NTNU – Trondheim
Norwegian University of
Science and Technology

Detecting a Hidden Radio Frequency Transmitter in Noise based on Amplitude using Swarm Intelligence

Jørgen Halvorsen
Nordmoen

Master of Science in Informatics
Submission date: June 2014
Supervisor: Keith Downing, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Abstract

Radio Frequency transmitters are becoming ubiquitous. For instance people often carry around a mobile phone containing several such transmitters. Having the ability to detect and locate such transmitters in noisy environments can therefore be very beneficial in search and rescue and disaster management. By using amplitude measurements, simple hardware can be deployed on small and cheap Unmanned Aerial Vehicles. The use of many cooperative Unmanned Aerial Vehicles should enable rapid search and more precise geolocation of such transmitters.

This thesis explores different search strategies to use on autonomous agents trying to detect a hidden Radio Frequency transmitter. By testing different Swarm Intelligence strategies important insight into the search process is gained. The experiments were run on a simulator to enable rapid testing of different strategies. In addition to testing baseline algorithms from the literature, a new strategy is created, the Call-out strategy.

The work done, shows that few cooperating searchers should prefer exploitation of the transmitted signal over exploration. However, as more agents are employed this focus is shifted over to exploration. In addition, this work show that several simple searchers can perform the task with high performance. This shows that Swarm Intelligence can be used to direct several cooperative searchers in detecting a hidden Radio Frequency transmitter.

Preface

This work started out as a large undertaking which have taken on a narrower scope as time has passed. Because of the different disciplines combined in this work it has at times been a lot to take in. This may have resulted in a longer thesis, but the different concept should be well explained and different domain experts should recognize their fields as they are.

Getting immersed in such different topics of research have been both a challenge and a great learning experience. While part of the challenge have been the different views and different vernacular used about overlapping concepts, these differences have also aided in expanding the horizon, often in different directions, and always with interesting outcomes.

The results may in hindsight look quite obvious, but there was still much useful information in doing the work. Even though the results might not be that groundbreaking this work should aid others in designing future experiments which should expand the knowledge within this very interesting domain.

I would like to thank my advisor Keith Downing for the opportunity to work within this interesting field and his support in letting me explore the different options that have arisen.

Furthermore I would like to extend my thanks to Jonas Moen and Thomas Thoresen which have been invaluable help during this long journey. Without their help this work would never have been possible.

In addition I would like to thank both Sondre Andreas Engebråten and Lars Sundnes Løvlie for their insightful discussions.

I would also like to thank Boye Annfelt Høverstad for his help in making me understand Lévy flight and how to generate it. In addition, Ingebjørg Kåsen deserves a special mention for helping me put together and guide me in the statistics used for this thesis. Her help made the statistics more sound and helped me understand why the rigour was necessary and beneficial.

Lastly I would like to thank NTNU and the faculty at IDI for their work in supporting the students.

Jørgen Halvorsen Nordmoen
Trondheim, June 1, 2014

Contents

List of Figures

Chapter 1

Introduction

In many different settings the process of rapidly collecting relevant information can be very important. In search and rescue gathering information quickly can be the difference between life and death. Environmental disasters require rapid assessment of possible dangers and damages to avoid severe consequences. Information gathered in situations like the above is time sensitive and needs to be gathered in a timely fashion. By using multiple autonomous agents this information gathering can be performed quicker and with greater reliability.

To enable this automatic information gathering there is a need for intelligent systems which can operate with little to no input from humans. By having systems which can react and adapt to unforeseen events, are scalable to new operational requirements and survivable to failures within the system, human involvement can be kept to a minimum.

One specific instance of such information gathering is the detection and location of a Radio Frequency (RF) transmitter. Detecting and locating a RF transmitter is an important endeavour because of its many uses in real world situations. Most people carry around a mobile phone which can work as a radio transmitter. Having the ability to locate such a transmitter can therefore aid in many search and rescue situations.

Amplitude measurements are easy to gather and require relatively inexpensive equipment. Power Difference of Arrival (PDOA) is based on measuring the effect and is very well suited for simple agents trying to locate a transmitter. This simplicity, both in hardware and in implementation, makes amplitude measurements well suited for exploring detection.

In order to detect and locate a radio transmitter there is a need for a mobile platform. Unmanned Aerial Vehicles (UAVs) are becoming a much more available technology used in several civilian contexts. Some major advances in miniaturization has made UAVs more affordable and available than previously. This makes UAVs a good platform for many different tasks [1–2] ranging from aerial photography to search and rescue. Their versatility and continued miniaturization makes UAVs a very good candidate as a mobile platform for detecting a hidden transmitter.

The exact position of a hidden transmitter could be possible to calculate given enough time and enough information about the environment. However, in most real life situations this might not be possible. Lowering response time in search and rescue is of paramount importance. This means that the optimal solution might not be feasible and some approximation is needed. If the approximation is good enough, the system can function faster, with less available information and still solve the information gathering problem. By having autonomous agents, which can dynamically approximate the problem, an acceptable solution might be achieved.

Artificial Intelligence (AI) is a branch of computer science which seeks to create intelligent systems. These systems can range from a simple vacuum cleaner robot to huge complex systems made up of several intelligent agents, which cooperate to complete a task. Within AI, systems containing several cooperating autonomous agents are often called agent based systems. Agent based systems can broadly, for this work, be divided into two categories; systems where individual performance is the primary focus and systems where the performance of the group as a whole is the focus. The main differentiating factor between these systems is how the individual agents within the system behave. When self interest is the focus each agent will optimize its own gain. It can accomplish this through cooperation, but the agent can never assume that another agent is willing to do something that is not in its own best interest. The group focused system, on the other hand, can assume that all other agents will cooperate to optimize the groups performance.

Swarm Intelligence (SI) is within the group focused scope and uses inspiration from nature to create systems which can complete tasks in various different ways. Specifically, SI looks at cooperative organisms that work together to perform tasks which are often more complex than what each individual organism could handle alone. Examples of such organisms are bees and ants which create large and complex hives, far too complicated for any individual to create on its own.

SI has been used in many different situations, from global optimization, in the form of Ant Colony Optimization (ACO)[4, 18], to network routing and real-life robotics. The interest in SI comes from the scalable nature of the algorithms.

[1] Norwegian article about the different areas in which UAVs are being used in Norway: `http://nrkbeta.no/2013/06/08/nyttige-droner/`

[2] Norwegian video describing how the Norwegian Air Ambulance service are exploring the use of UAVs: `http://www.nrk.no/video/norsk_luftambulanse_tester_droner/72C8204286BF6A8E/`

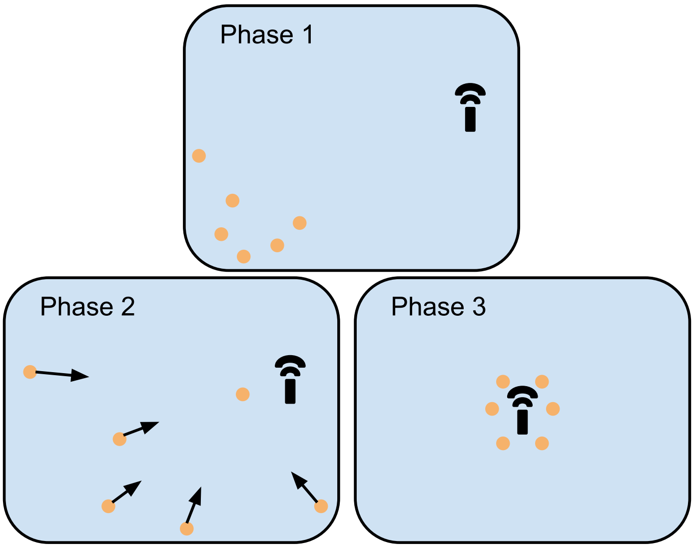

Figure 1.1: The different phases when trying to locate a transmitter. Phase 1 is to search for the transmitter. Phase 2 is to congregate at the transmitter. Phase 3 is to use some technique to estimate the transmitter position. Note that in the phase 3 illustration the transmitter has not moved, only the perspective is changed.

Because cooperation is the end goal of SI, the individual perspective is less important, making the swarm more fault tolerant. In addition, since cooperation is based on local interaction SI is often scalable. This means that the swarm can often grow large without loosing any performance.

1.1 Task Description

The problem considered in this thesis is only a part of a larger task. As mentioned above, the task is to locate a hidden RF transmitter. A hidden transmitter in this case is a transmitter for which there is no a priori knowledge, neither the radiated power or the position is known. It is also assumed that the transmitter is in a static location and that it transmits continuously. Broadly speaking the task can be split into three phases, illustrated graphically in figure 1.1.

The three phases in figure 1.1 roughly equates to:

Phase 1: Search the area trying to detect the transmitter

Phase 2: Once the transmitter is detected by an agent, converge at that agent

Phase 3: After converging spread out around the transmitter to precisely geolocate the transmitter.

If the area is large compared to the range of the transmitter, the first phase of the task is to detect the transmitter, obtaining the knowledge that there is a transmitter in the area. The second phase is for the agents to converge on the transmitter so that every agent, needed for the next phase, can detect the transmitter. The last phase is to use some form of geolocating algorithm to estimate the location of the transmitter. This last part has been studied in [16].

1.2 Goal and Research Questions

Goal *Gain a deeper understanding of the potential for swarm intelligence to solve problems related to detecting a hidden RF transmitter.*

The goal for this project is to explore the possibility of detecting a hidden transmitter using a swarm of UAVs. A swarm in this context means more than one agent. The task is to search an area for a single transmitter, detecting its presence. In figure 1.1 the task is illustrated graphically as phase 1.

For this thesis the search part of figure 1.1 is considered exclusively. This will aid in getting a deeper understanding for the spatial-search problem in addition to narrowing the scope of the problem domain. To enable future systems to encompass all parts of locating a hidden transmitter all three phases must be tackled. The intention is to have this thesis as one part of such an extended work.

1.2.1 Searching for a transmitter

Searching for an object is often just a means to an end. It is not the search itself one is interested in, it is the time it takes to find the object of interest.

Before location of a transmitter can take place, the transmitter itself must be detected and identified as the target of interest. In this thesis this is considered as the main problem. The problem is defined as: *Given a confined area. The knowledge, that within the area there is a transmitter the agents are capable of detecting. The agents should search the area, until the signal of the transmitter is detected above a certain threshold.* From the description of this problem it might seem trivial to solve, since all agents know the transmitter is present there is no need to detect it. However, it is not that straightforward, even though the agents in this problem know that a transmitter is present, in most real life scenarios this assumption may not be true. Often one assumes that a transmitter is present until the area has been searched, to a satisfying degree of certainty, before giving up that assumption. Therefore, the search in question is interested in the time it takes, from the agents start moving, until a detection is made.

To solve this problem a search strategy is needed to direct the agents so that the area is searched. One example of such a strategy is a random walk, if all agents move according to a random walk and can not move outside the search area, one of the agents will detect the transmitter in a finite amount of time. It is clear from this description that the transmitter will be found eventually, but one is often concerned about the time needed to locate the transmitter. This means that the time taken to find the transmitter is one important factor in the fitness of a strategy.

Related to this problem is the following research questions:

Research question I *Can several agents perform better than a single agent?*

Having several agents cooperate to achieve a goal is often a way to increase performance on a task. However this cooperation must be beneficial to the agents or else the benefit of being several agents will disappear and each agent is better off alone. For this reason it is interesting to compare the performance of algorithms with several agents, both algorithms designed for several agents and algorithms designed for single agents.

Research question II *How does random walk exploration compare to an organized search?*

In many real world scenarios, an organized search is performed by exhaustively checking an area to locate a given emitter. In an RF setting, exhaustively is not possible because of the need for a detection threshold. However, by making some assumptions about the threshold an area can be searched in an organized fashion. Such organization often requires a great deal of cooperation, and progress might be slow. With random searching each participant in the search can move without interruption or the need to coordinate with anyone. It is therefore interesting to compare the performance of a random search with an organized search.

Research question III *Can a technique which does not use any information available in the environment compare to a technique which uses that information?*

Detecting a transmitter can be quite complicated. Long before a true detection can be made the signal might be discernible from the noise in the environment. Using this information a search strategy might be able to direct the search towards the transmitter and thus perform better compared to a strategy which does not. The problem with this is that, as will become evident in section 2.1.2, there is no guaranteed way of separating the signal from noise. This means that an agent that tries to use the information in the environment might be following noise and not performing any better than a random strategy. For these reasons comparing different strategies with different utilization of the environmental information is of interest.

1.3 Research Method

Using several UAVs flying together is a large undertaking, having them cooperate to complete a task is even more challenging. Because of this, the work carried out in this thesis tests different hypotheses in a limited simulated environment. By keeping the environment limited, the work can concentrate on the implemented agents, studying their behavior and performance on the problem in section 1.2. This limits the scope of the results, but it creates a foundation which is possible to extend in future work and in later real world experiments.

1.4 Thesis Structure

The thesis is structured as follows. In chapter 2 background information and general theory about the field of research is presented. Chapter 3 introduces the concrete model used to validate the research questions. The results from simulations of the model are presented in chapter 4. In chapter 5 the result of the thesis is discussed along with future work and conclusions.

Chapter 2

Background

In the following chapter the background theory and related work is presented. First, information regarding radio signal propagation, detection and geolocalization is presented. Next is an introduction to UAVs, which should motivate them as an elevated mobile platform, before background theory related to agent based systems is presented.

2.1 Geolocating a Radio Transmitter

This section will introduce the concept of geolocating a radio transmitter. Electromagnetic (EM) propagation will be introduced first describing how radio waves propagate and how a signal is affected by elements in the environment. This section will motivate the desire to elevate the radio receiver and explain why this can lead to better location estimates. Then detection of RF signals is explained. Lastly geolocalization algorithms are introduced which explains different techniques to geolocate a hidden transmitter. This section motivates why several cooperating agents are envisioned for the future system.

2.1.1 Electromagnetic propagation

EM radiation deals with how EM waves propagate. A radio wave is a form of EM radiation where the frequency is between 3kHz and 300GHz[49].

