wonder what it's like to get a bird's eye view of your favorite city, beach or building? There are drones made specifically for photography that provide a new way to photograph some of your favorite destinations from above [2].

1.5 From Drones to Swarms

Unmanned vehicles and autonomously operating cyber-physical systems continue to gain popularity for a wide range of applications . Especially unmanned aerial vehicles (UAVs), often referred to as drones, are a trending technology , for which the number of applications is rapidly rising . Generally speaking, autonomous devices can be used as (a) mobile sensing platforms , (b) actors or (c) service providers . For all three, the deployment of multiple UAVs together as a single collaborative multi-robot systems (MRS) , known as a swarm , has become possible due to increasing device performance combined with more favourable unit costs [3]. The question really is not if, but when and where drone swarms, which is the next evolution of robotic warfare, will be utilised in real-time operations. It is pertinent to note that while drone swarms may not be ready as an end state 'product', proliferation of basic swarming technology is inevitable in the coming decade across the world. Here advances in drone swarming, which is the next evolution of robotic warfare are mostly classified, though governments have given glimpses of their progress over the years. The question is when and where drone swarms will be utilised as part of a mature concept of operations (ConOps)[4].

1.6 Flocking of Birds

When living in a group of individuals (flock), there are several risks that a bird can be exposed to.There is an increased intensity of competition of the resources in the group and the individuals have to share the amount of food with the rest of the flock. In some cases also predators conspicuousness increase since a large group is more visible and noisy . Why do individuals choose to live in a flock? According to Rob Nelson [1] there is two main reasons for flocking. Primarily it contributes to protection from predators,

the flock can both keep predators away by using aggressive group defence and individuals can use the group as cover.Secondly it is easier to find food and detect predators the more eyes that are looking. As a result the birds can reduce their vigilance and therefor spend more time finding food. Or in other words, they can take full time to reach their objectives. Therefore optimization is the main act of obtaining the best result under given situations[5].

1.7 Swarm Intelligence

Swarm intelligence (SI) is based on the collective behavior of decentralized, selforganized systems. It may be natural or artificial. Natural examples of SI are ant colonies, fish schooling, bird flocking, bee swarming and so on. Besides multirobot systems, some computer program for tackling optimization and data analysis problems are examples for some human artifacts of SI. The most successful swarm intelligence techniques are Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO). In PSO, each particle flies through the multidimensional space and adjusts its position in every step with its own experience and that of peers toward an optimum solution by the entire swarm. Therefore, the PSO algorithm is a member of Swarm Intelligence[6].

1.8 Particle Swarm Optimization PSO

The Particle Swarm Optimization algorithm (abbreviated as PSO) is a novel population-based stochastic search algorithm and an alternative solution to the complex non-linear optimization problem. The PSO algorithm was first introduced by Dr. Kennedy and Dr. Eberhart in 1995 and its basic idea was originally inspired by simulation of the social behavior of animals such as bird flocking, fish schooling and so on. It is based on the natural process of group communication to share individual knowledge when a group of birds or insects search food or migrate and so forth in a searching space, although all birds or insects do not know where the best position is. But from the nature of the social behavior, if any member can find out a desirable path to go, the rest of the members will follow quickly.

1.9 Scientific community works

In the past two decades, the scientific community also neutralized or rather found solutions to obstacles that prevent drones from flying in self-driving UAVs. The majority of research and military applications focused more on the path planning problem of reconnaissance mission assisted by UAV swarms without any prior knowledge of target positions. The past two decades also have seen a very rapid and unprecedented development in the field of computational intelligence with the emergence of many powerful optimization algorithms that provide solutions through efficient and very effective hypotheses about the nature of the problem. Particle swarm optimization (PSO) is one such technique that has received wide interest by researchers to adress the problems of poorly structured, constrained continuous/discrete functional optimization. As the researchers learned about this technique, they created and developed a set of new applications with different features and different requirements as well, starting from the first theory of PSO, some of which saw success and others needed to add some changes in the algorithm structure. they published theoretical studies of the effects of different parameters and suggested many variants of the algorithm. Agents in a natural computing paradigm are decentralized entities with generally no perception of the high-level goal in pursuit yet can model complex real-world systems. This is made possible through several low-level goals which when met facilitate meaningful collective behavior arising from these seemingly unintelligent and noninfluential singular agents. An early motivation can be traced from Reeves' introduction of particle systems in the context of modeling natural objects such as fire, clouds and water in computer-based animations while at [36]. In the course of development, agents or 'particles' are generated, undergo transformations in form and move around in the modeling environment and eventually are rejected or 'die'. Reeves concluded that such a model is able to represent the dynamics and form of natural environments that were rendered infeasible using classical surface-based representations . Subsequent work by Reynolds in the Boid Model (1986) established simple rules that increased autonomy of particle behavior and laid down simple low-level rules that boids (bird-oid objects) or particles could obey to give rise to emergent behavior[35]. Today, the scientific community works to solve problems in a separate context, problems of scheduling, allocation, and resource management. Sometimes there are algorithms that can handle it perfectly but have very high time complexity. Methods have been developed to solve this type of

approximation problem, among which is the conditioning of multi-objective particle swarm optimization. whose applications are still very limited.

1.10 Conclusion

Collective behaviour of large groups of animals, or flocking, is a natural behaviour that has puzzled many people. Examples of this is not only flocks of birds, but even fish schools and mammal herds. On the other hand, the PSO method does not always work well and still has room for improvement.

Chapter 2

UAVs Path planning

2.1 Introduction

Unmanned aerial vehicles (UAVs) are used in groups often to detect targets and keep them within range of sensors. The need for communication is fundamental and absolutely essential to the missions of a drones team. Human supervisors must be in continuous contact with informations about the target even when there are obstacles. Indeed, path planning is one of the primary tasks in the process of automating a drones system that moves in the air (environment) While avoiding obstacles and respecting various restrictions and physical limitations . But so far, the possibility of creating an effective path from a given initial point to an undefined final destination in real-time conditions remains one of the biggest challenges and a difficult goal to achieve. This is due to the lack of comprehensive investigation and intuitive presentation of UAV trajectory planning in an unknown complex environment. As path planning plays an important role in enhancing UAV's autonomy level, it has to be considered in the design of a UAV.

2.2 Path Planning for Multi-UAV Formation

The multi-UAV path planning is a process that the UAVs find their own paths from their starting points to their destinations cooperatively. It has been a research focus in recent years. Of course, it is based on the path planning of the single UAV. Comparing to the single UAV path planning, the multi-UAVs need to deal with their cooperative relationships. As the formation

fly is an important cooperative way in many situations, the concept of the formation path planning is presented to solve the multi-UAV path planning problem. Specifically, the formation path planning means each UAV finds its own collision free path and simultaneously tries to keep their formation structure[38].

2.3 Path Planning Overview

From a technical perspective, path planning is a problem of determining a path for a vehicle in a properly defined environment from a starting point to a target point such that the vehicle is free from collisions with surrounding obstacles and its planned motion satisfies the vehicle's physical/kinematic constraints . In a report by [37] and [39], There are a number of terms associated with path planning which are used in different ways by different people. Some are clarified here:

2.3.1 Motion planning

This term is frequently associated with manipulator robotics. It involves deliberative high level and low level planning of a way to move a robotic manipulator.

2.3.2 Trajectory planning

Planning the robot's next movement. This term is synonymous with motion planning.

2.3.3 Navigation

A very diverse term which can have a variety of meanings. Generally it means "getting from here to there", but it also encompasses the fields of path planning, motion planning, obstacle avoidance, and localization.

2.3.4 Global path planning

The planning is done prior to vehicle movement. It uses the information from the surrounding environment to reach a target point from a starting point.

As the information contains global data, the process is slow, but the planned path may be optimal.

2.3.5 Local navigation

The process of using only the UAVs current sensed information of its immediate world to avoid obstacles and to ensure vehicle stability and safety. It is much more reactive than path planning and runs in real time. The speed at which a vehicle can fly is limited by the speed at which the local navigator can operate.

2.3.6 Obstacle Avoidance

Used in a very similar manner to local navigation, but where local navigation considers vehicle stability, safety, and goal directness, obstacle avoidance is concerned with merely getting around objects that are in the drone way.

2.4 Criteria of Path Planning

Important criteria for path planning that are commonly taken into account are the computational time, path length and completeness. A path planning algorithm with less computational time is vital in real time application, which is desirable in dynamic environments. The generated optimal path in terms of path length by a path planning technique will minimise UAV flight time and hence prolongs the UAV's endurance and life cycle, minimises fuel/energy consumption and reduces exposure to possible risks. On the other hand, a path planning approach satisfies the completeness criterion if it is able to find a path if one exists. However, sometimes, there are trade-offs between such criteria. For example, a path planning method has to disregard the path's optimality in order to increase the computational efficiency. It means that finding a slightly longer path with less computational time may be preferable. On the other hand, higher computational complexity is necessary if an optimal path is required for some reasons. These criteria have to be considered before any path planning technique/algorithm design process takes place [36].

2.4.1 Single Vehicule Path Planning

One of the main functions of an autonomous vehicle is to move itself from some initial location to locations that are required for the vehicle to execute assigned tasks. This simple function actually involve several issues. First, the vehicle must know where it currently is and the next location it should go. Secondly, it must also have the ability to plan a route between the locations and then navigate itself along the planned route.

2.4.2 Multi-Vehicule Path Planning

The driving force in the research related to planning for multiple autonomous vehicles is increasing demand of autonomous systems in the applications where a single vehicle is no longer capable of performing the necessary tasks. In most real-world applications, autonomous vehicles operate in dynamic uncertain environments. Therefore, a practical planning system must have the ability to dynamically re-plan in the face of unexpected circumstances [40].