When a radio wave propagates it will incur a propagation loss, L, which will lower the received signal strength. This loss is defined as the ratio shown in equation 2.1 where P_t is the transmitted power for the transmitter and P_r is the received power at the receiver[49]. The equation also shows how the loss can be expressed as amplitude, where A_t is the transmitted amplitude and A_r is the received amplitude at the receiver.

$$L = \frac{P_t}{P_r} = \left(\frac{A_t}{A_r}\right)^2 \tag{2.1}$$

In a real world many components like buildings, structural elements and terrain will interact with the radiated EM wave. To account for all these interactions, it is common to characterize the loss by a propagation loss or path loss model. Common to all propagation loss models is that they contain the free space loss. The free-space path loss describes the loss an EM wave naturally incur when propagating through an environment without any elements creating reflections or diffractions. The loss is due to how an EM wave propagates, creating an expanding sphere. As the sphere expands the surface area increases, resulting in a lower power density. This is illustrated by the green circles in figure 2.1, the further away from the transmitter the thinner the green circles become. In the figure P_t represents the transmitter, P_r is the receiver, d is the distance between them and L represents the loss affecting the signal as it is propagating. The free-space path loss model is described by equation 2.2[49].

$$L = \left(\frac{4\pi d}{\lambda}\right)^2 \tag{2.2}$$

In equation 2.2, d is the distance between the transmitter and receiver, while λ is the wavelength of the transmitted RF signal.

The Log-distance path loss model[49] is a generalization of the path loss in different environments. The model is given in equation 2.3,

$$L(d)_{dB} = L(d_0)_{dB} + 10\alpha log_{10}\left(\frac{d}{d_0}\right) \tag{2.3}$$

where α is a loss factor accounting for different environments, $L(d_0)$ is the path loss at a reference distance d_0 and the result of the function is given in dB. As shown in equation 2.2 the loss factor for the free space loss is 2. In other environments, where buildings and other objects interact with the transmitted signal, the loss factor can reach values in the range of 2-6[23]. A higher α value means that the signal is damped more by the environment. As explained above the dampening can have many different factors which the α value encompasses. Using α in this form characterizes the propagation environment. In figure 2.2 the log-distance path loss model is plotted showing how α, the dampening in the environment, affects the received power.

Since antenna design influences the way both the signal is propagated from the transmitter and the way the signal is received, all propagation models must

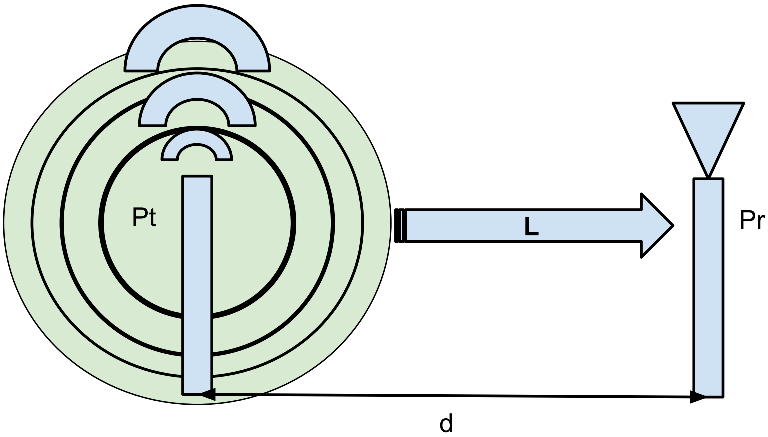

Figure 2.1: Path loss between the transmitter and receiver, illustrating how a signal propagates as a sphere.

make assumptions about antennas. Because the antenna can be optimized for many different scenarios selecting one design can often be challenging. A good initial model, is therefore an isotropic antenna. An isotropic antenna means that the antenna can receive a signal equally well from all directions and elevations.

Most of the time the environment that is being modeled is not static. This means that the received signal will be affected and will not be received as the pure signal in figure 2.2, in other words there is additional noise. This effect varies with time, frequency, position and is often modeled as normal random(Gaussian) variable[23, 44]. This noise represents additional effects, often called fading, and electronic noise experienced at the receiver. Formula 2.3 is therefore changed to include a random variable χ, equation 2.4 shows how this is added. It is important to note that both the α and this new noise are perceived as the same noise at the receiver, but they are often modeled as two different aspects. This is because they represent two different sources of noise where α is due to the environment while χ is due to other sources, most often thermal noise in the receiver[44].

$$L(d)_{dB} = L(d_0)_{dB} + 10\alpha log_{10}\left(\frac{d}{d_0}\right) + \chi_\sigma \tag{2.4}$$

In equation 2.4 χ_σ is a Gaussian random with a mean of 0 and a variance of σ. Figure 2.3 shows how this changes the received signal by the noise affecting it. The variance is dependant on the environment which the signal is propagating through and the environmental fading can be between 1.5dB to more than 16dB[23].

The free-space path loss model has often been used as a model when working

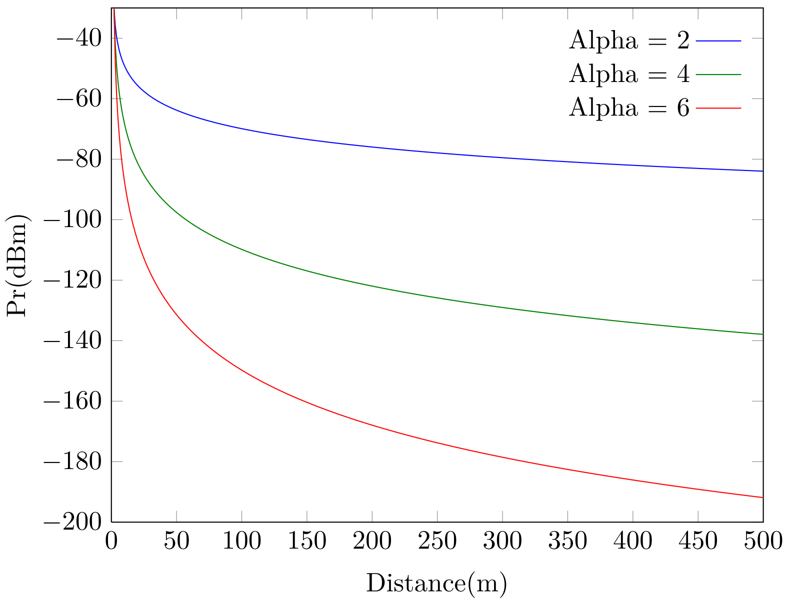

Figure 2.2: Log-distance propagation loss model, illustrating how α characterizes the path loss in different environments

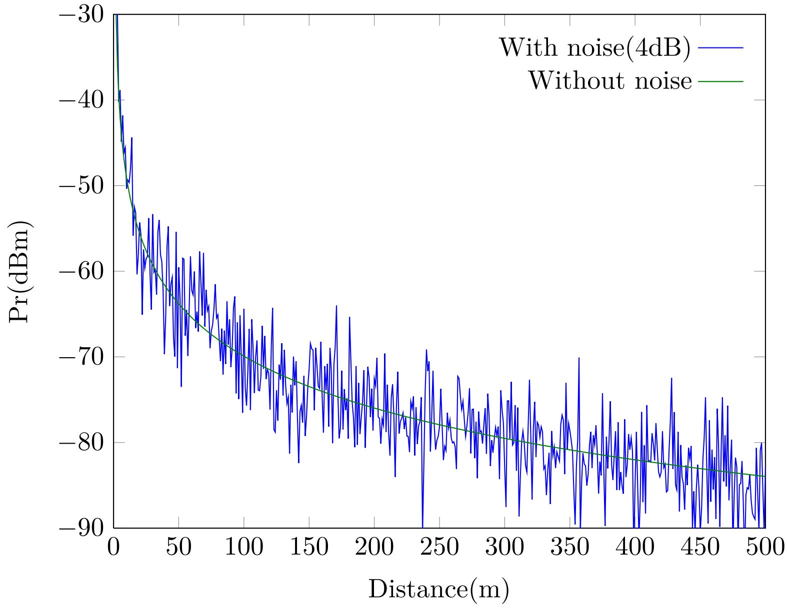

Figure 2.3: Log-distance path loss model with Gaussian noise

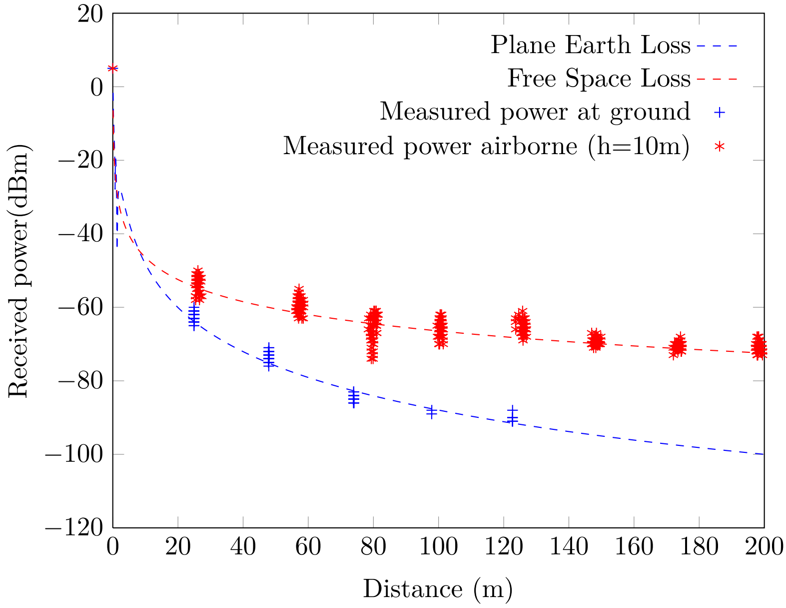

Figure 2.4: Plot showing real world received values, at ground and 10m above ground, compared to two different propagation loss models

with elevated receivers with line of sight to the transmitter. To test this an experiment was conducted at Norwegian Defence Research Institute (FFI) to see if the free-space path loss model equates to the actual Received Signal Strength (RSS) when 10m above ground[57]. The experiment used an UAV, a mobile phone and a regular Wi-Fi router, as the elevated platform, the receiver and the transmitter respectively. In figure 2.4 the result of the experiment is shown.

Figure 2.4 shows the measured data compared with expected theoretical propagation models. The red data points show values measured at 10m above the ground and the blue plus signs show values measured at ground level. The data shows that reflection from the ground affects the path loss and also show that α increases when there is more dampening in the environment in the form of reflections. Even though it is not clear on the figure, from the data one could also see that the distance at which the phone could receive was greatly enhanced by just being elevated 10m above the ground, somewhat evident by the lack of blue plus signs at distances over 150m. This data illustrates that when elevated the free-space path loss model is a good fit for actual data. This motivates the desire for elevating the sensor platform and will later be shown to be important when geolocating a transmitter.

2.1.2 Detecting Radio Frequency signals

Because of the physical nature of RF waves can only be detected within a certain range. This limitation comes from the receiver which has finite sensitivity and problems with filtering out noise from the environment. Because of this limitation receivers must have a threshold for when a signal can be classified as a distinct signal and not just noise looking like a signal. In figure 2.5 an example signal is compared to noise found in an environment. The figure illustrates the threshold, explained below, and it shows how noise can influence the detection of a signal. Only one of the two peaks above the threshold are an actual signal, meaning that there is one false alarm. Such a distinction, of false alarm, is not possible to make without complete information. This means that any sensible system must classify the two peaks above the threshold in figure 2.5 as detections. The threshold puts a limit on the statistical likelihood of the detected signal being a false alarm. By increasing the threshold fewer peaks will go above the threshold yielding fewer false alarms, but it can also mean that some signals are not classified as detections because of dampening noise. This highlights the importance of setting an appropriate threshold value. The threshold decides the probability of detection and is connected to the Signal-to-Noise-Ratio (SNR) and the false alarm rate. As mentioned above the threshold is connected to the probability of detection because it decides what is considered a detection. It is also connected to the SNR because the SNR is defined as the ratio of signal to noise. By increasing the SNR - e.g. by increasing the transmitted power or reducing the noise - the signal peaks increases and the difference between noise and signals becomes larger. This in turn means that peaks above the threshold are more likely to be actual signals and not noise. By relating the SNR to figure 2.5, one can see that by increasing the SNR the peak marked signal would become taller in comparison to the other peaks. The threshold and the false alarm rate are related, because the false alarm rate is defined by the threshold and the noise. Again relating to figure 2.5, if the false alarm rate was lowered the threshold would also be lowered. This means that if one allows more probability of a false alarm, the threshold for deciding a detection can be lowered which in turn gives a higher probability of detection.