2.5 Collision-avoidance Model for Multi-UAVs

The UAVs may collide with each other without a proper arrangement in the formation. The repulsion field of the artificial potential field method is introduced to handle this problem. When the distance between each two UAVs is less than the danger value, the repulsion field of UAV works only on the other UAVs and won't work on itself. At the same time, the path planning of the lead plane won't be affected by the repulsion fields of the wingmen so as to keep the dependence of the lead plane path planning. Therefore, the wingman is subjected to the repulsion field of other wingmen and the lead plane. the safe radius of the collision avoidance potential field is 20m. It means the UAV will be subjected to the repulsion when it gets into the other UAVs' safe radius [38].

2.6 Conclusion

It can be said as a conclusion that the UAV path planning is the trajectory planning of the UAV platforms, for example; The problem of trajectory

planning, speed, altitude, etc. is very complex and the reason for this is due to the necessity of adjusting the relationship between this study and various research fields such as control, robotics, data integration and artificial intelligence...etc., as a projection of it. Trajectory planning aims to choose and achieve the optimum path for the flight with the shortest distance and time, taking into consideration the need to bypass and ensure the most important factor, which is the safety of the vehicle throughout the flight. That is why the main techniques in path planning must include obtaining and processing information on terrain and obstacles. In this chapter, we touched on a simple introduction to drones path planning in particular, as an introduction to what comes next.

Chapter 3

Particle Swarm Optimization (PSO)

3.1 Introduction

For almost all the human activities, there is a desire to deliver the most with the least. For example, in the business point of view maximum profit is desired from least investment; maximum number of crop yield is desired with minimum investment on fertilizers. The concept of optimization has great significance in both human affairs and the laws of nature, which is the inherent characteristic to achieve the best or most favorable (minimum or maximum) from a given situation. In addition, all aspects of optimization can be viewed and studied as design optimization without any loss of generality. This makes it clear that the study of design optimization can help not only in the human activity of creating optimum design of products, processes and systems, but in the understanding and analysis of mathematical/physical phenomenon and in the solution of mathematical problems. The constraints are inherent part if the real world problems and they have to be satisfied to ensure the acceptability of the solution. There are always numerous requirements and constraints imposed on the designs of components, products, processes or systems in real-life engineering practice, just as in all other fields of design activity. Therefore, creating a feasible design under all these diverse requirements/constraints is already a difficult task, and to ensure that the feasible design created is also 'the best' is even more difficult.

3.2 Contemporary Optimization Approaches

There are several mathematical optimization techniques being practiced so far, for example gradient methods, Integer Programming, Branch and Bound, Simplex algorithm, dynamic programming, etc. These techniques can efficiently solve the problems with limited size. In addition, they could be more applicable to solve linear problems. In addition, as the number of variables and constraints increase, the computational time to solve the problem, may increase exponentially. This may limit their applicability. Furthermore, as the complexity of the problem domain is increasing solving such complex problems using the mathematical optimization techniques is becoming more and more cumbersome. In addition, certain heuristics have been developed to solve specific problem with certain size. Such heuristics have very limited flexibility to solve different class of problems. In past few years, a number of nature-/bio-inspired optimization techniques (also referred to as meta-heuristics) such as Evolutionary Algorithms (EAs), Swarm Intelligence (SI), etc. have been developed[9].

3.3 Meta-heuristic algorithms

Meta-heuristic algorithms work independently of the mathematical features of the problems of whether the function involved in the problem is differentiable, continuous, or convex, etc. The algorithms just evaluate the objective function at given decision variables and consider the optimization problem as a black box. These algorithms can be classified into three categories: evolutionary algorithms, physics-based algorithms, and swarm-intelligence-based algorithms[10].

3.4 Swarm Intelligence Algorithms

Swarm intelligence is a discipline that deals with natural and artificial systems composed of many individuals that coordinate based on the decentralized, collective and self-organized cooperative behavior of social entities like flock of birds, or school of fishes, ant colonies, animal herding, bacterial growth, and microbial intelligence. The members of a swarm must be active, dynamic and simple (with no or very little inherent knowledge of the surroundings). Within the swarm, due to this cooperative behavior, a

search strategy, better than random search, emerges. The so obtained intelligent search strategy may be referred to as swarm intelligence, in general. A well-accepted definition of swarm intelligence is , the emergent collective intelligence of groups of simple agents [15]. In addition, Swarm-intelligence-based algorithms mimic the concept of food foraging behaviour of a swarm of various creatures like ants, fish, and birds. They shared the information of visited the best places among the other creatures to find the food location. This intelligent behaviour has been observed by the researchers and the algorithms to solve the optimization problems[10]. Some examples are:

1. Particle Swarm Optimization (PSO)[11].
2. Artificial Bee Colony Algorithm (ABC)[12]
3. Ant Colony Optimization (ACO)[13]
4. Whale optimization algorithm (WOA)[14]

Where The Particle Swarm Optimization (PSO) is inspired by birds flocking or fish schooling, the Artificial Bee Colony (ABC) is motivated by foraging behavior of honeybees, while the Ant Colony Optimization (ACO) is inspired by foraging behavior of ants. An intelligent swarm can, therefore, be defined as a population of interacting individuals that optimizes a function or goal by collectively adapting to the local and/or global environment. In the area of global optimization, the swarm intelligence first appeared with PSO in 1995 and ant colony optimization (ACO) in 1992. After their invention, there was an exponential growth in the number of scientific works related to swarm intelligence and the appearance of new journals devoted to the innovations in swarm intelligence.

3.5 Particle Swarm Optimization

PSO in most basic terms belongs to the swarm intelligence paradigm, which studies the collective behavior and social characteristics of organized, decentralized, and complex systems known as “swarms.” A swarm is an apparently disorganized collection (population) of moving individuals that tend to cluster together while each individual seems to be moving in a random direction. Each individual in the swarm has the capability of interaction with the other individuals, or the so-called “agents” (or “particles” in PSO), although the

capabilities of each agent are rather limited by certain set of rules. Therefore, the behavior of an agent in a swarm is often insignificant, but their collective and social behavior is of paramount importance, in that, the swarm intelligence comes from both the collective adaptation and stochastic nature of the swarm. The main motivation stems directly from the organic swarms in nature such as bird flocks, fish schools, ant colonies, and other animal herds and packs, which exhibit an amazing self-organization and collective/social adaptation capabilities. This cannot be explained simply by the aggregated behavior of each individual members in the swarm but their collective adaptation to the environment, which in turn makes the survival in nature possible. In a PSO process, a swarm of particles, each of which represents a potential solution to the optimization problem in hand, navigates through the search space. The particles are initially distributed randomly over the search space, and the goal is to converge to the global optima of a function or a system. Each particle keeps track of its position in the search space and its best solution so far achieved. This is the personal best value (the so-called pbest) and the PSO process also keeps track of the global best solution so far achieved by the swarm with its particle index (the so-called gbest). So during their journey with discrete time iterations, the velocity of each agent in the next iteration is computed as a function of the best position of the swarm (position of the particle gbest as the social component), the best personal position of the particle (pbest as the cognitive component), and its previous velocity (the memory term). Both social and cognitive components contribute randomly to the position of the agent in the next iteration. This is illustrated in Fig. 2.1 where particle a has a new velocity update (at time t1), which can evade the nearby local optimum. This is of course an optimistic illustration and there is absolutely no guarantee that it will happen as such, since the cognitive and social components' contributions to the velocity update are all random; however, the tendency toward local and global best (similar to the "survival of the fittest" paradigm in the other EAs), and repeated trials with random scales may yield a convergence to the global optimum sooner or later. Note that its probability of success further rises due to the numerous number of particles in the swarm, since it does not matter if all fail to achieve this but one. This is the main philosophy behind the PSO and the next section is devoted to its detailed description and formulation[1].