Figure 2.6 illustrates the different aspects of detection. In the figure the red line is white noise, the light blue line is the signal with added noise, the black line is the threshold, the blue area is the probability of detection and the green area is the probability of false alarms. In the figure everything above the threshold is considered a detection, but as the green area illustrates there is a small chance of a false detection, i.e. noise being classified as a detected signal. The figure also illustrates the different dependencies between these factors. If the threshold

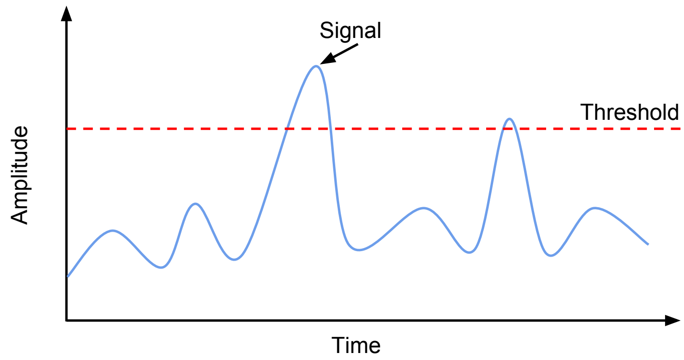

Figure 2.5: Detecting a noisy signal with threshold, adapted from [36]

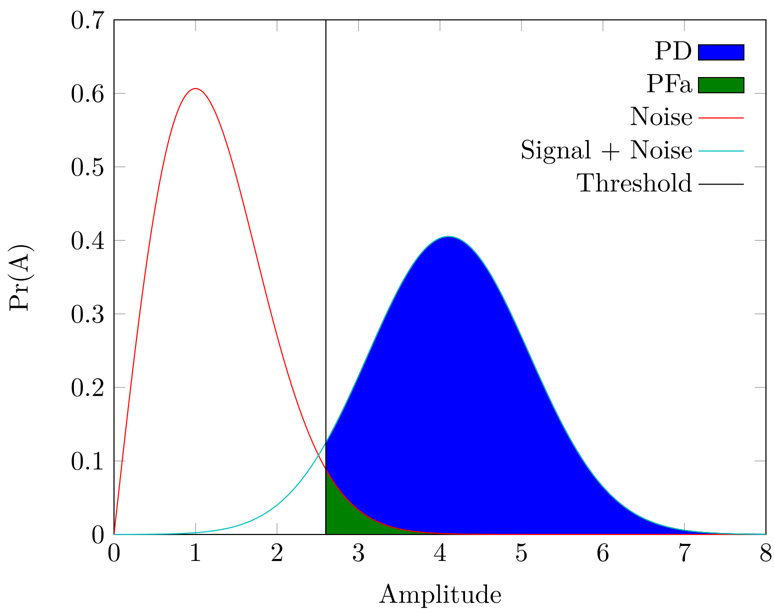

Figure 2.6: Illustration of the noise, signal, threshold and the probability of detection, reproduced from [36].

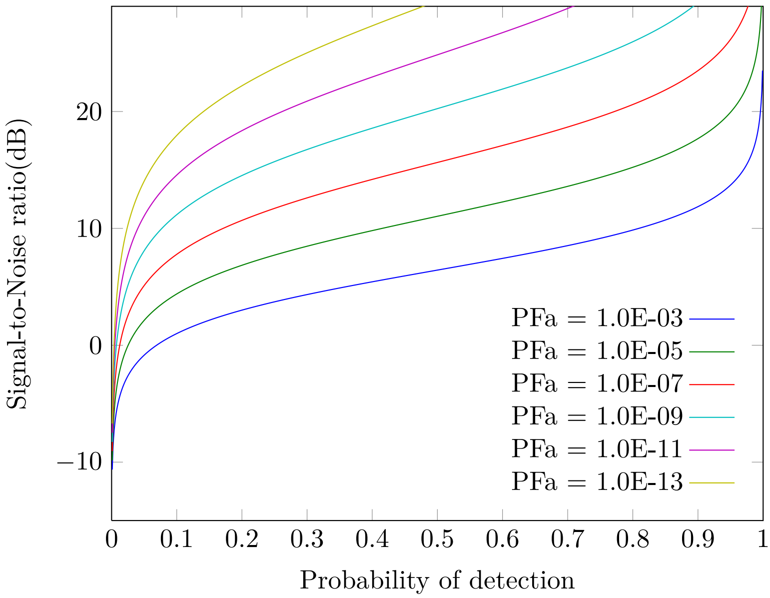

Figure 2.7: Illustration of how the SNR is connected to the probability of detection and the false alarm rate(PFa), reproduced from [36].

is increased the probability of false alarms decreases, but at the same time the probability of detection will decrease. From this one can see that the threshold and the probability of detection is connected, by increasing the probability of detection, the probability of false alarms will increase. This connection is further illustrated in figure 2.7. In figure 2.7 *PFa* is the probability of a false alarm. From the figure the connection between the false alarm rate, SNR and the probability of detection is illustrated showing how an increase in each leads to different characteristics of the others. More specifically if one decreases the probability of a false alarm, for example move from the blue to the green line, the probability of detection decreases for the same SNR.

Calculating the threshold can be accomplished by modeling the noise in the environment and selecting a value which is large enough so that the probability of such large noise values are unlikely. The noise added to the signal, as described in section 2.1.1, is modeled as a normal random. As the signal is modeled using the amplitude the noise can be modeled through a complex number. By generating this number from two Gaussian random variables with equal variance and zero mean, and by using only the magnitude the noise is distributed according to the Rayleigh distribution. The magnitude is used because it describes the amplitude of the signal. This simplifies the calculation of the threshold and there is no need to bootstrap, explained below, the process, the threshold can be calculated directly. The threshold can be calculated by setting the Cumulative Distribution Function (CDF) of the Rayleigh distribution equal to the desired probability of detection. The CDF of the Rayleigh distribution is shown in equation 2.5 and is rearranged in equation 2.6 to output the threshold.

$$CDF = 1 - e^{\frac{-x^2}{2\sigma^2}} \tag{2.5}$$

$$f(p) = \sqrt{-2\sigma^2 ln(1-p)} \tag{2.6}$$

where p is the defined to be $1 - PFa$.

As mentioned above the threshold could also be bootstrapped by generating a large vector of complex numbers where the two parts comes from two randomly selected values from a normal distribution. This vector represents the noise at the receiver with no other signal present. The amplitude of this vector can be extracted by calculating the magnitude of the complex numbers. Giving the threshold by way of bootstrapping.

Whether a detected signal which is above the threshold is a true detection or a false alarm, can then be checked with only the amplitude using equation 2.7.

$$f(r_t, d, p) = \frac{A_t}{\sqrt{d^\alpha}} \geq A_p \ \wedge \ d \leq r_t \tag{2.7}$$

In equation 2.7 r_t is the radius of the transmitter, i.e. the radius where the transmitted amplitude is equal to the threshold, A_t the amplitude of the signal, d is the distance between the transmitter and receiver, α is used as a variable to characterize the propagation environment, and A_p is the amplitude of the false alarm rate, e.g. if $p = 0.99$, 99% of the time the signal is properly detected and 1% of the time the receiver is either unable to detect a signal that should be detected or detects noise as a signal. By using this equation a detection can be ensured during simulation since it ensures that the amplitude the agents receive is above the threshold while also ensuring that the agent is within range of the transmitter. Ensuring that the agent is within range of the transmitter is important because of the probability of a false alarm.

2.1.3 Geolocation algorithms

Geolocating a hidden transmitter can be very difficult for a number of reasons. Because nothing is known about the transmitter, techniques based on frequency and transmitted power can not be used. If both of these were known one could use a propagation loss model and calculate the distance from a given measurement to the transmitter obtaining a radius in which the transmitter must lie within. Having several measurements of the same transmitter would yield several intersecting circles and would indicate that the transmitter is located in one of the intersection points.

Several techniques have been developed to overcome this challenge, using different properties of the received signal. Below is a list of techniques developed to locate a hidden transmitter.

- Angle of Arrival
- Time Difference of Arrival
- Frequency Difference of Arrival
- Power Difference of Arrival

Angle of Arrival (AOA) is based on measuring the direction of the signal at the receiver[23]. To measure the direction a directional antenna is used. The technique usually requires an array of receivers which can be large and bandwidth limited. AOA has the advantage of requiring only a single receiver to get line of direction, but two receivers are required to obtain a position in two dimensions. AOA is well understood and simple to use, but require specialized hardware.

Time Difference of Arrival (TDOA) measures the difference in arrival time at different locations[23]. By assuming that the signals travels at the same speed in air, the difference in arrival time can be used to calculate the position. Different receivers record the time of arrival which is then sent to a common site where the difference in arrival time between pairs of receivers can be correlated. Using three or more receivers a location can be calculated. The advantage of TDOA is the good estimates it can produce, but at the cost of complicated hardware. TDOA is often considered better than AOA because the antenna in TDOA is omnidirectional.

Frequency Difference of Arrival (FDOA) uses the apparent change in frequency occurring because of Doppler shift to the signal when a receiver is moving towards or away from the transmitter[23]. By knowing the velocity and direction of the receiver the Doppler shift can be calculated. Using the Doppler shift a line towards the transmitter can be extracted. Using two receivers which relays the information to a common site, a hidden transmitter can be detected by comparing the received frequency and the differing Doppler shift.

PDOA uses the difference of intensity at two receivers to obtain position[23]. By using the RSS at several different locations together with a path loss model, the distance between transmitter and receiver can be calculated. This distance result in several possibly overlapping circles which can be used obtain a position of the transmitter, again by looking at the intersections. This technique requires relatively inexpensive equipment which makes it suitable for several different situations. However the reliance on a path loss model makes the technique susceptible to deviations from that model. PDOA is therefore often less accurate than the other techniques described above.

From the description above several different geolocating techniques are available. In a related work[16], PDOA was chosen for several different reasons. Because PDOA uses simple hardware it is much easier to adapt the technique for use on a small UAV. In addition to this because the UAV is assumed to elevate the sensor the free-space path loss model can be used. This simplifies the calculation and, as shown in figure 2.4, is a reasonable assumption given that the sensor is elevated. It is important to note this when working with spatial location and detection as the assumptions made by PDOA could have an influence.

In addition to using intersecting circles to decide the position with PDOA other techniques have been developed[23]. The following are methods investigated in [16, 23]:

- Non-Linear Least Square
- Maximum Likelihood
- Discrete Probability Density

All these methods estimate the most likely place the transmitter is located in, but do so in different ways. The different methods have very comparable performance and in [16] Non-Linear Least Square (NLS) was chosen because of its simplicity and performance. NLS calculates the pairwise difference in received signal power at a given position, estimating the probability of a transmitter located in that position. The method minimizes the sum of the difference between the actual received value for the pairwise receivers, and the theoretical received strength given that a transmitter is located in a position (x, y), by assuming that the path loss model is equal at both receivers. Equation 2.8 describes the calculation for a given position (x, y).

$$Q(x,y) = \sum_{k<l} \left[\bar{P_{kl}} - 5\alpha log_{10} \left[\frac{(x-x_l)^2 + (y-y_l)^2}{(x-x_k)^2 + (y-y_k)^2} \right] \right] \tag{2.8}$$

In equation 2.8 the summation is over all receivers, where (x_l, y_l) is the location of receiver l, (x_k, y_k) is the location of receiver k and $\bar{P_{kl}}$ is the actual difference in received signal between receiver k and l. In simpler terms, the method tries to find the position, (x, y), which minimizes the difference from actual RSS measurements.

Equation 2.8 is non-linear meaning that to estimate the transmitter position the function must be applied to a discrete grid where the size of the grid limits the accuracy.

(a) RQ-1 Predator (b) RQ-11 Raven (c) Black Hornet

Figure 2.8: Different UAVs illustrating the different sizes and different configurations possible for a UAV. Black Hornet photo: Richard Watt/MOD.

2.2 Unmanned Aerial Vehicle

As mentioned in the introduction UAVs are envisioned as the vehicles to use the results of this research on. It can therefore be of some use to introduce UAVs as a concept.

In general a UAV is an aerial vehicle without a human pilot inside the aircraft. It is often controlled from a ground station by a human operator and has a level of autonomy ranging from none to completely autonomous. A typical system consists of an elevated platform, the UAV, and a ground control station with a communication link to the UAV.

UAVs come in many different sizes and many different types. The size can range from the large UAVs operated by defence forces around the world to smaller crafts designed for e.g. aerial photography or ground mapping. Figure 2.8 shows different UAVs, their sizes are displayed showing the range and type differences between some selected UAVs.

While there are different classes of UAVs differing in size, endurance, range, payload capacity, etc., the most important property in the context of this work is whether it is able to hover or not. Typically a UAV which can hover is some form of helicopter which either uses a main rotor and a tail rotor or an even number of pairwise counter-rotating rotors. The ability to hover influences how algorithms working on the UAV has to operate. On an aircraft not able to hover, the flight planner must always take into consideration that the UAV must move forwards which influences how the aircraft can be used. As an example looking at the work done in this thesis the model itself assumes that the agent is able to come to a complete stop and wait for some period of time. The reason for this requirement is to sample the RF environment or wait for other agents to perform some task. This could possibly be extended so that the wait is a holding pattern where the aircraft circles, but that will at the very least mean that the algorithms must be updated to take into account the change in position for each

RF sample. Both types of UAVs, hovering and non-hovering, have their own strengths and weaknesses resulting from the differences in how they fly and what other features they have in regards to movement. As noted this work is based on the assumption that the vehicle is able to hover, but this ability usually means that the aircraft has lower flight endurance.

The reason behind the choice of UAV as the intended platform is based on a number of factors. Most importantly, the work considered here deals with locating RF transmitters which have several possible noise sources. As explained in section 2.1, these noise sources are greatly reduced by elevating the sensor platform and thus UAVs can aid in the work as compared to other ground based vehicles. Having the ability to hover means that the UAV is able to stop for some time in one location and sample the RF environment. This can lead to better performance as the platform can get multiple samples at the same location which could be important[1]. The flight model also gives full flexibility with regards to behavior which makes, future, real world experiments easier. In addition UAVs are becoming more available, have better sensors and can have a small form factor which also plays a role in the selection. UAVs are also highly mobile which means that tasks can be performed with less time spent travelling and more time performing the task.

2.3 Autonomous Control using Agent Based Systems

Controlling a vehicle without the use of a dedicated human requires some form of autonomy. This section introduces the concept of an agent based system as a control scheme for several vehicles. The section will first introduce agent based systems before going into SI. SI will have a special focus and several techniques are explained in relation to autonomous control.

Agent based systems are systems containing several agents interacting with each other[67]. The interaction among the agents within the system can be either:

- Cooperative
- Selfish

Cooperative means that the collective good is the goal, as seen in SI, and selfish meaning the individual good is the goal, as seen in a free market economy. An agent in this context is simply some entity which can observe and influence the environment it is situated within. It is important to note that this definition

[1] When the RF environment is noisy, multiple samples can be correlated resulting in a reduction in noise.

does not specify if the environment is real or software defined, and does not specify the intelligence of the agent.

An important concept is that an agent will always decide for itself what to do next. This separates agents from a class in an object-oriented programming language[67]. A regular class will have its methods called by other classes, or objects. Because of this it could be viewed as an agent, but crucially a class such as this has no choice to handle an incoming method call. This is in contrast to an agent which will observe some change and decide to react to it. Nothing is forcing it to react and the decision is made within the agent. An example of a system which can be viewed as an agent is a simple thermostat system. The thermostat system will react to the environment by sensing the temperature and decide to turn on or off. This can be viewed as an agent because it can sense the environment and perform some action within that environment. It would seem that this example is no different than the object-oriented class since one could seemingly force the thermostat system to react by lowering the temperature. The difference however lies in how the two systems control their behavior. The thermostat system senses the environment and decides to act, while the object-oriented class must act if another object calls one of its methods. Wooldridge [67] put it quite elegantly,

> Objects do it for free; agents do it because they want to.

The environment an agent operates in can have several different properties which influences how the agent operates. Since the agent and environment is so closely linked, understanding the environment is important in understanding how the agent should behave. The most important properties of the environment is what information is available for the agent and how the environment is updated. If every piece of information is available for the agent and nothing except the agent updates the environment, the agent can operate with great proficiency, deliberating in eternity before executing the perfect action. However this is rarely the case and in a real world scenario things like uncertain measurements makes the decision process of most agents quite difficult. Moreover the environment is often dynamic where other agents make updates, meaning the agent can not deliberate forever as assumptions made are subject to change. With such uncertainty any agent must be able to react to events and must act within reasonable time.

Another important aspect of agent based systems are their social ability. Because agent based systems are assumed to be able to interact with other agents they need to have some form of social ability. This result in agents that are capable of cooperation, coordination and negotiation. Without this social ability the agents would not be able to form coalitions and work together to achieve some goal. This further separates agent systems from other AI systems such as expert systems. The social ability comes in different forms as mentioned above and agent based systems do not have to cooperate. In many systems the agents represents some interests and will cooperate only to achieve its own interests. In contrast a cooperating system will sacrifice individual performance for global performance if necessary.

The alternative to an agent based system is a centrally controlled system. In such a system there is only one agent, the central controller, which controls several other objects[2] in the environment. Often the objects will be capable of sensing and very simple autonomy, but coordination and decision making will derive from the central. The major difference between the two systems is the way the central controller directs several agent like objects as opposed to those objects being agents with their own agenda. The central controller can still be thought of as an agent built up of several unconnected pieces, but it is usually contrasted with agent based systems as it limits autonomy of the objects inside the environment. This sort of autonomous system is oftentimes easier to create as coordination is handled in one place, the challenge however is the reliance on one agent making all decisions. If the agent has to direct several objects the calculation of these actions may take a long time. In addition the objects must always be in contact with the central controller and if that central controller encounters a problem the whole system must halt.

2.3.1 Swarm Intelligence

As mentioned in the previous section SI can be viewed as an agent based system where the group performance is the objective. SI is attractive as a control system for distributed agents because of its focus on group performance, local interaction and emergent behavior.

SI takes its inspiration from nature, more specifically social insects. The field grew from observation of social insects in their natural habitat wondering where the level of organization came from. By observing social insects as a whole, impressive structures emerges where some parts of the swarm are tasked with finding food while others with hive defense. Looking closer at the individuals reveal that they have their own agenda. They are not directed by a single leader, but instead follow some internal guidance, they are autonomous agents. Only as a whole are the social insects able to do the impressive array of building nests, care for the brood or forage for prey to sustain the hive [4].

What seem to guide these social insects are simple local interactions which together gives rise to complex behavior.

One important paper to mention is Mamei et al. [38]. The paper creates a taxonomy for SI and will be used in this section as a common classification to

[2]These objects are distinct from the objects on object-oriented programming and are just actors within the environment which act according to the central controller.

base the theory around.

Social Insects

Social insects are a fascinating field of study. These insects live in societies of several individuals cooperating to reach a common goal. These insects do not have a leader[3], but still complex behavior arises. Several such insects have been studied, examples include the honey bee(*Apis mallifica*), the wasp(*Polybia occidenyalis*), army ants(*Eciton*) and termites(*Macrotermes*) Bonabeau et al. [4]. These examples all show behavior much more complex than their individual capacity would allow.

Self-Organization

As mentioned before Self-Organization (SO) is an important concept within SI. SO is the order which emerges at a global level from strictly local interactions within a collective of agents. Local interaction is in this case interaction happening between the lower level agents without any global mechanisms to guide them. SO is the driving force behind the emergent behavior observed in social insects.

SO is driven by two prevailing feedback forces. Positive feedback reinforces behavior making an insect more likely to do something. As can be seen in ants, where worker ants have a higher probability of depositing material if there is an unfinished structure. The other type of feedback is negative feedback which dissuades behavior. This can be seen when social insects are clustering. When an insect is clustering it will pick up items. The probability of picking up items is influenced by its surroundings. If there are several similar items near each other negative feedback will ensure that it is less likely for the insect to pick up the item. In the last example already formed clusters provide negative feedback. While positive feedback is used to reinforce or encourage behavior, negative feedback is needed to stabilize the organization. Without negative feedback the organization could never come to a stable equilibrium. In the clustering example, if there were no negative feedback affecting the ants, they would continue to cluster already deposited items indefinitely, never reaching an equilibrium of clustered objects.

In addition to feedback there is a need for some random element. The example of clustering above alluded that workers might pick up objects already clustered, but the chances should be small. This random element helps with exploration of different solutions or alternate ways to achieve a goal. One example can be seen with ants when creating a trail between the hive and a prey. In this situation the ants will deposit pheromones, described in section 2.3.1, to create a path, other ants will follow. Other ants will only follow the path with a given probability,

[3] Even though queens may sound like a leader they in fact do not perform any such task and are just there to create offspring in most social insects.

which is reinforced if more ants follow the path laying down more pheromones. Such a path may be longer than another and over time some ants could chose the shorter path randomly. If this new path is shorter the pheromone trail should increase because ants using it will be able to take shorter trips resulting in more pheromone deposition on the shorter path. Eventually this new shorter path should overtake the longer path, in pheromone concentration, resulting in the ants finding the shortest path. In this example the continuous random element induces the ants to explore the environment which could lead to better solutions being found.

Communication

Until now there has just been mentions of local interaction, but this interaction needs to be facilitated somehow. For most social insects this is achieved through some from of communication, either direct or indirect.

Stigmergy is a way to indirectly communicate by changing something in the environment. Social insects have developed several ways to perform this task from chemical deposits, known as pheromones, to indirect signs in the hive. Stigmergy is a powerful tool which allows the social insects to cooperate without a specific leader or explicit communication. One example of stigmergy is the example of clustering among ants, as mentioned above. In the example clusters of objects acts as a communication to other ants, that objects should be deposited here and not removed. The communication is through the environment and thus can be viewed as stigmergy. It is important to note that this communication is very scalable. Because the environment is often much larger than the agents within, it can accommodate many agents working in the same place at the same time. Going back to the clustering example, one can see that if one ant deposit an item at a cluster communication is happening. Yet the same thing is true if ten or twenty ants deposit at the cluster at the same time. This is important because an ant looking to pick something up will get the same information and it will not be too much information. This last statement can be contrasted with electronic communication where the number of messages and the number of entities communicating can not grow over some given bound. If there are too many communicating entities the system will uses all its time processing messages until it can not handle more messages and stops communication.

Pheromones is an important concept used by ants to communicate. The pheromone is marker based[38] and is used in many different ways. Pheromones can be used both as a trigger based mechanism and also as a follow through mechanism. In the shortest path example above, the ants used pheromones, as an indirect form of communication (i.e. stigmergy), as a follow through mechanism.

Pheromones is a powerful way of communicating and since it is a form of stigmergy it has the property of being scalable. Because the communication is through a marker being placed down, the communication is a diffusion of information which enables the ant to place the marker and then forget about the event. If the same ant travels over the pheromone, it will again detect it and base its action around the environment and not its own short term memory. This means that an agent using this technique can be simpler in its design or utilize its resources for other tasks.

In 1967 Karl von Frisch discovered that bees communicate by dancing[62], a discovery which later resulted in a Nobel Price[4].

By dancing, bees can communicate different needs. One example is when there are few foragers. In this situation one of the bees responsible for handling the incoming foragers may notice that many bees are idling waiting for foragers. The bee may then begin to dance to communicate that more bees are needed to forage, which might influence other bees to start foraging.

Through this language bees can affect the environment and talk to each other. It is important to note that this is not a direct form of communication and is therefore a form of diffusion of information. The dance is not directed at any specific drone in the hive, but the communication is directed at anyone taking the time to notice and pay attention. This property leads to scalable communication which is efficient in regards to autonomous agents.

Emergent Behavior

The different forms of communication all lead up to the interesting property within the social insects; the fact that behavior on the global level emerges as a property of the interactions on the basic level. This property is why SI is so interesting, because local interactions can scale to large populations. Understanding why these behaviors emerge can help design systems which can also scale. By knowing why a few interactions among separate agents lead to clustering, sorting, nest construction or recruitment, one can build ever larger systems with many of the same capabilities. Most of the systems designed with SI in mind, try to utilize emergent behavior in some form. Because much of the communication is directed at anyone willing to receive, the agents are often interchangeable. This leads to more fault tolerant systems which more gracefully handles failure of some of the agents comprising the system. Since the communication arise on a local level and the desired effects on a global level, the systems designed in this way can be efficient and scalable.

[4]"The Nobel Prize in Physiology or Medicine 1973". Nobelprize.org. Nobel Media AB 2013. Web. 19 Mar 2014. http://www.nobelprize.org/nobel_prizes/medicine/laureates/1973/

2.3.2 Cooperative search

Cooperative search, in this context, looks at several agents, $N > 1$, which together are trying to achieve the goal of finding a given target. The agents may communicate and can search in any manner which results in the agents as a whole finding the target. The techniques presented in this section take inspiration from several different places, not only from SI.

Cooperative search relates to the problem described in section 1.2.1, detecting a transmitter. As the previous section on geolocating, section 2.1 explained using more than one agent is often needed. Since noise is a factor which can be reduced by having several independent samples, using several communicating agents can diminish the impact of noise. This leads to the realization that using a cooperative search strategy can improve detection time.

One important point to note here is that the search can be performed in one of two ways:

- Utilizing the information in the environment
- Not utilizing the information in the environment

Since the object of the search is an RF transmitter there exists some information in the environment. Since the transmitter is sending out a signal there is an opportunity for the agents to search, directed by this signal.

Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a search and optimization technique inspired by how birds fly and search for food. PSO was first described by Eberhart and Kennedy [15] who created the optimization technique by trying to recreate the flocking of birds. They noticed that by using simple rules for flight the flock could move in unison and also seemed to be directed towards a goal. A good summary of PSO and different versions can be found in [26].

The simplest version of PSO is built around the notion that each solution, or particle in PSO terms, move towards some previous best location this particle has recorded and the best location that the swarm as a whole has currently. In addition some randomness is included so that the swarm might also search other areas where it might not have gone otherwise. In figure 2.9 a graphical overview is given for the simple version of PSO.

The movement update in figure 2.9 is carried out by the following equation,

$$v_{id} = v_{id} + \varphi_1 * (p_{id} - x_{id}) + \varphi_2 * (p_{gd} - x_{id}) \tag{2.9}$$

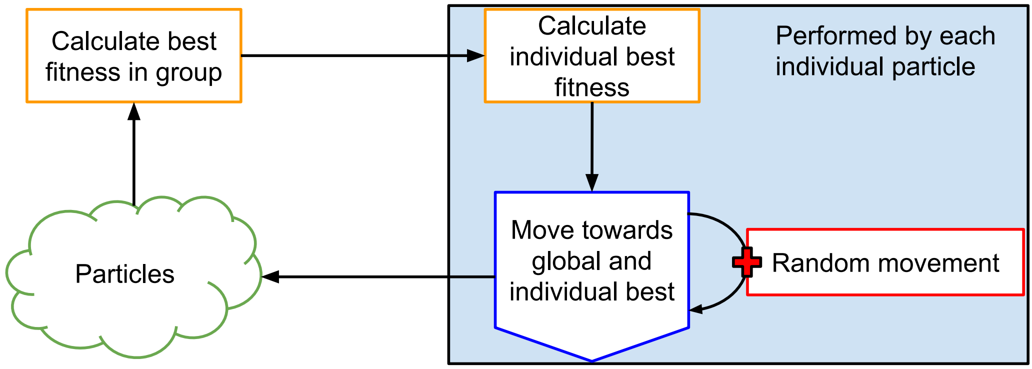

Figure 2.9: A graphical overview of the PSO algorithm.

where v_{id} is the velocity of particle i in dimension d, φ_1 and φ_2 are two uniform random variables, p_{id} is the individual best, x_{id} is the current location and p_{gd} is the group best.

In a follow up paper Eberhart and Kennedy [14] changed the optimization model to work with more local information, by having each particle converge towards the best of some limited number of its neighbors instead of the population best. Instead of comparing against p_g each particle compares itself against some number of particles from the neighborhood. This change had the effect of making the swarm search a larger portion of the search space, but still converge quickly. This version is also less susceptible to local minima. This change has many similarities with simulated annealing[28] and the use of roulette wheel selection in Genetic Algorithms (GAs)[18] which enables some inferior solutions to propagate and give some diversity to the system.

An analysis of the convergence of PSO was undertaken by Clerc and Kennedy [9]. The paper analysed the convergence and stability of PSO in five dimensional space and found that by altering the original version of PSO one could achieve better performance on several standard non-linear optimization problems. This increased performance came with only a small increase in computational and memory resources.

In addition to the analysis of PSO some authors have tried to simplify PSO to better understand which elements of the original algorithm contributes which characteristics. In [25] Kennedy developed a simplified version of the original PSO algorithm and compared different versions to see the effect. This simplified version performed better than the standard PSO algorithm. Another simplification can be found in [6] where the authors created a version of PSO which does not use a particle best, but utilizes the neighborhood in a different way to the standard PSO. They showed that by removing the stochasticity and the memory from each particle, and using the particle neighbors in a novel way, they removed the bursting behavior observed in most other forms of PSO. They also showed that one of the most important things in PSO is the social interactions.