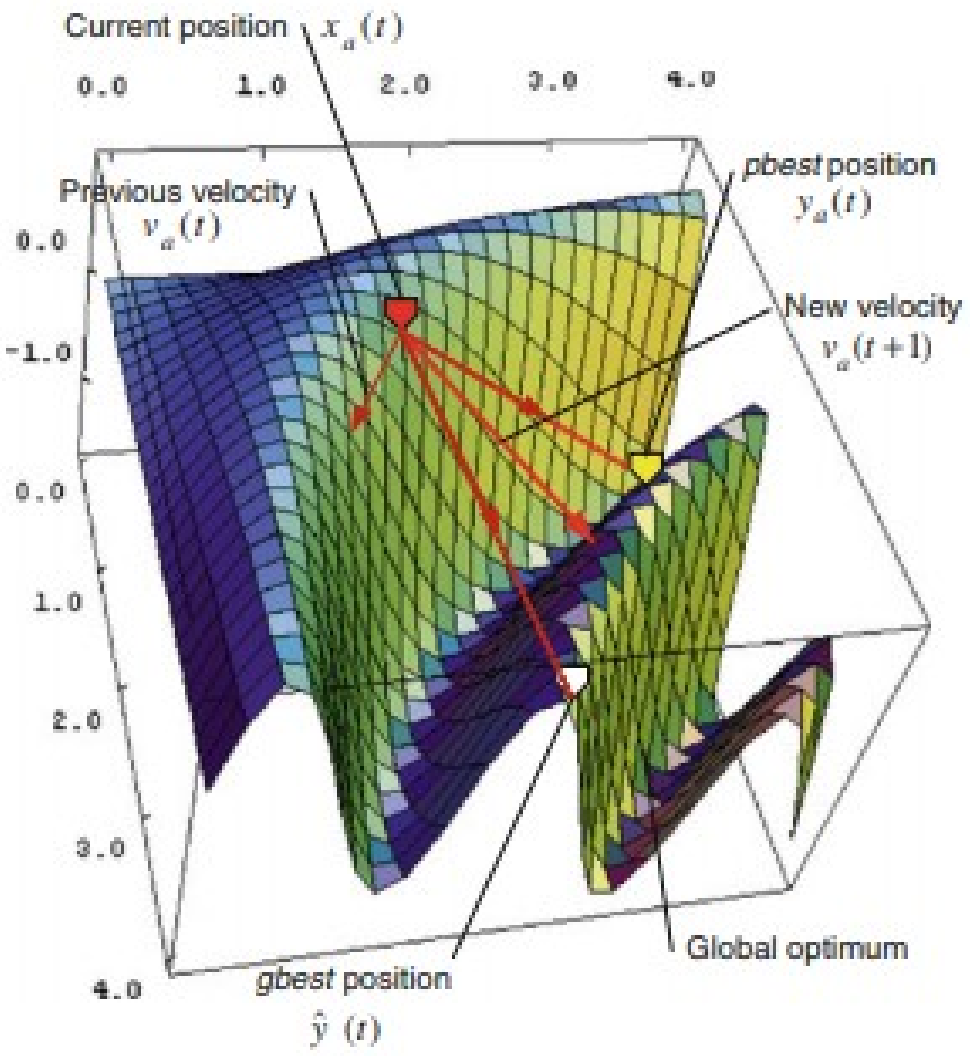


Figure 3.1: Illustration of the basic velocity update mechanism in PSO.

$x_{a,j}(t)$: jth dimensional component of the position of particle a, at time t.
$v_{a,j}(t)$: jth dimensional component of the velocity of particle a, at time t.
$y_{a,j}(t)$: jth dimensional component of the personal best (*pbest*) of particle a, at time t.
$\widehat{y}_j(t)$: jth dimensional component of the global best position of swarm at

time t.

3.6 PSO over other optimization algorithms

Over the ages, nature has constantly been a rich source of inspiration for science, with much still to discover about and learn from. Swarm Intelligence (SI), a major branch of artificial intelligence, was rendered to model the collective behavior of social swarms in nature. Ultimately, Particle Swarm Optimization algorithm (PSO) is arguably one of the most popular SI paradigms. Over the past two decades, PSO has been applied successfully, with good return as well, in a wide variety of fields of science and technology with a wider range of complex optimization problems, thereby occupying a prominent position in the optimization field. Through in- depth studies, a number of problems with the algorithm have been detected and identified; e.g., issues regarding convergence, diversity, and stability. Consequently, since its birth in the mid-1990s, PSO has witnessed a myriad of enhancements, extensions, and variants in various aspects of the algorithm, specifically after the twentieth century, and the related research has therefore now reached an impressive state. The main advantages of the PSO algorithm are summarized as:

1. Simple concept
2. Easy implementation
3. Robustness to control parameters, and computational efficiency when compared with mathematical algorithm and other heuristic optimization techniques [11].

Also, in modern sciences like Artificial Intelligence, most of complex problems are solved by inspiring from narture, which makes PSO the best algorithm to apply for objects like drones since this algorithm is inspired from the movements of flock of birds, which makes it a good choice in this field[1].

3.7 Basic PSO Algorithm

Particle swarm optimization (PSO) is a population-based search algorithm and searches in parallel using a group of particles similar to other AI-based heuristic optimization techniques. The original PSO suggested by Kennedy

and Eberhart is based on the analogy of swarm of bird and school of fish [12]. Each particle in PSO makes its decision using its own experience and its neighbor's experiences for evolution. That is, particles approach to the optimum through its present velocity, previous experience, and the best experience of its neighbors. The main advantages of the PSO algorithm are summarized as: simple concept, easy implementation, robustness to control parameters, and computational efficiency when compared with mathematical algorithm and other heuristic optimization techniques[41]. In a physical n-dimensional search space, the position and velocity of particle-i are represented as the vectors: $X_i = (x_{i1}...x_{in})$ and $V_i = (v_{i1}...v_{in})$ In the PSO algorithm, let: $P - best(ti) = (x_{i1}^{Pbest}...x_{in}^{Pbest}$ and $G - best(ti) = (x_{i1}^{Gbest}...x_{in}^{Gbest}$. be the best position of particle i and its neighbors' best position so far, respectively. The modified velocity and position of each particle can be calculated using the current velocity and the distance from Pbesti to Gbest as follows[15]:

$$V_i^{k+1} = \omega * V_i^k + c_1 * r_1 * (Pbest_i^k - X_i^k) + c_2 * r_2 * (Gbest_t^k - X_i^k) \quad (3.1)$$

$$X^{k+1} = X_i^k + V_i^k + 1 \quad (3.2)$$

Where:
V_i^k: velocity of particle i at iteration k
ω: inertia weight factor
c_1, c_2: acceleration coefficients
r_1, r_2: random numbers between 0 and 1
X_j^k: position of particle i at iteration k
$Pbest_t^k$: best position of particle i until iteration k
$Gbest_i^k$: best position of the group until iteration k.

In the PSO world, there exist global and local PSO versions. Instead of learning from the personal best and the best position achieved so far by the whole population, in the local version of PSO, each particle's velocity is adjusted according to its personal best and the best performance achieved so far within its neighborhood. Focusing on improving the local version of PSO, different neighbourhood structures are proposed and discussed[41].

3.8 Some PSO variants

The first set of improvements has been proposed for the problem-dependent performance of PSO due to its strong parameter dependency. There are mainly two types of approaches: The first one is through self-adaptation, which has been applied to PSO by Kenji Yasuda[45], Miqin Zhang[47], Yuhui Shi[46] and Eberhart[44]. The other approach is via performing hybrid techniques, which are employed along with PSO by Angeline, Reynolds, Higashi and Iba, Esquivel and Coello, and many others. Some of the other improved PSO algorithms we can find:

3.8.1 Attractive and Repulsive PSO (ARPSO)

Attractive and Repulsive PSO (ARPSO) proposed[18] alternates between attraction and repulsion phases. During attraction, ARPSO allows fast information flow between particles causing a low diversity but a better convergence to the solution. It is reported that 95% fitness improvements can be obtained within this phase. In the repulsion phase, the particles are pushed away from the GB solution so far achieved to increase diversity. ARPSO exhibits a higher performance compared to both PSO and GA.

3.8.2 Guaranteed Convergence PSO (GCPSO)

In this algorithm, the velocity of the *gbest* particle will only depend on the memory term since $xgbest = ygbest = \widehat{y}$. To address this problem Van den Bergh introduced a new PSO variant, the PSO with guaranteed convergence, (GCPSO)[19]. In GCPSO, a different velocity update equation is used for the gbest particle based on two threshold values that can be adaptively set during the process. It is claimed that GCPSO usually performs better than the bPSO when applied to unimodal functions and comparable for multimodal problems; however, due to its fast rate of convergence, GCPSO can be more likely to trap to a local optimum with a guaranteed convergence, whereas the bPSO may not. Based on GCPSO, Van den Bergh proposed the Multi-start PSO (MPSO)[19], which repeatedly runs GCPSO over randomized particles and stores the (local) optimum at each iteration. Yet, similar to basic PSO and many of its variants, the performance still degrades significantly as the dimension of the search space increases.

3.8.3 Self-Organized Criticality PSO (SOCPSO)

In another approach, Lovberg and Krink presented Self Organized Criticality (SOC) PSO[20][21]. The criticality measures the proximity of particles, so that the particles that are too close to each other can be relocated in the search space to improve the diversity of the swarm. They propose two types of relocation: The first one is random initialization and the second one is random displacement of particles further in the search space. SOC PSO outperformed basic PSO only in one out of four cases.

3.9 Multiswarms

PSO was initially proposed as an optimization technique for static environments; however, many real problems are dynamic, meaning that the environment and the characteristics of the global optimum can change in time. Therefore, such problems require systematic re-optimizations due to system and/or environmental changes. Even though it is possible to handle such dynamic problems as a series of individual processes via restarting the optimization algorithm after each change, this may lead to a significant loss of useful information, especially when the change is not too drastic. The main problem of using the basic PSO algorithm in a dynamic environment is that eventually the swarm will converge to a single peak—whether global or local. When another peak becomes the global maximum because of an environmental change, it is likely that the particles keep moving near the peak to which the swarm has converged earlier, and thus they cannot find the new global maximum. Blackwell and Branke have addressed this problem in [22] and [23] by introducing multiswarms that are actually separate PSO processes. Each particle is now a member of one of the swarms only and it is unaware of other swarms. This is one of the main differences compared to "Tribes", which otherwise is another good example of multiswarms. The main idea is that each swarm can converge to a separate peak. Swarms interact only by mutual repulsion that keeps them from converging to the same peak. For a single swarm, it is essential to maintain enough diversity, so that the swarm can track small location changes of the peak to which it is converging. For this purpose Blackwell and Branke introduced charged and quantum swarms, which are analogs to an atom having a nucleus and charged particles randomly orbiting it. The particles in the nucleus take care

of the fine-tuning of the result while the charged particles are responsible of detecting the position changes. However, it is clear that, instead of charged or quantum swarms, some other method can also be used to ensure sufficient diversity among particles of a single swarm, so that the peak can be tracked despite of small location changes. As one might expect, the best results are achieved when the number of swarms is set equal to the number of peaks. However, it is then required that the number of peaks is known beforehand. In [24], Blackwell presents self-adapting multiswarms, which can be created or removed during the PSO process, and therefore it is not necessary to fix the number swarms beforehand. The repulsion between swarms is realized by simply reinitializing the worse of two swarms if they move within a certain range from each other. Using physical repulsion could lead to equilibrium, where swarm repulsion prevents both swarms from getting close to a peak.