Other approaches to more robust PSO algorithms have been explored. In [31] the authors combined PSO with a Gaussian distribution instead of a uniform distribution in the standard PSO variant. Their work showed that the velocity update in the PSO algorithm could be simplified by observing that the mean values for the velocity update usually end up between some known limits. By exchanging the uniform random number and the local best and neighborhood best with a Gaussian function the algorithm became simpler, and less variables had to be specified. As the Gaussian distribution has much larger tails than a uniform distribution, the algorithms ability to escape local minima is larger. Their results showed that this Gaussian PSO did not perform any worse than the regular PSO, and in many cases performed better. An extension to this work and the bare bones PSO created in [25] was explored in [46]. In their work the authors changed the sampling distribution from a Gaussian and uniform to one based on a Lévy distribution. Taking inspiration from the foraging pattern of animals (more on this in section 2.3.2) this extension was tested against both Gaussian and standard PSO and again showed improvements over both, especially in regard to the spatial distribution of the particles. The last part should be of special notice as it mitigates the risk of local minima in addition to sampling a larger space.

PSO has also been used in many practical applications. Some of which are optimization of a function or several functions and some of which are related to search. In [12] the authors used several instances of PSO to search for objects in a closed off area. They used one PSO to tune the parameters of another PSO which was used for searching in a two dimensional space. Derr and Manic [11] used PSO and simulated robots to search for a wireless signal in an enclosed area. They found that by distributing a number of agents randomly within an area and using a fitness function based on the free space propagation loss model they could find and converge on one and several transmitters. They also noted that because of propagation loss due to reflection the task could become slower, but the task would still be completed.

Artificial Potential Field

A large challenge in many mobile robots is the task of obstacle avoidance. Avoiding obstacles can be a challenge because of unreliable sensors and inaccurate actuators. It can also become difficult because of varying environments and the need for overall good performance. Artificial Potential Field (APF) tries to solve this problem by creating repulsive fields around obstacles which a robot must avoid [27]. Khatib [27] showed that by constructing repelling forces between

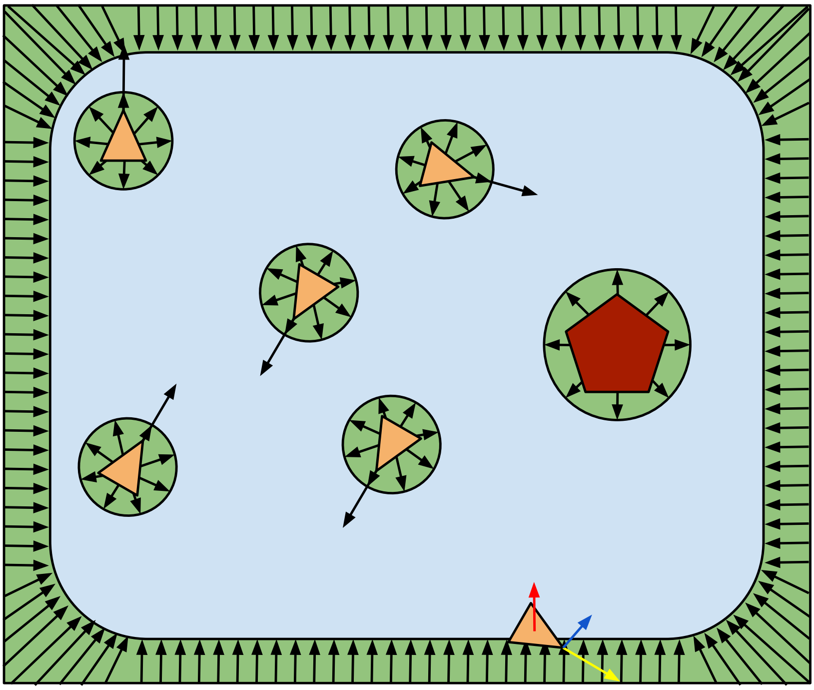

Figure 2.10: Illustration of APF in a robot domain. The orange triangles are robots, the green areas are the reach of APF forces and the black arrows are the current travel vector. Also shown is a red obstacle which will influence the robots if approached.

obstacles, a robotic manipulator could operate without collisions with a simple mathematical basis suitable for less powerful computers.

Figure 2.10 illustrates how APF could function in a mobile robot domain. The bottom right robot is experiencing some forces, its current travel vector, shown in yellow, points it down towards the outer wall, the red arrow is the APF vector pushing it away from the wall and the blue arrow represents the updated travel vector safely moving the robot away from the obstacle.

In [5] the authors used a combination of APFs and certainty grids[40], to achieve a fast and accurate obstacle avoidance and route following algorithm. By using certainty grids to contain unreliable sensor readings and APFs to manage obstacle avoidance their algorithm could be used on a relatively fast robot.

Warren [65] used APFs and global knowledge of the environment to enable global path planning through an obstacle cluttered environment. By knowing in advance where obstacles were placed the author created APFs at these obstacles, constructed a path through the environment before updating the path under the influence of the APFs. By doing this a global path was constructed for both a robotic manipulator and a mobile robot. The major contribution of this work was a path planner which was not as affected by local minima as previous APF methods. In a follow up paper[66] the same author showed how the algorithm above could, with some changes, be applied to the path planning of multiple robots operating in the same environment. By giving each robot a priority the path planning could be performed by having the robots plan according to priority. The robot with the highest priority could plan its path by only considering the stationary obstacles, as the previous algorithm by the same author. The next highest priority robot could then plan its path by viewing the highest priority robot as a moving obstacle to avoid.

Another work, by Lee and Park [35], used APFs together with simple neural networks to generate a path. The authors used several agents representing points on the path connected by line segments which updated their position according to local rules to generate a near optimal path. By using neural networks to represent obstacles efficient calculations could be performed with the addition of output ranging from 0 to 1 used later. The authors used simulated annealing[28] to escape local minima by utilizing the neural networks possibility of outputting a degree of collision. This resulted in a fast algorithm capable of real-time operation. The algorithm was also massively parallel because of the local update rules used.

By considering only the local path planning problem the authors in [34] showed that local minima could be escaped by putting down virtual obstacles when a local minima is detected. Their algorithm could efficiently maneuver an environment with a local minima the plain APF method could not. Even though the authors do not mention it, the solution is very similar to pheromones used by ants.

As described above using APFs for local path planning has had academic interest for some time. A challenge with APFs however is its one step reduction of obstacles into vector forces. This problem was investigated in [29] where several limitations with regards to APFs for local path planning was identified.

Other areas of robot control has also used APF with success. Howard et al. [21] used APFs to solve the area coverage problem. The problem is to cover a given area with some sensor platforms. The authors showed that by using APFs and only local movement rules a swarm of robots could create a very good coverage while simultaneously avoiding obstacles and each other.

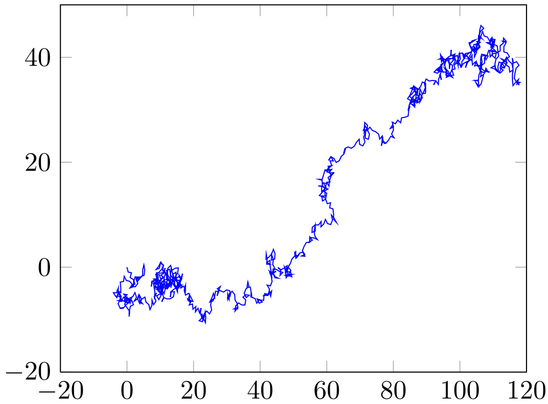

Figure 2.11: An example Lévy flight

Lévy Flight

A random walk is a path separated into segments where the direction and length of each segment are sampled from one or two random distributions [43]. Often the direction is sampled by a uniform random distribution, while the length is sampled by another random distribution. In cases where the direction is uniform and the length is of some special type, the random walk is often characterized by the distribution of the length. Several such random walks exist from the uniform random walk to the Brownian random walk. Of special interest is Lévy flight, first mentioned by Mandelbrot [39], which is a random walk sampling lengths from equation 2.10.

$$P(l) \sim l^{-\mu} \tag{2.10}$$

Equation 2.10 describes the probability of obtaining a length l which is distributed according to l to the power of a parameter μ.

The Lévy flight is characterized by many small consecutive jumps followed by, often, one long jump. This can be seen in figure 2.11 where a sample path is shown. From the distribution one can observe that the numbers will tend to smaller values, but there is a possibility for large values as the distribution is tail heavy. Another important factor is that Lévy flight is scale free, or scale invariant. The scale invariant property means that the distribution can be scaled arbitrarily large or small, which in turn makes the flight usable in several different scenarios. For instance in figure 2.11 the scale could be an agent searching along the northern coast of Norway looking for cities, or it could be bacterial movement within a cell.

Lévy flight can be observed in many patterns of animal foraging. In [59] the authors observed how the foraging of Albatrosses can be modeled using Lévy flight. Later the same authors showed how for an unknown number of sparsely separated foraging sites the optimal strategy is to chose a Lévy flight with $\mu = 2$ [60]. In a related work Reynolds [45] showed that for $N > 1$ foragers the optimal search strategy when targets sites are randomly and sparsely separated is for $\mu \sim 1$. The author noted that when cooperative foragers are able to communicate target sites the above limit holds. The author also noted that this behavior might also explain the orientation flight of honeybees, which was mapped by Capaldi et al. [8]. An important note to make here is that this special flight where $\mu \sim 1$ is applicable to both communicative foragers as well as non-communicative foragers if target sites are densely populated. In [13] the authors disrupted the waggle dance of three different spatially located colonies of honeybees and observed that only the colony which had sparsely separated target sites where affected. This hints at when target sites are densely populated around the foragers cooperation has little effect on overall foraging.

Sutantyo et al. [56] used Lévy flight to search a two dimensional area coupling it with APFs and comparing it with a uniform random walk for area exploration. The authors noted that for exploration both Lévy flight with and without APF performed better than random walk exploration.

In addition to these uses within search, other authors have used Lévy flight for other purposes. Lee and Yao [33] used the Lévy probability distribution to enhance mutations in a GA showing how the long tail of the distribution can help with variability.

2.3.3 Source seeking

A lot of research has been done within the field of source seeking. The goal of most of the research is the detection and localization of some source, most often some form of chemical (for a good introduction to chemical plume tracking see [20, 30]). Other sources have also been explored, but the unifying factor is that the source is emitting some information into the environment. Most of the early work went into single agent source seeking, a trend which has shifted over to a more distributed approach recently[30]. By having several agents cooperate the source seeking task can often be performed quicker and with less probability of getting stuck in local minima. The research done within this field has mostly focused on two areas, gradient climbing and memory based map building.

Gradient climbing

Most of the literature surrounding source seeking have worked with sources which forms some sort of gradient. Moving on this gradient has therefore been given a lot of attention.

Because of the focus on chemical tracking a lot of inspiration has come from biological systems with a special interest in the chemotaxis performed by bacteria and some other animals[48]. Bacteria such as *Escherichia coli* search for nutrients and avoids repellent by moving randomly for some lengths before selecting a new random direction. The selection of direction is influenced by the current environment and the change in direction occurs more frequently when in an repellent or attractant environment[3, 37, 48], illustrated in algorithm 2.1. This random walk, often called a biased random walk, directs the bacteria from repellent and neutral areas towards attractant nutrients. Another early use of taxis was the realisation that moving upwind when detecting a chemical, anemotaxis, can be a good strategy to find a source[47]. In [22] the authors combine anemotaxis and chemotaxis to perform a guided search towards a chemical source. This combination performed better than either taxis on its own, and they showed that by using each taxis at the correct time can alleviate some of the problems experienced with each taxis individually.

Algorithm 2.1 Algorithmic illustration of chemotaxis

while Not at goal **do**
  C_t = Read sensor
  if $C_t > C_{t-1}$ **then**
    Turn $\pm$ Random($5°$)
    Move forward
  else
    Turn $\pm$ Random($180°$)
    Move forward
  end if
end while

Some work has also been undertaken in the single agent search for EM signals. In [55] the authors simulated a single agent with an RSS receiver which searched for a transmitter. By estimating the RSS gradient their agent could move towards the transmitter. They also noticed that by adapting the number of previous measurements used, the agent could improve its movement depending on where it was located. A similar study was undertaken in [69] where the authors also estimated the gradient towards a transmitter. Their analytical approach showed that estimating the gradient was possible and they created and algorithm which was tested in a simulator. The same authors extended their work in [42] where they expanded their analysis to two dimensions and showed again that their algorithm was robust to noise. Wadhwa et al. [63] used a chemotaxis inspired RSS based estimation algorithm which moved along the gradient, but made some special maneuvers when the rate of progress slowed beyond a given threshold. By moving in a pattern to sample RSS the authors could more reliably estimate the gradient. Because of their focus on a single agent searching, these results are included as textual references, but is not implemented.

In recent years a lot of work has focused on distributed source seeking[1, 30, 68]. The difference is simply to use several cooperating agents to search for the source. Much of the research has again revolved around chemical plume tracking, making it comparable to the previous research. Hayes et al. [20] used several, simulated and real world, robots to implement a cooperative chemical plume tracking algorithm. They used energy and time as metrics for their result and showed that several agents are better than singular, but also showed that crowding could become a problem. In [2] the authors used collaborative formation control to give the agent a stable form which could be used to estimate a gradient to descend. However this form needed to be completely rigid which makes the technique unsuitable for real world UAVs. The authors use inspiration from fish schooling to derive an algorithm which can guide the agents down the gradient. Ogren et al. [41] used APF for stable formation control and a gradient decent algorithm to move in a noisy scalar field. Their important work showed that by decoupling formation control and the gradient decent algorithm, the design of the agent could become more robust. They also showed that by letting the formation expand and contract in response to the scalar field, the agents could overcome noisy measurements. Though the work is important, the need for a rigid formation makes their result hard to transition to UAVs. In [24] the authors modified the regular PSO algorithm to make it better suited for plume tracking, by combining APF and PSO they showed that a plume tracking task could be completed and increased the diversity of the area searched.

In a similar vein, Zarzhitsky et al. [70] used their previously developed physicomimietics [52, 53, 54] formation control and created a new form of taxis, called fluxotaxis, to search for a chemical plume. Fluxotaxis is based on fluid dynamics and uses the rigid formation created as a distributed computer to solve the fluid flow problem. Their algorithm can intuitively be understood as using the rate of fluid change in an area to guide the search. By comparing their new algorithm with chemotaxis and anemotaxis they showed that their algorithm could perform better both in regard to time-to-find and detection frequency. The drawback of their implementation was the need for a rigid body to create the distributed sensing platform. This could be interesting to test in future work and their physicomimetics is an important technique to review when formation control is desired.

Memory based

Some work has also been undertaken in developing non gradient based source seeking algorithms. What most of these have in common is their use of a persistent map. By using a map to store information about past measurements, these algorithms can both calculate a gradient to follow and can also utilize the map to calculate where to measure in order to gain the most information. In [51] the authors used a mobile robot to track and estimate the position of several non persistent transmitters. Their work consisted of a directional antenna, used to estimate the direction of signals, and an occupancy grid[40] based solution to store measurements and calculate movement from. By using Bayesian reasoning to update the grid and using heuristics to move towards grid cells to increase their probability of detection, the authors showed that this algorithm outperformed an organized and a random search. A similar development was undertaken by Scerri et al. [50]. In their work they also employed an occupancy grid inspired map which they used Bayesian reasoning to update. From this map each UAV calculated the point of which the swarm would gain the most out of measuring. This was done by assuming that the UAVs knew about the transmitted power of each transmitter. This assumption means that the transmitter in question was not hidden or unknown to the agents within. By sharing some information between the UAVs this approach could avoid searching already explored areas and because of the sharing of measurements the estimation of sources became more robust. Ferri et al. [17] also used a Bayesian updated occupancy grid. Their focus was in tracking multiple stationary indoor odor sources and showed that by using an occupancy grid, continued search in the area could be performed even after one or more odor sources was located. In [58] the authors used a Bayesian updated occupancy grid, and because of the nature of the work they combined a three part greedy heuristic to select where to explore next. By combining frontier exploration and RSS the authors created an indoor source seeking algorithm suitable for a single agent. Since only one agent was considered this algorithm was not implemented in this work.

Chapter 3

Model

This chapter will introduce the model and simulator used in this thesis. The model will describe how different concepts from chapter 2 is used to build a system which will try to answer the research questions presented in section 1.2.1. In this chapter the simulator will be described in broad terms.

3.1 Simulator

In order to test different search strategies a simulator has been created. This section will describe the different aspects of this simulator and the models which are used in the simulations. For the simulations to work a propagation model for the RF-signal and a movement model for the agents is needed. The simulations are used to compare different operations and collect statistics. For this thesis a general setting for searching was created with the intention of letting a number of simulated agents move around inside an enclosed area.

The simulation loop is given in algorithm 3.1. This algorithm describes how the simulator updates the different models.

3.1.1 RF propagation model

The goal of this thesis is to research different ways to use SI in order to detect a hidden transmitter. For this to be possible there need to exist an RF propagation model. To keep the propagation model simple, the signal is modeled as amplitude with noise added. The noise is assumed to only come from the receiver and there is no other noise in the environment, i.e. thermal noise. Dampening in the environment is included in the α variable. The equation for this amplitude is

Algorithm 3.1 simulate(agents, transmitter)

$numberOfSteps = 0$
repeat
  update transmitter {Let the transmitter update its internal state if needed}

  for agent a **to** Agents **do**
    update agent calling algorithm 3.2
  end for
  $numberOfSteps = numberOfSteps + 1$
until termination requirement is met
return $numberOfSteps$

shown in equation 3.1,

$$A_r = \frac{A_t}{\sqrt{d^\alpha}} \tag{3.1}$$

where A_r is the amplitude at the receiver, A_t is the amplitude the transmitter is sending at, d is the distance between the receiver and transmitter and α is a variable defining the propagation loss of the environment. Noise is then added to equation 3.1 to produce the final received amplitude, shown in equation 3.2.

$$A_r = \frac{A_t}{\sqrt{d^\alpha}} + N(0, \sigma) \tag{3.2}$$

This model was chosen because it accurately represents the signal propagation with line-of-sight to the transmitter when the α parameter is 2. In addition this model is a good approximation when dealing with elevated sensor platforms as was explained in section 2.1.1.

For this work the signal is assumed to be sending at all times which narrows the scope. This means that an agent can always sample and assume that the signal is present even if it is hidden in the noise. Thus intermittent transmission is not considered.

3.1.2 Agent model

In addition to the RF propagation model there is also a need for an agent model which defines the agents movements in the simulated world.

To simplify, the agents movement is restricted to two dimensional space. This restriction keeps the model simple and can intuitively be viewed as a group of UAVs flying at a constant specified altitude. This should not affect the overall conclusion since the comparisons performed are in between algorithms. The assumption being that if an algorithm performs better than another in two dimensions, it will also perform better in three dimensions. In algorithm 3.2 the update function for an agent moving randomly is shown. All agents must implement such a movement method, but the strategy decides exactly how the movement is performed.

Algorithm 3.2 updateAgent(time amount, world, transmitter)

$dx = rand(-1, 1);$
$dy = rand(-1, 1);$
$newX = dx * timeAmount * agentSpeed;$
$newY = dy * timeAmount * agentSpeed;$
if $newX$ inside world **and** $newY$ inside world **then**
  update current position
else
  select new position
end if

In algorithm 3.2 the arguments to the update function is the *time* since last update, the *world* representing the search area and the *transmitter* which the agents can query to obtain the amplitude at the current location. The *agentSpeed* is a known value for all agents which is the length an agent can move in a second. As will be shown later, the simulator is responsible for checking if the stopping condition has been met. For instance when search is the current goal of the agents the simulator will check if any agent is within range of the transmitter. It is important to note that the agent can not query the simulation environment more frequent than its own update. This is done to enable the simulator to be run as a single process and avoids many pitfalls related to multiprocess design. In a real world situation an agent would be able to sample RF signals much more frequently, but this can be included in the model by incorporating less noise. The implication being that if the agent could sample more often it could calculate a better average which would reduce the noise. At the same time, if the agents are allowed to sample less frequently, by reducing the noise, their perceived view of the world could be an acceptable simplification.

Another simplifying assumption made is that there is no collision between the agents. This assumptions means that there is no problem for all agents to move towards the same location at the same time. It also means that agents do not have to be aware of other agents to move. This assumption is made to make the strategies tested easier to implement and also aid in translation from abstract concepts to concrete strategies. The assumption is justified because most of the strategies would work if a collision avoidance layer was built into some part of the agents movements, without the strategies having to incorporate knowledge about that layer. This means that most algorithms would work, but they might perform slower than the results in this work would indicate.

The world model is a simple rectangle that the agents are not allowed to move outside. For this thesis the agents know there is a transmitter somewhere inside this area, but they have no other information.

To simulate a real search scenario all agents start in one corner of the search area. There they are distributed randomly within a small circle, so that all the agents do not move out from the same location. This is done to simulate a more realistic start. The random placement is the same for each agent strategy.

3.2 Search Strategies

In this section different search strategies will be presented and the termination requirement for the simulator during searching will be explained.

In this setting all the agents starts in one corner, because of symmetry which corner is not important, and are then able to move around searching for the transmitter. Each different strategy has a given way of moving each agent, but parameters such as movement speed, transmitter parameters and world size is kept constant over the different strategies. The termination requirement in this setting is for at least one agent to discover the transmitter with some degree of certainty. After this happens the simulation is stopped and statistics about the run is recorded.

By stopping the simulation just after a detection is recorded means that only the search strategy is compared. This makes comparisons between different strategies easier to perform. It also makes it easier by dividing the problem of locating the transmitter into two separate parts. However this has some downsides as several aspects of the agents as a group is not compared. If the simulation had simulated the whole problem, several aspects of the emergent behavior of the group would become evident. One example of this emergent behavior could be that, given two search strategies, strategy A is slower than strategy B, but because the agents are more grouped in strategy A they can come together faster and locate the transmitter faster if both strategy A and B used the same locating behavior. This would make comparisons harder, as one would need to analyze which parts of the agents behavior results in the emergent behavior. This would also mean that experiments would have to be controlled to a different extent, because statistics alone could not reject the possibility that the combination of different strategies is the problem and not individual strategies alone. In addition to these concerns the implementation would become harder as several parts would need to be working before comparisons could be made, e.g. both a search strategy and a locating strategy would need to be implemented before simulations could be run. By only comparing the strategies their individual strengths should become easier to recognize and the result should become more versatile as the comparison is more general.

In the searching configuration the simulator will run quite a few times for each set of parameters. This is shown as pseudocode in algorithm 3.3.

Algorithm 3.3 Simulation setup for the searching problem

```
Get Transmitter configuration
Get World configuration
Create t number of Transmitters locations by using a uniform probability
within the given World configuration
for agentConfiguration a to AgentStrategies do
  for numAgents n to NumberOfAgents do
    for transmitter t to TransmittersLocations do
      Create Set of agents from AgentStrategy a and numAgents n
      repeat
        Perform algorithm 3.1
      until Number of simulations times
    end for
  end for
end for
```

In algorithm 3.3 $NumberOfAgents$ is a list of different amount of agents to simulate, e.g. $(2, 3, 4)$ which would simulate with 2 agents, then perform the same simulation with 3 and lastly with 4 agents. The *AgentStrategies* variable is also a list which contains the different search strategies to apply. The reason for the innermost repeat is so that stochastic strategies can be compared. If no repetition was present the statistics collected would be based on one run where the initial random number state would dictate the outcome. By repeating the simulator, this random state will be mitigated by many runs and will give a clearer picture of the strategy. On the other hand, for a deterministic strategy this repetition is unnecessary as the number of steps in algorithm 3.1 would be identical.

The transmitter is placed at a random location using a uniform random distribution. The range of the transmitter is controlled by the experiment performed, but will be equal for all the different search algorithms tested.

3.2.1 Random strategy

The random search is one of two baseline strategies used to compare the other strategies. As noted in section 1.2.1 the random strategy will in a finite amount of time eventually find the transmitter. Since the strategy does not use any form of direction when searching, the strategy does not take in to account the information in the environment. From this one can see that no strategy should perform worse than the random search strategy. The pseudocode for the algorithm can be seen in 3.2 which can be explained as selecting a direction at random and moving until a wall is meet.

This strategy has some advantages and some disadvantages. The main disadvantage of the strategy is that it does not use any of the information in the environment to guide its movement. This makes the strategy simple to use, but it is expected to be slower than an equivalent strategy which make use of the information. In addition to being simple it is also highly distributed and uses no form of communication which makes it very scalable, which is desirable.

3.2.2 Call-out with Random

After some initial testing and some early results it was discovered that the random strategy described above performed quite well. By studying the random strategy in the graphical environment some thoughts around why this was (this is further elaborated on in chapter 5) appeared. From this understanding, a new strategy was created which should perform at least no worse than the random strategy. By incorporating what was perceived as the best parts of the random agent with some exploitation of the information in the environment this new agent was created.

This agent differs from the random strategy above by incorporating some of the information in the environment. By moving randomly until a sufficiently strong RSS measurement is made before acting on it, a more efficient search can be performed. When an agent measures above a certain configurable threshold it calls agents close by. The recalled agents will then stop their current path and turn towards the point it was called to. This call-out means that many agents will try to move towards a point of some interest. If the recalled agents are spread out before the call-out they will converge on the call-out point from many different angles. This results in a high chance that one of the converging agents will find the transmitter, if the call-out point is near it. Since the point it self is not that interesting a random element is added to each converging agent so that all agents move towards the vicinity of the call-out point. This random element diversifies the search around the area of interest and adds additional exploration of the immediate environment around the call-out point.

To allow for some movement a cool down period was added so that no agent currently participating in a call-out can perform or join another call-out before the cool down period is over.

3.2.3 Organized strategy

The organized strategy is the other reference implementation. This strategy shares the trait of not utilizing any information in the environment, and will perform an exhaustive search of the area. In a deterministic environment this strategy would on average always find the target quickest by searching every point in the area only once. The strategy will behave in a similar manner in the simulation performed, but because of the probability of not detecting the transmitter, even though an agent is within range, the strategy will search the area once before backtracking on the same path. The path is illustrated with the thick black lines in figure 3.1. When the agent reach the far right of the enclosed area they will move on the same black lines back towards the left, backtracking on the path. This ensures that the strategy will eventually terminate, but it is no longer guaranteed to be the quickest strategy as other strategies can utilize the information in the environment and can adapt to the noise.

When more than one agent uses this strategy they will be evenly spaced and move in a coordinated fashion. This coordinated search is illustrated in figure 3.1. This leads on to another problem with the strategy, the spacing between the agents can lead to the transmitter not being detected because of noise. This problem leads to a trade-off between utilization of the number of agents and the desire to detect the transmitter on the first pass. Both of these concerns are a consequence of wanting to find the transmitter as quickly as possible and both present different ways of approaching the problem. One could try the maximum spacing of agents, relying on low noise in the environment and quicker search time because each agent can cover more of the search area thus leading to a faster sweep. Or one could lower the spacing, taking a longer time to search the area once, but having a higher probability of detecting the transmitter between agents and not needing to backtrack.

In figure 3.1 a small sample of how five agents search is displayed. The large orange circle is the transmitter, the small red circles are the agents with green lines as the next destination and the black square is the search area. The total path the agents will travel is also shown in the figure, marked as black lines.