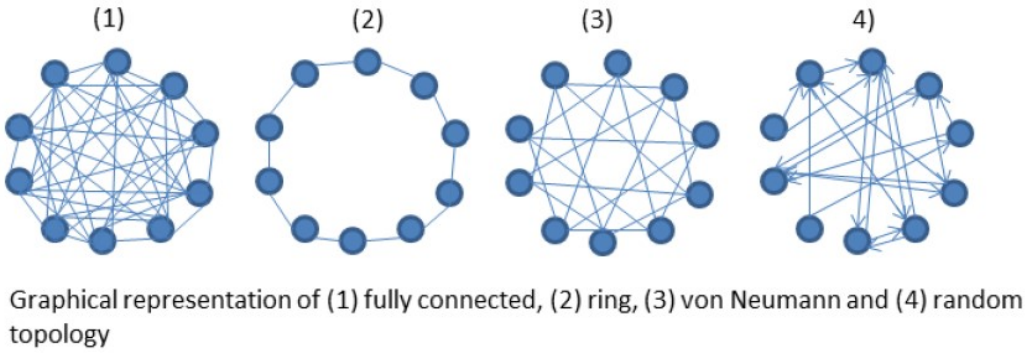


Figure 3.2: Static Neighborhood Topologies.

3.10 Geometrical illustration of PSO

The PSO algorithm begins by initializing the population first. The second step is calculating the fitness values of each particle, followed by updating individual and global bests, and later, the velocity and the position of the particles get updated. The second to fourth steps get repeated until the termination condition is satisfied. The update velocity for particles consist of three components in equation (3.1). Consider a movement of a single particle in a two dimensional search space.

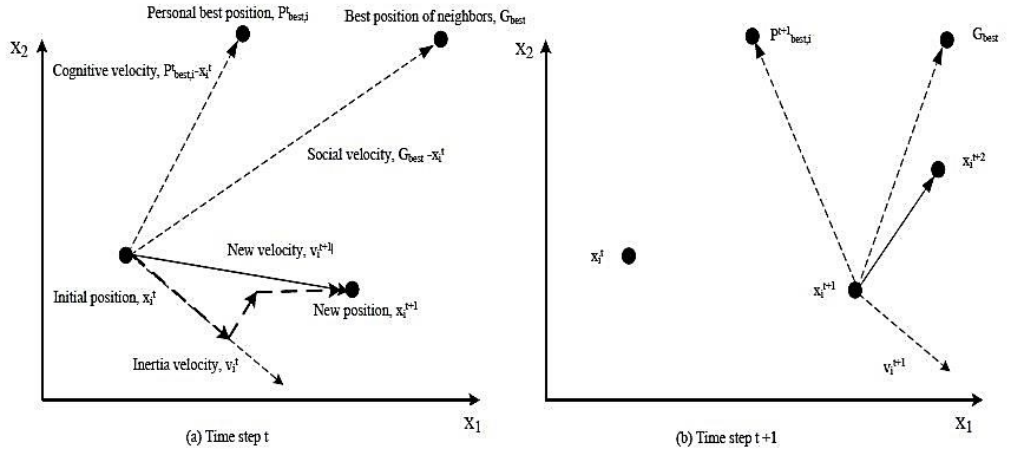


Figure 3.3: velocity and position update for a particle in a two-dimensional search space.

Figure 3.3 illustrates how the three velocity components contribute to move the particle towards the global best position at time steps and respectively.

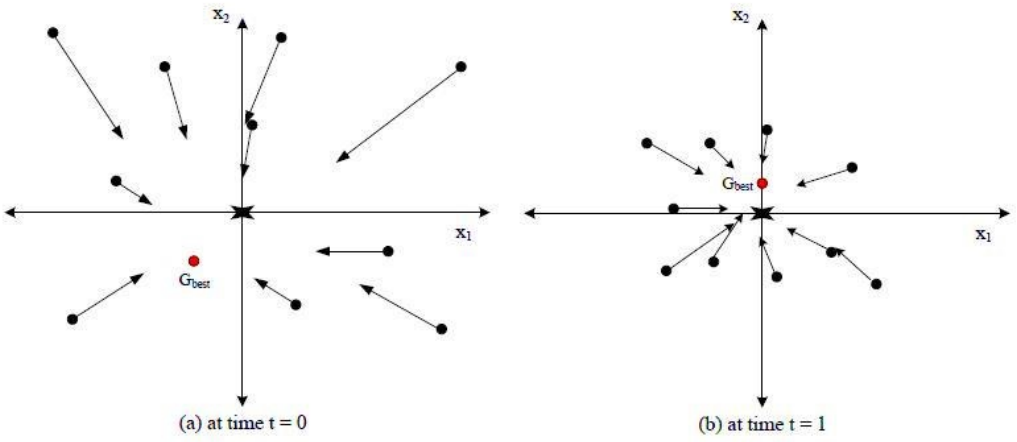


Figure 3.4: velocity and position update for a particle in a two-dimensional search space.

Figure 3.4 shows the position updates for more than one particle in a two dimensional search space and this figure illustrates the gbest PSO. The

optimum position is denoted by the symbol *. Figure 3.4.1 shows the initial position of all particles with the global best position. The cognitive component is zero at t = 0 and all particles are only attracted toward the best position by the social component. Here the global best position does not change. Figure 3.4.2 shows the new positions of all particles and a new global best position after the first iteration i.e. at t = 1.

3.11 Advantages and Disadvantages of PSO

It's said that A PSO is considered as one of the most powerful methods for resolving the non-smooth global optimization problems and has many key advantages as follows[15]:

1. PSO is a derivative-free technique just like as other heuristic optimization techniques.

2. It is easy to implementation, so it can be applied both in scientific research and engineering problems.

3. PSO is less sensitivity to the nature of the objective function compared to the conventional mathematical approaches and other heuristic methods.

4. PSO techniques can generate high-quality solutions within shorter calculation time and stable convergence characteristics than other stochastic methods.

5. The calculation in PSO algorithm is very simple.

6. It is conceptually very simple

While there are some disadvantages of the PSO algorithm[13]:

1. The method easily suffers from the partial optimism, which causes the less exact at the regulation of its speed and the direction.

2. Problems with non-coordinate system (for instance, in the energy field).

3.12 Path Planning using Particle Swarm Optimization

path and deterministic search algorithms were used to find the very shortest path. The definition of the problem has since evolved and the best path is now associated with the path that minimizes the distance travelled, the average altitude, the fuel consumption, the radar exposure, etc. These are a few examples of the factors to be considered and clearly show that the complexity of the problem has grown. To cope with this complexity, researchers have slowly moved from using deterministic algorithms to using nondeterministic algorithms[18].

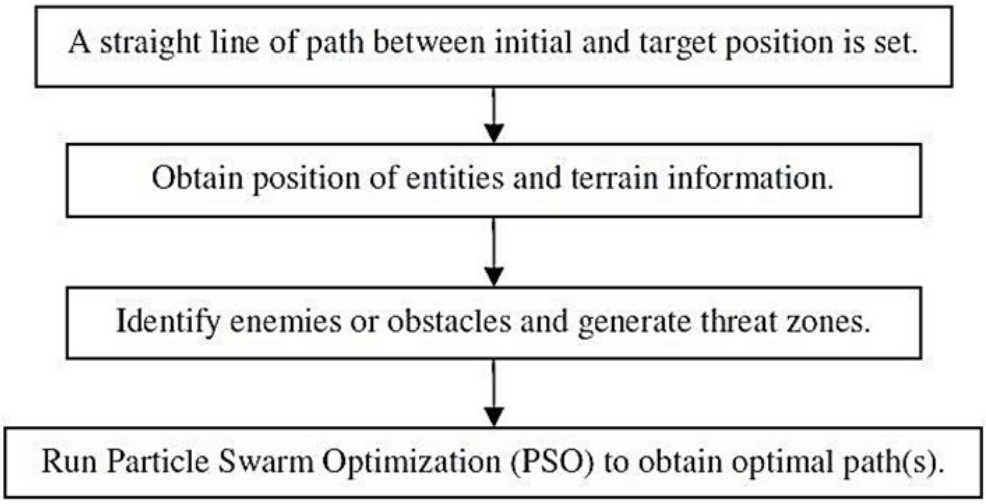


Figure 3.5: Flowchart of the path planning process using Particle Swarm Optimization.

Fig 3.5 shows the process flowchart of the path planning using PSO. It shows that the algorithm consists of 3 basic phases before the general algorithm of PSO. The path planning process begins with identifying a target location for a specific UAV. Once the current and target positions are defined, this becomes the initial solution of the problem. From this initial solution, a search space is defined to scan and locate other UAVs within range and identify possible threats. The size of the search space is left open to the

user's judgment, setting it too large will incur a longer computation time, while having a search space that is too small might cause some UAVs to be unaccounted for. Once position data of the UAVs within range are obtained, enemy entities are singled out and a 3D threat zone is generated for each of them. A threat zone is defined as a sphere (a hemisphere for ground vehicles) of radius R (user defined) surrounding the obstacle that the path needs to avoid. Threat zones are also generated for non-enemy (friendly) entities to avoid collision, but with a smaller radius[19].

3.13 Conclusion

This chapter discussed the basic Particle Swarm Optimization algorithm, geometrical and mathematical explanation of PSO, particles' movement and the velocity update in the search space, the acceleration coefficients and particles' neighborhood topologies. It also discussed the possibility of employing these theory to solve the path planning problems.