This strategy is expected to be quicker compared to the random strategy, because it searches each point only once [19]. But this comes at the cost of being more complicated and not nearly as scalable. In the implementation used a central controller is dictating the movement of each agent, this means that the strategy is not distributed and has a single point of failure. Furthermore it means that the strategy can not scale above what the single controller can handle. In a real world scenario this complication could be even more difficult to overcome as communication overhead would become a challenge. It is likely that some of these problems can be overcome by changing the controller from a central to a distributed model, but this would most likely lead to a higher communication

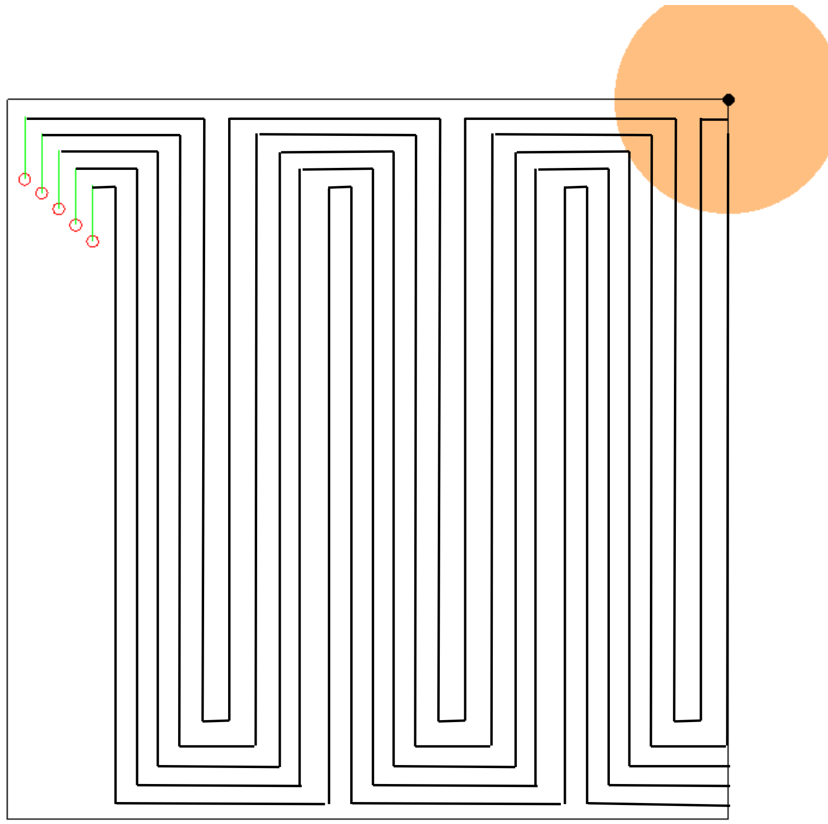

Figure 3.1: A sample of how the organized search is conducted with five agents. The red circles are the agents. The green lines is the next destination of the agents. The black lines illustrate the whole path that the agents eventually will travel and the orange circle is the transmitter.

overhead as the agents would need to communicate among themselves to become organized.

3.2.4 Chemotactic strategy

The chemotactic strategy is a direct implementation of chemotaxis shown in algorithm 2.1. This strategy is included so that a comparison between a strategy not utilizing the information in the environment can be compared to a strategy that uses the information. Even though the utilization is quite naive the strategy could still give some valuable insight into what is needed to search in a noisy environment. This strategy should be comparable to the random strategy, but it could be able to perform better as it actively tries to seek out the source. The strategy seeks out the source by moving in directions where it perceives an improvement on measurements.

One small alteration that was applied to this strategy was to give it the possibility of collecting several samples before calculating an average. The number of samples can be controlled and can be set to any desired amount. This gives the strategy some measure of noise reduction.

3.2.5 Lévy flight search strategy

Lévy flight is an interesting concept that could possibly result in a good search strategy. As explained in the background chapter, Lévy flight is a random walk inspired by nature. That inspiration is the basis of why it has been included in this work.

The strategy works by selecting a random direction and then select a random length where the length is calculated according to equation 2.10. In the same way as the random search strategy, this strategy limits itself to the search area, meaning some destinations will be cut short. This last part could become a problem as the distribution of lengths are not distributed according to Lévy flight when truncated, but this was seen as an acceptable solution since it should happen relatively infrequent.

Lévy flight is interesting because of its derivation from natural phenomena. Since many animals forage for food according to a Lévy flight pattern, there could be some evolutionary advantage compared to a uniform random or Brownian walk. Because of this, Lévy flight becomes very interesting as it might be that Lévy flight solves the problems of searching for a target by encapsulating the underlying mathematical basis for a search. This makes Lévy flight very attractive since its random walk nature makes it extremely distributed and easy to implement(to see how the Lévy flight distribution is implemented for this thesis, see appendix A).

3.2.6 Simple PSO strategy

The simple PSO strategy is a direct implementation of PSO explained in section 2.3.2. This strategy is included as a reference to how one of the basic SI algorithms perform.

The major change from the assumptions made about the simple PSO in section 2.3.2 to the implemented version is related to the experiment setup. In most versions of PSO there is an assumption or a requirement that all the agents are randomly placed, as uniformly as possible, to enable the algorithm to work. This assumption is not used in the implementation of the simple PSO since all agents must start in one corner as mentioned above. This will most likely affect the performance of the strategy because PSO is built around moving towards the best position gaining variability from this movement where a random agent might happen upon a better result. Since all agents start at the same location their only source of variability is the random noise. This noise could help the agents explore the search area by introducing a random element. When some agents follow the noise they could also move towards the transmitter, thus moving the swarm in the right direction. However if the distance is too large there is a greater possibility that the swarm can get stuck in a local minima.

This strategy makes use of the information in the environment by incorporating the received signal in its movements. In the normal setting where PSO is used as an optimization algorithm this information would always be of some use as there is no noise on the metrics used. The inclusion of noise could therefore become a problem for this strategy as it means that the local and global best could constantly move between samples dominated by noise.

One interesting aspect of including this strategy is the realization that it could potentially utilize each currently received signal in its update algorithm to cancel out noise. This notion is further explored in the next strategy which is an extension of the simple PSO search strategy.

3.2.7 Extended PSO strategy

By seeing how the simple PSO strategy operates several observations were made. First it was noted that cooperation among several agents was advantageous. Second, since the agents all start in one location, moving towards the best position does very little to direct the search. From these two observations a new strategy was created which tries to remedy some of the problems in the simple PSO strategy.

By changing how the agents estimate their next position, better movement could be accomplished. Instead of moving towards the best location in the swarm a travel vector is calculated which guides the agents. After this travel vector is calculated the agents move a certain distance along the vector before calculating a new vector. This ensures that no agent travel too far away if the vector was heavily influenced by noise. The vector calculation is illustrated in figure 3.2. To calculate the travel vector two other vectors are first calculated before being summed. The global best vector represents the best direction of the swarm as a whole, and the personal best vector is a locally calculated vector for each agent. Both vectors are calculated in the same way, but they differ in how they compare the agents. To calculate the global best vector, the agent with the current best signal estimate is found. This agent is then compared with all the other agents and vectors from the poorer agents to the best agent is created. These vectors are then added to create the global best vector, in figure 3.2 this is illustrated with the green arrows. If noise is taken out of the equation the global best vector will move every agent closer to the transmitter. The personal best vector is created in much the same way, but instead of comparing with the best agent, each agent compares all the other agents against itself. For each comparison the agent checks which agent, it or the other agent, has the best current estimate and creates a vector from the worse to the better agent. These vectors are again added and the personal best vector is created, this is illustrated in figure 3.2 with the orange arrows. The global best vector and the personal best vector are then added to create the travel vector.

The vector calculation above is influenced by PSO in the way PSO moves towards the global and personal best. By moving towards the global and personal best the swarm is moved towards the best position experienced by the swarm as a whole, but retaining some exploration by including the personal best. When considering that there is noise in the system the personal best vector is quite important as it can diminish the effect of spurious noise measurements. Since the global best could be selected on the basis of a noisy measurement, the local best can in some ways counteract that vector as only a small part of the local best vector is influenced by a noisy measurement.

To retain some stochasticity in the system, the travel vector is slightly modified by changing the direction according to a normal random. This aids in keeping the agent exploring, searching a larger part of the search space than it otherwise would.

To estimate the signal and reduce noise each agent collects some samples before calculating an average. This average is controlled by the experiment and can be changed to different amounts. This reduction in noise is contrasted with the time needed to obtain the samples. The longer an agent must stand still and sample, the more time it will use to find the transmitter. At the same time, if the agent moves while collecting, the longer it collects while moving the less representative it is for the area the agent is in when recalculating.

One thing that was noticed in the performance of the agent when experiments began, was that the ability to locate the transmitter was very dependant

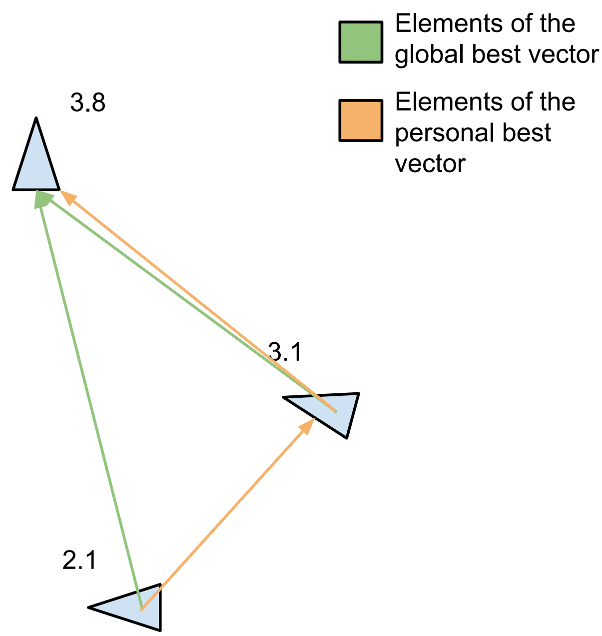

Figure 3.2: Illustration of the elements in the vector calculation of the travel vector for extended PSO

upon the lengths of its movements. This motivated the inclusion of Simulated Annealing (SA) like temperature changes. When the agent is far removed from the transmitter it should make large leaps to try to get closer to the transmitter. When it is close it should make small movement updates to not become too far removed from the transmitter. The inclusion of SA was then implemented into the agent which makes the movement dependant upon the perception of distance between the agent and the transmitter. For this reason using the threshold as the global maximum makes the inclusion of SA easier to implement. Noise can interfere with this estimate, but since the noise will be periodic and not constant it should only mean that the agent is trapped for a little while before moving larger leaps again. The movement length update equation is given in 3.3.

$$length_i = max(threshold - average_i, \theta_{min}) \tag{3.3}$$

In equation 3.3 $length_i$ is the length to move, $threshold$ is the threshold value given by the transmitters false alarm rate and noise, $average_i$ is the sampled average and θ_{min} is the minimum movement length.

3.3 Statistics Collected

To enable comparisons between different strategies some statistics are collected from each run. As shown in algorithm 3.1 the number of simulation steps are collected. The number of simulation steps is the most interesting statistic to collect, because it can be directly compared between different strategies and it gives a measure of the total time a strategy uses to complete. This statistics is also independent of actual time needed to perform an algorithm which can be both an advantage and a disadvantage. The advantage of it being independent of real time is that the simulation is not affected by scheduling or other such operations a modern operating system performs. This means that a direct comparison is possible since the number of simulation steps is only a measure within the simulation. The disadvantage of this measure is that it does not take into account the possibility of one strategy using more computational time calculating its next action compared to another strategy. This might mean a good strategy might be better because it uses a lot of computational power, but is not realizable for a real world system.

The statistical methods applied in this thesis are described in appendix B.

Chapter 4

Experiments and Results

In this chapter the results of the thesis is presented. First the experiment methodology is explained before the results are shown. The experiments are designed to answer the research questions presented in section 1.2.1 and are the result of running the simulator explained in section 3.2. For a more detailed explanation on how to setup and run the simulator, see appendix E. Discussion and analysis is performed in chapter 5.

4.1 Experimental Plan

In order to test different search strategies and highlight their strengths and weaknesses some experiments were created. The first experiment was designed to test the search strategies over a range of different transmitter placements. The experiments are run in a general search scenario. By comparing strategies in such a general search scenario quite a lot of data is produced. Average and standard deviation from the different transmitter placement is not possible to calculate because of the huge differences which can occur between different transmitter placements. Appendix B describes how the data is interpreted to create a good statistical comparison. In short terms the data is compared using a Paired t-test which results in a confidence interval. The confidence intervals are then summarized using a Binomial test.

Since all strategies have the same movement speed and experiences the environment in the same way, it was decided that only one search area was needed for the experiments. The Organized strategy is the search strategy most severely restricted by the size of the search area. Consequently, it became the basis to decide the size of the search area. If the size of the search area and the distance between the organized agents are correct, the agents will line up and do a singular

Variable	**Value**
$Amplitude_t$	500
False alarm rate	$1e^{-6}$
α	2
σ	1

Table 4.1: Transmitter configuration parameters for the search experiments

sweep of the search area. For this reason the search area is large enough to encompass the organized strategy, when the maximum amount of agents analysed can perfectly fit the area.

During the experiments all other configurations variables were kept similar. By keeping the other variables constant the results become easier to compare, even across different tests. The transmitter signal configuration was the exact same between the different tests. As stated this eases the comparison across different tests, but it also facilitates an easier discussion about the properties of the transmitted signal.

Since the system was created to be very easily reproducible, the exact configuration of each experiment is added in the appendix and will be referenced in the corresponding experiment setup.

The transmitter is setup with the parameters given in table 4.1. These parameters equates to the radiated signal shown in green in figure 4.1. As mentioned above this signal is kept consistent throughout the different experiments, which means that for the different tests the noise will be equal. The α value was chosen on the basis of the free space path loss model, described in chapter 2, and is consistent with perceived values experience in real world UAV testing[57]. The false alarm rate was also chosen for its relation to real world values. The σ value was chosen so that enough noise was introduced, this is also consistent with real world values. The amplitude of the transmitter was chosen so that the area covered by the transmitter was suitably large for the search area.