Chapter 4

PSO characteristics analysis and application

4.1 Introduction

Recently, the attention of some scientists turned to imitate the behavior of birds in a fleet in order to make an autonomous flying formation as we mentioned before, some solutions use leader, and in these approaches, there are various flight training control strategies : Leader-follower (hierarchical approach), Virtual Leader and control based on behavior (decentralized approach), the method considered in our approach consists in replacing the head of the formation of the leader-follower approach by a virtual leader. All training entities receive the path of the mission which is considered as the virtual leader itself. A major disadvantage of the classical leader-follower strategy is that the risk of collision between UAVs increases,because there is no return (feedback) training. In our approach, each UAV also receives information from its neighbors. The PSO algorithm includes some tuning parameters that greatly influence the algorithm performance,often stated as the exploration–exploitation tradeoff: Exploration is the ability to test various regions in the problem space in order to locate a good optimum, hopefully the global one. Exploitation is the ability to concentrate the search around a promising candidate solution in order to locate the optimum precisely. Despite recent research efforts, the selection of the algorithm parameters remains empirical to a large extent. A complete theoretical analysis of the algorithm has been given by Clerc and Kennedy .Based on this analysis, the

authors derived a reasonable set of tuning parameters, as confirmed by [18] and other researchers.The reference contains a good deal of mathematical complexity, however, and deriving from it simple user-oriented guidelines for the parameter selection in a specific problem is not straightforward[27].
PSO made use of huge memory to keep both the local and global solution found. In this project, the UAVs did not need to remember its past experience because the movement of bird cannot be predicted.

4.2 UAV Assumptions

- UAV swarms are equipped with short-range wireless communication devices. Any pair of UAVs separated a distance smaller than range restrain will be able to establish a communication link with each other.
- UAV swarms are equipped with optical sensors which have the fixed detection range and circular projection on the ground.
- Two common types of UAV are the rotor and the fixed wing. the rotor UAV can hover, the turn radius is smaller and the maneuverability is better, which is more in line with the mission requirements. The kinetic characteristic of the rotor UAV is considered as dynamic constraint in problem modeling[28].

4.3 The method of using particle swarm optimization technique

The PSO is used to obtain an optimal solution using a group of particles where every particle is known minimal knowledge and sensing of the environment.Here in this project the group of UAVs are associated as a swarm of elements in the PSO. Swarm size is the number of coordinating UAVs involved in searching.A maximum number of iterations is the parameter to determine the number of time instances[29].

4.3.1 Standard algorithm

Velocity vector at time instance $(t + 1)$ for a particle i is updated using particle swarm optimization as follows:

$$V_i^{k+1} = \omega^* V_i^k + c_1 * r_1^*(Pbest_i^k - X_i^k) + c_2 * r_2^*(Gbest^k - X_i^k) \tag{4.1}$$

Note: The symbol * denotes element-by-element vector multiplication.

4.3.2 One-dimensional algorithm

It appears from Eq. (3.2) and (3.3) that each dimension is updated independently from the others. The only link between the dimensions of the problem space is introduced via the objective function, i.e., through the locations of the best positions found so far p1 and p2. Thus, without loss of generality, the algorithm description can be reduced for analysis purposes to the one-dimensional case:

$$V_i^{k+1} = \omega * V_i^k + c_1 * r_1^*(P_1 - X_i^k) + c_2 * r_2^*(P_2 - X_i^k) \tag{4.2}$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \tag{4.3}$$

Where, ω is the coefficient of inertia weight, c1 and c2 two real representing the intensity of attraction or acceleration coefficients and r1 ,r2 two random values between 0 and 1. They are usually selected as uniform random numbers in the range [0, 1].

4.4 PSO Algorithm Parameters

There are some parameters in PSO algorithm that may affect its performance. For any given optimization problem, some of these parameter's values and choices have large impact on the efficiency of the PSO method, and other parameters have small or no effect[30].

The basic PSO parameters are swarm size or number of particles, number of iterations, velocity components, and acceleration coefficients illustrated bellow. In addition, PSO is also influenced by inertia weight, velocity clamping, and velocity constriction.

4.4.1 Swarm size

Historical discussion

In the first paper on PSO Kennedy and Eberhart referred to the zoological studies in which the movements of flocks composed of 15–30 birds were simulated. This seems to be an initial guess as Kennedy and Eberhart use 20 particles in their first PSO tests. In a subsequent study[31] one may find information that the results discussed there were obtained with 20 particles, even though the authors used 10–50 particles for other applications. Also, 20 particles were used by Shi and Eberhart , who introduced an inertia weight into PSO, which is now considered a standard approach (PSO-iw). This initial guess seems to be reproduced in the vast majority of subsequent PSO studies, even though some empirical tests were eventually performed at the edge of the new millennium, as will be discussed below.

In 2004 van der Bergh and Engelbrecht tested a number of early PSO variants on five classical benchmark functions (also in the rotated framework) with the population size set to 10, 15 and 20 particles. The results depended on the specific algorithm, but the basic PSO performed best with 20 particles (the highest tested value). showing that better results are generally obtained with larger swarm sizes. However, as the population size settings were limited to low values, the conclusions from this study are limited too[32].

4.4.2 Number of iterations

The PSO algorithm is an iterative optimization process and repeated iterations will continue until a stopping condition is satisfied. Within one iteration, a particle determines the personal best position, the local or global best position, adjusts the velocity, and a number of function evaluations are performed. Function evaluation means one calculation of the fitness or objective function which computes the optimality of a solution. If n is the total number of particles in the swarm, then n function evaluations are performed at each iteration[33]. The maximum number of iterations is taken as the stopping criterion, and the best, the mean and the worst values of the objective function are presented to compare the performance of all candidates, where the maximum number of iterations is set to 1000 for most of algorithms.

4.4.3 Inertia weight

The inertia weight, denoted by ω, is considered to replace by adjusting the influence of the previous velocities in the process, i.e. it controls the momentum of the particle by weighing the contribution of the previous velocity. The inertia weight "ω" will at every step be multiplied by the velocity at the previous time step, i.e. V_i^k . The inertia weight was first introduced by Shi and Eberhart in 1999 to reduce the velocities over time (or iterations), to control the exploration and exploitation abilities of the swarm, and to converge the swarm more accurately and efficiently.
Inertia component weight $\omega = 1$. In this phase, UAVs haven't encounter with any targets, they search in the initial direction with fixed value of speed. The parameter setting can also show the characteristic of UAVs in this phase which is to try their best in exploration to find targets as soon as possible[25].

4.4.4 Velocity Components

The velocity components are very important for updating particle's velocity. There are three terms of the particle's velocity in equation (3.2):

1. The term V_i^k is called inertia component that provides a memory of the previous flight direction that means movement in the immediate past. This component represents as a momentum which prevents to drastically change the direction of the particles and to bias towards the current direction.

2. The term $c_1 * r_1^*(Pbest_i^k - X_i^k)$ is called cognitive component which measures the performance of the particles i relative to past performances. This component looks like an individual memory of the position that was the best for the particle.

3. The term $c_2 * r_2^*(Gbest^k - X_i^k)$ is called social component which measures the performance of the particles i relative to a group of particles or neighbors. The social component's effect is that each particle flies towards the best position found by the particle's neighborhood[6].

4.4.5 Setting parameters

The algorithm includes several setting parameters to act on the compromise Exploration - Exploitation. To simplify the study, we consider the determin-

istic version of the algorithm, where random numbers are replaced by their average values $\frac{1}{2}$ With simplifications, the algorithm can be written as:

$$V_i^{k+1} = \omega * V_i^k + c * (P^k - X_i^k) \tag{4.4}$$

$$X_i^{k+1} = X_i^k + v_i^{k+1} \tag{4.5}$$

where:

$$P(k) = \frac{c_1 * p_1 + c_2 * p_2}{c_1 + c_2} \text{ and } c = \frac{c_1 + c_2}{2} \tag{4.6}$$

To make a dynamic analysis of the algorithm, the equations (4.4) and (4.5) are rewritten in the matrix form:

$$V_i^{k+1} = -c * X_i^k + \omega * V_i^k + c * P^k \tag{4.7}$$

$$X_i^{k+1} = X_i^k + [-c * X_i^k + \omega * V_i^k + c * P^k] \tag{4.8}$$

Then:

$$V_i^{k+1} = -c * X_i^k + \omega * V_i^k + c * P^k \tag{4.9}$$

$$X_i^{k+1} = (1 - c) * X_i^k + \omega * V_i^k + c * P^k \tag{4.10}$$

The equation of the algorithm can be written in the following matrix form:

$$\begin{pmatrix} X_i(k+1) \\ V_i(k+1) \end{pmatrix} = A * \begin{pmatrix} X_i(k) \\ V_i(k) \end{pmatrix} + B * P(k) \tag{4.11}$$

Where: $\begin{pmatrix} X_i(k+1) \\ V_i(k+1) \end{pmatrix}$ is the state of the system, consisting of the position of the particle and its velocity, P is the system input,
$A = \begin{bmatrix} 1-c & \omega \\ -c & \omega \end{bmatrix}$ is the dynamical matrix, and $B = \begin{bmatrix} c \\ c \end{bmatrix}$ is the input matrix.
The equilibrium point of the system is such that the particle is positioned in P. $X_i(k)$and$P(k)$ has a zero velocity $V_i(k) = 0$.
Behaviors of the particle depend on the eigenvalues of the matrix A are the solutions of: $det(\lambda * I - A) = 0$.

$$\lambda^2 - (\omega - c + 1) * \lambda + \omega = 0 \tag{4.12}$$

The behavior and the convergence of the algorithm depend on the parameters ω and c. The analysis of the equation (4.12) leads to determine the area of convergence of the PSO according to their values[25].

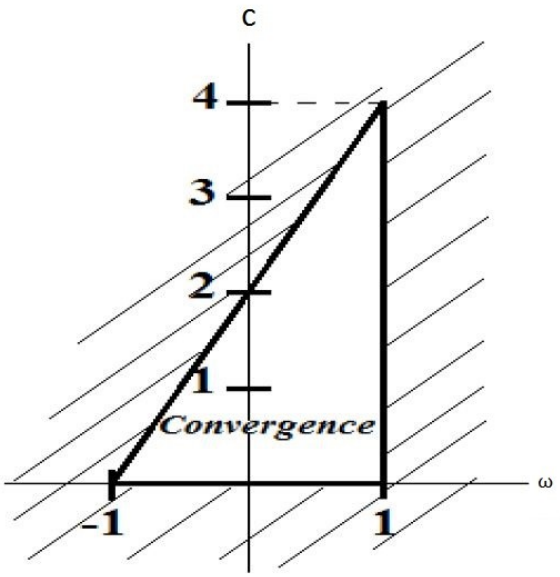


Figure 4.1: PSO convergence area.

Figure 4.1 shows the values (plain area) that should have the parameters ω and c. for a convergence of the algorithm.

4.5 Organigram of the PSO

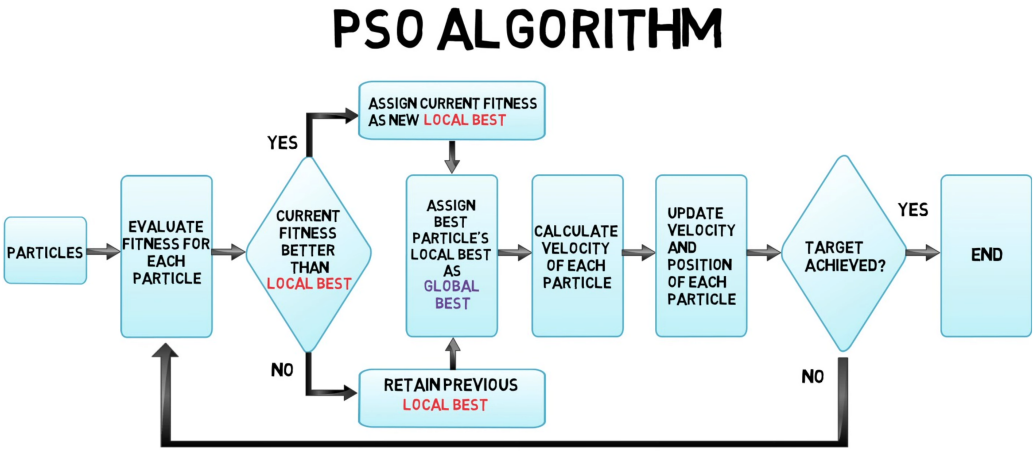


Figure 4.2: Organigram of PSO algorithm.

4.5.1 Block details

Swarm initialization

The UAVs are randomly scattered in the initial step of swarm production without a specific criterion. Given subsequent iterations in the program , PSO begins with a randomly generated swarm of dimensional UAVs simplified with real-valued position vectors representing the initial candidate solutions by initializing each particle's (UAV) position x_i^0 (at iteration $k = 0$) to a random position in the search space. And there is a possibility of an instability problem in the whole swarm, therefore we need another information that must be added in the algorithm which is personal fitness. In PSO algorithm, initialization of the swarm is very important because proper initialization may control the exploration and exploitation trade-off in the search space more efficiently and find the better result.

Fitness evaluation

The swarm particles are evaluated at the end of each iteration for their own fitness values as well as global optimum fitness. It is assumed that each UAV

i has a unique fitness value $f(X_i^k)$ at each iteration k , which is calculated through an objective function evaluation. PSO memorizes the personal best solution (candidate global best solution) which every particle has ever met until the current iteration k.

Velocity initialization (calculate velocity)

All particles are frequently moving through the search space with a velocity (step size) that reflects particles experiential knowledge as well as socially exchanged information about the promising areas visited in the search space, thereby driving the optimization process to the most feasible regions. Similarly to position initialization, without this velocity information in the searching space the process will be prone to explode and particles' positions change rapidly.

Velocity and position updating

At each current iteration, $k + 1$ for example, the k^th UAV's velocity V_k is first regulated by sending its parameters soaring in the positive or negative direction, contingent on the convergence of the current position, attracting the particle towards positions in the search space that are known to be good from past personal experience, as well as from the ex- perience of other particles in the neighborhood of the particle. In fact, the original PSO was implemented for two different neighborhood topologies, global best ($Gbest$) PSO and local best ($Lbest$) PSO as we mentioned before.

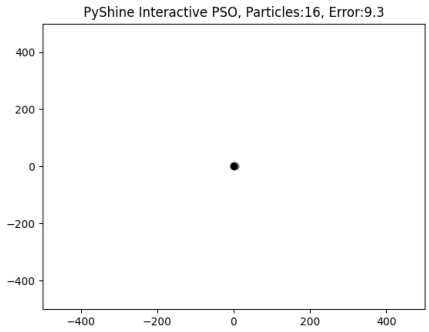
4.6 Primary results

In this section, we'll discuss the primary results of the program by showing the implementation of the algorithm using Python as a programming language.
Note: The parameters values in this try are:

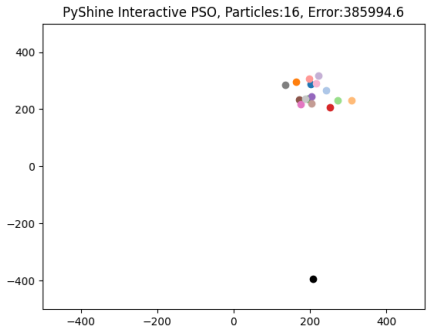
- $\omega = 0.5$
- $c_1 = 1$
- $c_2 = 2$

- $\gamma = 0.0001$

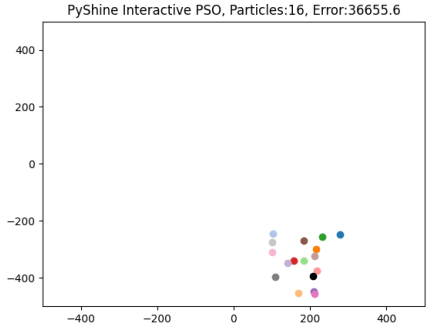
after executing we get the following results:



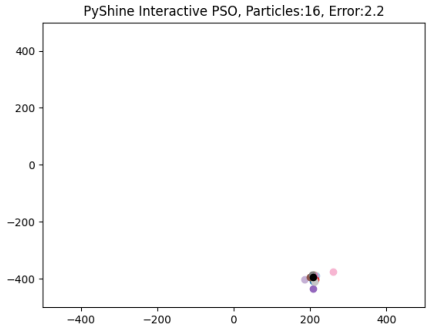
(a) Initial position



(b) Changing the target's position



(c) Particles looking for the Global best position



(d) Particles convergence to the target

Figure 4.3: The results of the PSO program

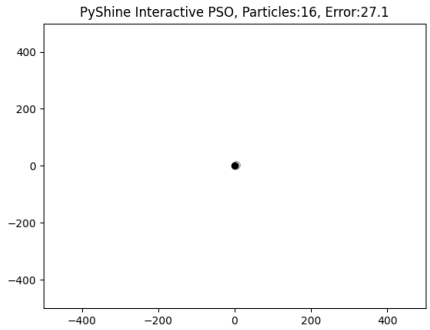
4.7 Results discussion

We can say from the first view that the algorithm works as we want with the previous parameters; the particles converge to the target as we expected, but during the simulation we saw that the particles move slowly and they were almost stuck to each other. The main reason to this is because of the parameters values that we chose. First, we have the $WeightInertia\omega = 0.5$,

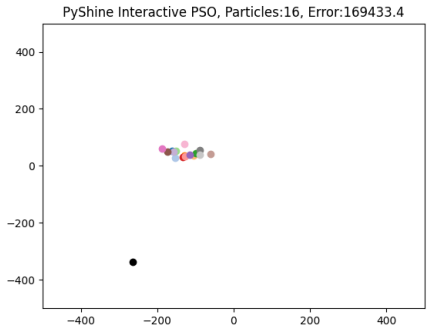
we can see that this value is little bit small which will lead to the slow movement of the particles since it's a main factor in the velocity update, so increasing this factor to an acceptable value will make a good difference on the performance of the algorithm. Also the coefficients c_1 and c_2, those two doesn't affect the particles a lot since they're in the convergence area, but every value has it's influence on the particles even thought sometimes it's not remarkable, for that we will give those two there standard values used by most of the researchers which is 2.Finally the last and the most sensible factor γ, the factor has a big influence on the algorithm such that a small change of it's value can make the particles diverge completely, it's similar to factor ω, which means it affects the velocity of the particles but unlike ω it's much more effective. So for this purpose we made a lot of tries to find the domain of convergence for this factor and we found that it vary from 0.0001 to 0.00019.

4.8 Improved results

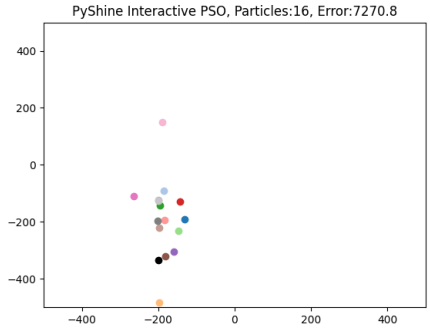
By modifying the previous parameters to: $\omega = 0.6$ $c_1 = c_2 = 2$ $\gamma = 0.00013$ we get the following results:



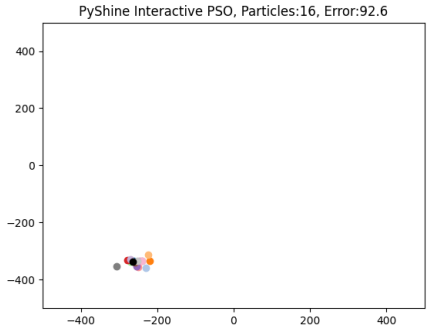
(a) Initial position



(b) Changing the target's position



(c) Particles looking for the Global best position



(d) Particles convergence to the target

Figure 4.4: The improved results of the PSO program

4.8.1 Comments

In this simulation, the convergence speed is reasonable and in the order of 5s. The simulation is also repeated 15 times with different initial conditions and different parameters within their affinity domain. The results are equivalent to the case illustrated in the above example. The approach considered in our simulation is able to reproduce without difficulty geometric configurations and flight training, but we need to ensure more the constraint of the anti collision.

4.9 Discussion of the results

From this results, we can see that there's a remarkable improvement in the algorithm due to the change of the previous parameters, we saw a relatively fast convergence by the particles to the target which means the algorithm took less time, and also we saw that the particles were more free than the previous time, that can be translated to less collision between the particles. However, we still need to improve the algorithm more, until we get the least of collisions between particles. For that problem to be solved we need a more improved technique of the PSO.

4.10 Multi-Objective Particle Swarm Optimization (MOPSO)

In multi-objective problems, we can distinguish two fundamental approaches for designing PSO algorithms (Reyes-Sierra and Coello, 2006a). The first approach consists of algorithms that consider each objective function separately. In these approaches, each particle is evaluated only for one objective function at a time, and the determination of the best positions is performed similarly to the single-objective optimization case. The main challenge in such cases is the proper manipulation of the information coming from each objective function in order to guide the particles towards Pareto optimal solutions. The second approach consists of algorithms that evaluate all objective functions for each particle, and, based on the concept of Pareto optimality, they produce non-dominated best positions (often called leaders) that are used to guide the particles. In these approaches, the determination of leaders is not straightforward, since there can be many non-dominated solutions in the neighborhood of a particle, but only one is usually selected to participate in the velocity update. In the aforementioned approaches, the problem of maintaining the detected Pareto optimal solutions must be addressed. The most trivial solution would be to store non-dominated solutions as the particles' best positions. However, this choice is not always valid, since the desirable size of the Pareto front may exceed the swarm size. Moreover, two non-dominated solutions are equally good, arising questions regarding the selection of the one that will be used as the best position of a particle. The size problem can be addressed by using an additional set, called the external archive, for storing the non-dominated solutions discov-

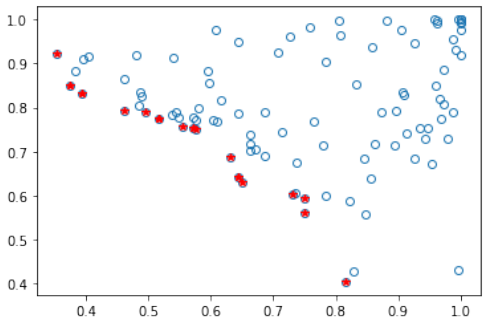
ered during search, while the problem of selection of the most proper archive member depends on the approach. Nevertheless, an external archive has also bounded size, thereby making unavoidable the imposition of rules regarding the replacement of existing solutions with new ones[34].

4.11 MOPSO Algorithm

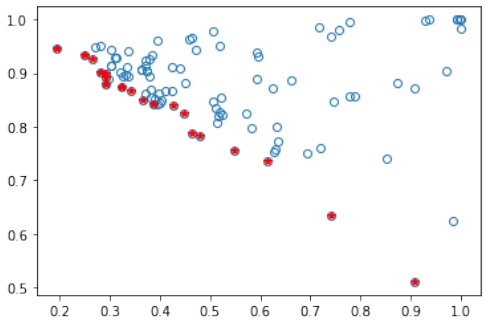
```
Begin
    Initialize swarm, velocities and best positions
    Initialize external archive (initially empty)
    While (stopping criterion not satisfied) Do

        For each particle
                Select a member of the external
                archive (if needed)
                Update velocity and position
                Evaluate new position
                Update best position and external
                archive
        End For
    End While
End
```

Figure 4.5: Multi-Objective PSO

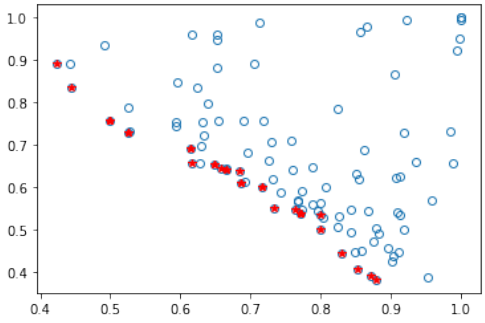
By Executing the algorithm using Python as a programming language we get the following results:



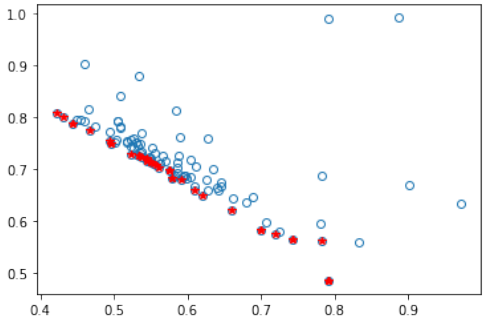
(a) Initial phase



(b) Searching the targets



(c) Particles starting to converge to the targets



(d) Final results

Figure 4.6: Multi-Objective PSO results

4.12 Discussion of the results

From the results, we can say that the particles are well positioned around the targets solving the problem of collision between them. We also solved the problem of the weight inertia and the other coefficients by evaluating them in every iteration making it stop at the best value of each parameter that fits the particles.

4.13 Conclusion

This chapter presents a swarm UAVs control strategy and it's application based on a multi-agent system.The use of the PSO algorithm allowed us to

have better results in terms of convergence speed and optimal solution. This approach has been tested in the first simulation on a group of 25 UAVs. In our simulations, both the cases of the parameters standing and in motion has been considered. We added a multi-objective subprogram to the main program to solve the collision avoidance problems we encountered in the first simulation. Currently, A future application of control strategy on non-linear complex systems and implementations on real systems are planned. An improvement would be to introduce the dynamics of controlled systems in the optimization algorithms to better find the most appropriate one for the studied systems solutions.

General Conclusion

Future Works and Advanced Topics of PSO

The devlopement works will focus on evaluating PSO theory on benchmarking functions and exploring its capability to solve other complex optimization problems. This algorithm needs to work more on the exploration phase because the development of this phase is associated with a key factor which is increasing the swarm size, but scaling this parameter in our application is limited, especially when it comes to the application of UAVs path planning because it creates swarm stability problems that may lead to severe consequences Thus, as developers of this algorithm, we need to ensure stability between parameters as an essential step that cannot be ignored. And we may find in the series of research published on this theory from 1995 to 2004 and then to this day that we often find the addition of a new coefficient to the velocity equation to ensure more control. Nevertheless, PSO may need to increase the swarm size and number of iterations if the search space increases . In this scenario, parallel implementation is required to effectively reduce the computation time, and hence, improve the scalability of the proposed algorithm for largescale systems.

Basic PSO

In the basic PSO, one of the major problems is lack of diversity when particles start to converge to the same point. . . .

To prevent this problem of the basic PSO, several methods have been developed to continually inject randomness, or chaos, into the swarm. These types of methods are called the Multi-start (or restart) Particle Swarm Optimizer (MSPSO).

Multi-Start PSO (MSPSO)

The Multi-start method is a global search algorithm and has as the main objective to increase diversity, so that larger parts of the search space are explored . It is important to remember that continual injection of random positions will cause the swarm never to reach an equilibrium state that is why, in this algorithm, the amount of chaos reduces over time. Kennedy and Eberhart first introduced the advantages of randomly reinitializing particles and referred to as craziness.[06] In view of the time difference of 25 years, it is not a long period, and the theory has received an amazing development during a quarter of a century, but it still lacks the methods of its application because these methods change with the change in the field of their applications. This is what we, as students and researchers, are required to do in the field of aeronautics.

PSO convergence

It is interesting to note that global convergence can be proved without requiring the local part to be a guaranteed local search algorithm. All that is required is that the local part of the algorithm must be able to satisfy some termination criterion not necessarily convergence onto a local minimiser which means that it is not necessary for all drones to converge at the same rate, but rather what matters to us is the final method of convergence of the swarm while maintaining the condition of the safety distance between the drones.

Conclusion

We have presented a new application algorithm, the particle swarm optimization PSO, to solve the problem of optimal search for a static target using UAVs. Through extensive simulations, as described in this thesis, we can say that PSO presents better performance than other state-of the-art heuristic algorithms in most search scenarios and is suitable for practical UAV search operations. The rationale for the success of PSO lies in the exploitation and learning of information taken from the external environment and nearby individuals just as birds do in the sky and fish in the sea that prevents the algorithm from generating invalid paths during the searching process so that

it can avoid the need for re-initialization, and as such, to accelerate the convergence. The PSO algorithm has some problems that ought to be resolved. Therefore, the future works on the PSO algorithm will probably concentrate on the following:

- Find a particular PSO algorithm which can be expected to provide better performance.
- Combine the PSO algorithm with other optimization methods to improve the accuracy.
- The evolution of drone technology has to parallel the evolution of PSO's algorithms so that we can ensure feasibility.

Appendix A

Appendix A

List of abbreviations

UAV: Unmanned Aerial Vehicle.
RPA: Remotely Piloted Aircraft.
RPV: Remotely Piloted Vehicle.
UAS: Unmanned Aircraft Systems.
USA: United States of America.
AUV: Autonomous Underwater Vehicle.
MRS: Multi-Robot Systems.
SI: Swarm Intelligence.
PSO: Particle Swarm Optimization.
ACO : Ant Colony Optimization.
EAs : Evolutionary Algorithms.
WOA: Whale Optimization Algorithm.
ABC : Artificial Bee Colony.
ARPSO: Attractive and Repulsive Particle Swarm Optimization.
GCPSO : Guaranteed Convergence Particle Swarm Optimization.
MPSO: Multi-start Particle Swarm Optimization.
SOCPSO: Self-Organized Criticality Particle Swarm Optimization.
MOPSO: Multi-Objective Particle Swarm Optimization.

Appendix B

Appendix B

Symbols

$X_{a,j}(t)$: j_th dimensional component of the position of particle a, at time t.
$Va, j(t)$: j_th dimensional component of the velocity of particle a, at time t.
$Y_{a,j}(t)$: j_th dimensional component of the personal best (pbest) of particle a, at time t.
$y_j(t)$: j_th dimensional component of the global best position of swarm at time t.
V_i^k : velocity of particle i at iteration k.
$Gbest$: global best.
$Lbest$: local best.
ω: inertia weight factor.
c_1, c_2: acceleration coefficients which are used to level the contribution of the cognitive and social components respectively.
$r1, r2$: random numbers between 0 and 1.
X_j^k: position of particle i at iteration k.
$Pbest_i^k$: best position of particle i until iteration k.
$Gbest^k$: best position of the group until iteration k.
* : Denoted the optimum position.
t : Denotes time or time steps.
A: the dynamical matrix.
B: the input matrix.
P: the system input.
$f\ (X_i^k)$: fitness value of particle i at iteration k.

Y: fitness value included in the program.

Bibliography

[1] Kiranyaz S., Ince T., Gabbouj M. *Multi-dimensional Particle Swarm Optimization*. Multidimensional Particle Swarm Optimization for Machine Learning and Pattern Recognition. Adaptation, Learning, and Optimization, vol 15. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-37846-1_4, 2014.

[2] *Drones*. https://builtin.com/drones.

[3] Rajalakshmi Krishnamurthi Anand Nayyar Aboul Ella Hassanien, Development and Future of Internet of Drones (IoD),
`Insights, Trends and Road Ahead` springer September 2020

[4] 'Drone swarms' are coming, and they are the future of wars in the air *https://theprint.in/defence/drone-swarms*.

[5] Utilizing Particle Systems for Strand Animation *Kungliga Tekniska högskolan, Degree Project in Computer Science, First Level - DD143X*. May 20, 2012

[6] Satyobroto Talukder "Mathematical Modelling and Applications of Particle Swarm Optimization", Master's Thesis. February 2011

[7] A BRIEF HISTORY OF DRONES *https://www.iwm.org.uk/history/a-brief-history-of-drones*

[8] S. Kiranyaz et al. *Multidimensional particle swarm optimization for machine learning and pattern recognition* Springer - 2014

[9] Bonabeau, E., Dorigo, M., Theraulaz , *G: Swarm Intelligence: From Natu ral to Artificial Systems, Number 1* Oxford University Press, Oxford (1999)

[10] Gupta, S., Abderazek, H., Yıldız, B.S., Yildiz, A.R., Mirjalili, S., Sait,S.M *Comparison of Metaheuristic Optimization Algorithms for Solving Constrained Mechanical Design Optimization Problems* Expert Systems with Applications - 2021

[11] Kwang Y. Lee and Jong-Bae Park, "Application of Particle Swarm Optimization to Economic Dispatch Problem: Advantages and Disadvantages," IEEE, 2010.

[12] D Karaboga, B Basturk *Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems* International fuzzy systems association world - 2007

[13] M Dorigo, M Birattari, T Stutzle *Ant colony optimization* IEEE Computational Intelligence Magazine - 2006

[14] S Mirjalili, A Lewis *The whale optimization algorithm* Advances in engineering software, 2016

[15] KY Lee, JB Park *Application of particle swarm optimization to economic dispatch problem: advantages and disadvantages* IEEE PES Power Systems Conference and Expositio- 2006

[16] J. Kennedy, R. Eberhart *Particle swarm optimization* Proceedings of ICNN'95 - International Conference on Neural Networks - 1995

[17] V Selvi, R Umarani *Comparative analysis of ant colony and particle swarm optimization techniques* International Journal of Computer Applications - 2010

[18] IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, VOL. 9, NO. 1, FEBRUARY 2013.

[19] Foo, Jung Leng; Knutzon, Jared S.; Oliver, James H.; and Winer, Eliot H., "Three-Dimensional Path Planning of Unmanned Aerial Vehicles Using Particle Swarm Optimization" (2006).

[20] Y Lin, B Bhanu *Evolutionary feature synthesis for object recognition* IEEE Transactions on Systems - 2005

[21] M Lovbjerg, T Krink *Extending particle swarm optimisers with self-organized criticality* Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)

[22] T Blackwell, J Branke *(2004) Multi-swarm Optimization in Dynamic Environments.* In: Raidl G.R. et al. (eds) Applications of Evolutionary Computing. EvoWorkshops 2004. Lecture Notes in Computer Science, vol 3005. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-24653-4-50

[23] T. Blackwell, J. Branke *Multiswarms, exclusion, and anti-convergence in dynamic environments* IEEE Transactions on Evolutionary Computation (Volume: 10, Issue: 4, Aug. 2006)

[24] Blackwell T *Particle swarm optimization in dynamic environments* Yang S., Ong YS., Jin Y. (eds) Evolutionary Computation in Dynamic and Uncertain Environments. Studies in Computational Intelligence, vol 51. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-49774-5-2

[25] A Belkadi, L Ciarletta and D Theilliol *Particle swarm optimization method for the control of a fleet of Unmanned Aerial Vehicles* Journal of Physics: Conference Series, Volume 659, 12th European Workshop on Advanced Control and Diagnosis (ACD 2015) 19–20 November 2015, Pilsen, Czech Republic

[26] R.C. Eberhart, Y. Shi *Comparing inertia weights and constriction factors in particle swarm optimization* Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No.00TH8512)

[27] IC Trelea *The particle swarm optimization algorithm: convergence analysis and parameter selection* Information processing letters, 2003 - Elsevier

[28] Y Wang, P Bai, X Liang, W Wang, J Zhang, Q Fu *Reconnaissance mission conducted by UAV swarms based on distributed PSO path planning algorithms* IEEE Access, 2019

[29] Phoebe Chen, Alfredo Cuzzocrea, Xiaoyong Du, Orhun Kara, Ting Liu,Krishna M. Sivalingam, Dominik Slezak, Takashi Washio, and Xi-

aokang Yang *23rd International Computer Symposium, ICS 2018 Yunlin, Taiwan, December 20–22, 2018*

[30] Anthony Carlisle, Gerry Dozier *An off-the-shelf PSO* Workshop Particle Swarm Optimization, Indianapolis, 2001

[31] R. Eberhart, J. Kennedy *A new optimizer using particle swarm theory* MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science

[32] AP Piotrowski, JJ Napiorkowski, AE Piotrowska *Swarm and Evolutionary Computation* 58 (2020) 100718.

[33] Andries P. Engelbrecht *Computational intelligence: an introduction* John Wiley and Sons, 2007, ch. 16, pp. 289-358

[34] Michael N. Vrahatis, Konstantinos Parsopoulos *Multi-objective Optimization in Computational Intelligence: Theory and Practice* Edition: April 2008 Chapter: 2

[35] Saptarshi Sengupta, Sanchita Basak and Richard Alan Peters II *Particle Swarm Optimization: A Survey of Historical and Recent Developments with Hybridization Perspectives*

[36] Lucasfilm Ltd. (1983)

[37] Rosli b.O Path planning for unmanned aerial vehicles using visibility line-based methods. PHD thesis. March 2011.

[38] YongBo Chen · JianQiao Yu · XiaoLong Su · Path Planning for Multi-UAV Formation.2014.

[39] J. Giesbrecht. Global Path Planning for Unmanned Ground Vehicles. 2004.

[40] Waseem Ahmed Kamal. Safe trajectory planning techniques for autonomous air vehicles. PHD thesis. 2005.

[41] J. J. Liang and P. N. Suganthan School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.

[42] Clerc, Yasuda, Zhang, and Shi and Eberhart

[43] James Kennedy (born November 5, 1950) is an American social psychologist, best known as an originator and researcher of particle swarm optimization.

[44] Russell C. Eberhart, an American electrical engineer, best known as the co-developer of particle swarm optimization concept (with James Kennedy (social psychologist)). He is professor of Electrical and Computer Engineering, and adjunct professor of Biomedical Engineering at the Purdue School of Engineering and Technology.

[45] Kenji Yasuda is a Postdoctoral Scholar at Massachusetts Institute of Technology.

[46] Yuhui Shi is a pioneer in particle swarm optimization algorithms, and the developer of brain storm optimization algorithms. He was an electrical engineer from Xi'an Jiaotong-Liverpool University in Suzhou.

[47] Miqin Zhang is a professor of materials science and engineering at the University of Washington.