In figure 4.1 the signal is plotted showing how it will propagate from the transmitter. It is important to note that from a distance of above 200 the signal itself is quite hard to separate from the noise floor. This is important because it means that agents further away must use some technique to filter the signal to be able to move towards the transmitter. Another view can be seen in figure 4.2 where a three dimensional plot of the signal is shown. The emitter is located at the origin and the color represents the RSS. The figure illustrates the scale of the search area in comparison to the detectable range of the emitter, as an agent approaches the center it should be able to gain more information. It is not until the dark red areas where an agent is sufficiently close to determine a detection

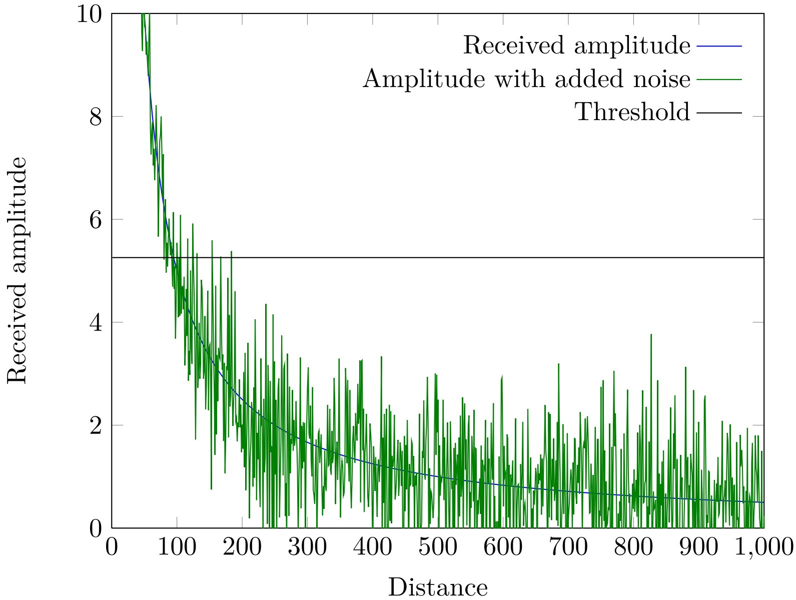

Figure 4.1: Illustration of the received amplitude at a given distance, with noise and showing the threshold

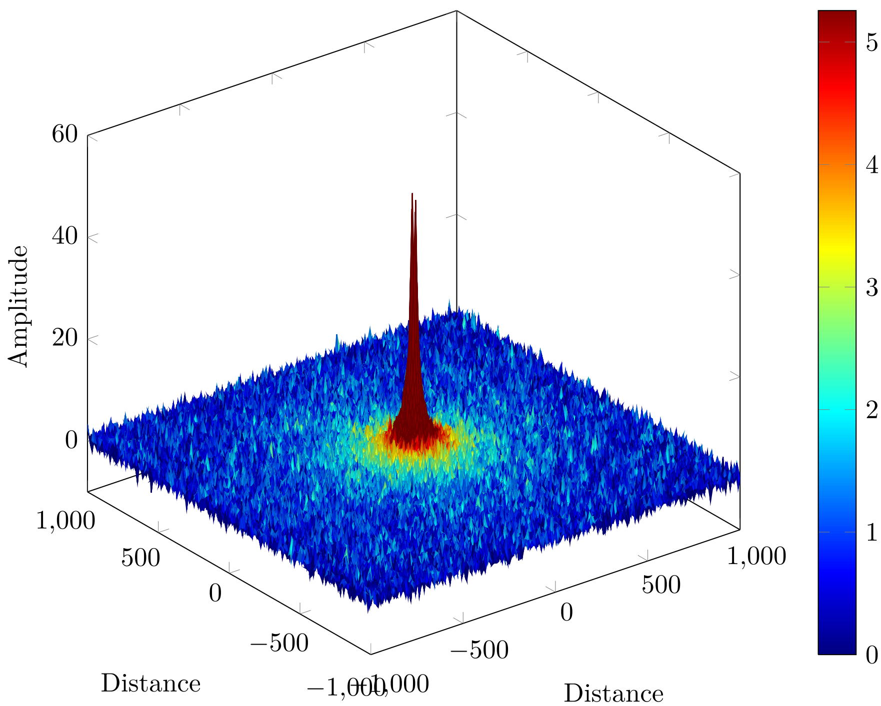

Figure 4.2: Illustration of the amplitude shown in figure 4.1 as a three dimensional plot in an area 2000x2000. In the figure only the area shaded in the dark red is above the threshold and thus is recognizable as a signal.

Size	Threshold coverage(%)
1590 x 1590	1.25%

Table 4.2: The search area size in relation to the estimated signal coverage and the threshold coverage.

using a false alarm rate at 10^{-6}.

In table 4.2 the search area is shown. The transmitter estimate is derived from the discussion above using information from table 4.1 and the plotted signal in figure 4.1. The threshold coverage in the table illustrates the percentage of the area in which an agent is able to detect the transmitter with the given certainty, expressed as the false alarm rate. The agent travel speed, the number of repetitions per transmitter and the number of transmitters were all kept constant.

In order to be able to compare the different search strategies a number of different runs were set up. A list of randomly placed transmitters were generated. The same list was used for each agent configuration and was static throughout the experiment. The transmitters all share the same configuration as mentioned, which results in comparable statistics. The results from these runs are then compared through the statistical analysis shown in appendix B.

Included in the result is both the simulated time needed for each strategy to detect the transmitter as well as the total length traveled. Simulated time is necessary since time is the objective of interest in the search. The simulated time gives an understanding of the real world time needed and because it stems from the simulator it is unbiased. The time used also aids in illustrating the energy needed. As explained in section 2.2, this work envisions UAVs capable of hovering. This means that time is also a metric of energy used as the UAVs would need to use energy to stand still in one location. Traveled length is also included to identify if any strategy behaves different in regards to how it moves. Length is interesting because it can give an idea about the energy expended by each strategy, but it can also help illustrate unusual behaviors. One example of such behavior could be a strategy which uses a long time estimating the gradient towards the transmitter. Such a strategy may use more time than a strategy which is actively searching, but because it is standing still it could expend less energy.

4.2 Experimental Setup

As explained in the previous section the experimental setup is kept consistent. For this reason the shared variables will be explained before the results are shown in the next sections.

Variable	**Value**	**Description**
Seed	42	Global seed for the pseudo random generator. This variable will initialize the global random generator which ensures that the experiments are repeatable. The basis for this value is quite random, the only real necessity is that it is configurable to enable repeatable experiments.
Simulations	250	Value representing the number of repetitions performed per transmitter. The number of simulations and the number of transmitters are closely linked with the quality of the statistics the simulator produces. Because of this the numbers were set reasonably high, this contrasts the need for good statistics and the available computational time.
Number of transmitters	250	Number representing the amount of different randomly generated transmitter positions. This number decides how many randomly generated positions are generated. Se above point about why this value was chosen.
Speed	45	The speed which the agents are allowed to move by. This number represent the number of movement units that the agents can travel per perceived second. When running graphically the movement is given per real world second. The reason behind this number is also quite arbitrarily. As long as the number is equal for all agents no one will have any sort of movement advantage.
Number of agents	(2, 5, 10, 15)	This list represents the number of agents to create during a run. For each element in the list the simulator will create a run which consists of the given number of transmitters above with the number of simulations for each transmitter. The numbers for this list was chosen so that strategies which benefit from more agents are given a chance.
Confidence	0.95	This value represents the confidence used when comparing the different strategies. In appendix B this value is used to calculate the confidence level and shown with α in equation B.5, B.6 and B.7.
Output types	"SimulationSteps" or "LengthTraveled"	This variable is a list of the different outputs that one wants to extract data from. The variable has two possible values, as shown. Simulation steps represents the time each strategy used. While length traveled is the combined length of all agents tested, e.g. if Random is run with two agents, the length is the sum of these two agents.
Output method	"RoundRobin" or "RelativeMean"	The output method decides how the results are compared. The round robin alternative compares all strategies against all other strategies. Relative mean individually compares each strategy where the first amount of agents, e.g. 2 above, is compared. This, output type and confidence are all related to the output produced.

Table 4.3: A description of each variable and value of all shared variables for the search problem

In table 4.3 a description of each variable and its corresponding value is shown. The table is complete in regards to simulator specific variables, but agent specific variables are not shown. With these values the transmitter is placed according to figure 4.3. These settings corresponds to the JSON code shown in appendix C, where the JSON code can be used with the simulator to recreate the results shown below.

The values found in table 4.4 shows the specific configuration variables used for the different search strategies. The values was chosen after preliminary testing of each strategy testing different values. For this reason it is important to note that these values are not optimal so more could be done in this respect. However, an optimization of all the different algorithms employed is outside the scope of this thesis. Before the experiments were run, each strategy was tested with several different parameters before the ones shown in table 4.4 were selected. The testing was done graphically, and was done to give each strategy a reasonable default value.

Agent strategy	Values	Description
Random	N/A	Because of the way the random strategy operates there are no configuration variables.
Call-out with Random	**Number of samples** 30 **Steps between recall** 200 **Threshold for call-out** 3.9 **Random element** 1.0	• The number of samples describes how many samples the agent should collect before calculating an average to use to estimate the current RSS. • Since this strategy will call on all other agents to converge on suspicion of a transmitter there is a need for a cool down period. This ensures that no other agent performs a recall until at least a certain time has passed. • The threshold describes the amplitude an agent has to experience before performing a call-out. • To ensure that the agents do not crowd over one another a slight random element is added. This random element is here one standard deviation from the point at which a call-out occurred.
Lévy	α **parameter** 2.0 **Scale parameter** 100	• The α parameter specifies the alpha value used to generate the Lévy flight. • The scale parameter scales the flight so that it makes larger leaps.
Organized	**Distance between agents** 150	• The distance between agents is the maximum distance any agent is from any point in the search area.
Extended PSO	**Number of samples** 30 **Maximum amplitude** 5.2565 **Smallest step** 25 **Largest step** 50	• The number of samples taken before a new estimation of the direction to travel in is taken. • The maximum expected amplitude which is used to calculate the current ”temperature”. • The smallest step the agent will take, depending on the current ”temperature” the agent might try to take a smaller step, this ensures that it tries to move some to avoid local minima. • The largest step the agent will take, this is again used to limit the agent somewhat in its movement, it should not move to fast away from an area.
Chemotaxis	**Number of samples** 30 **Step length** 40	• The number of samples taken before a new estimation of the direction to travel in is taken. • The step length used in movement.

Table 4.4: A description of each agent specific configuration variable.

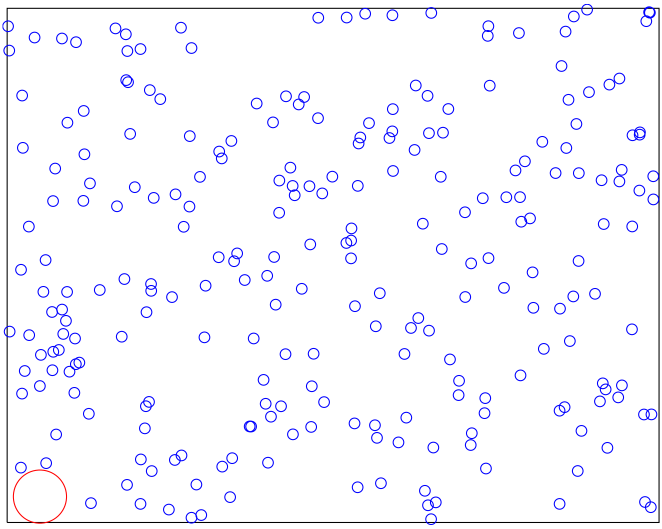

Figure 4.3: Transmitter positions generated from the values in table 4.3. The blue circles are transmitter positions and the red circle encompasses the area where the agents start from.

4.3 Results

In table 4.5 the simulated time results are shown for the search experiment. In the table the overall performance of the agents are compared. The metric for this comparison is the amount of simulated time a strategy has used to find the transmitter. The data shows comparisons between all strategies for the different amount of agents. The data shown in bold is the amount of agents used by the strategies. The next number shows the probability of the strategies having equal performance. The next number is the amount of times the strategy along the row outperformed the strategy on the column. While the last number shows the amount of times the strategy along the column outperformed the strategy along the row.

The probability, of the strategies being equal, is calculated using a Bernoulli test. As mentioned the probability shows the certainty of two strategies being equal. It is important to note that while one strategy can have what appears to be a significant advantage, it is the number of tests that end in a draw that influences the probability the most. In other words, two strategies may appear different, but if the number of draws outnumber the total number of times the strategies were different, i.e. one strategy beat the other, the probability of those strategies being equal is significantly increased. Another important point to note is that when the probability is low, but the numbers are quite similar, e.g. one strategy beat the other 25 times while the other strategy won out 27 times, the strategy with most victories can be declared better. This is because the certainty with which a victor is declared is so high that when deciding between two strategies the one outperforming the other must be strictly better. The last point to note is that the results are not transitive.

The results shown are calculated according to appendix B and the comparison is done with every one against everyone meaning that equation B.7 is used to calculate the confidence level.

	Call-out with Random	Lévy	Organized	Extended PSO	Chemotaxis
Random	**2** (1.0, 19, 72) **5** ($1.41e^{-1}$, 16, 117) **10** ($3.29e^{-1}$, 26, 102) **15** ($6.74e^{-3}$, 55, 89)	**2** (0.0, 188, 5) **5** (0.0, 188, 10) **10** (0.0, 209, 0) **15** (0.0, 234, 0)	**2** (0.0, 47, 143) **5** ($2.85e^{-8}$, 24, 143) **10** (0.0, 12, 190) **15** ($1.83e^{-2}$, 37, 104)	**2** (0.0, 179, 40) **5** (0.0, 169, 44) **10** (0.0, 229, 1) **15** (0.0, 240, 0)	**2** ($2.28e^{-14}$, 9, 174) **5** ($2.28e^{-14}$, 52, 131) **10** ($3.33e^{-16}$, 103, 84) **15** ($5.30e^{-5}$, 130, 25)
Call-out with Random	N/A	**2** (0.0, 200, 2) **5** (0.0, 209, 8) **10** (0.0, 227, 5) **15** (0.0, 232, 2)	**2** ($1.11e^{-16}$, 70, 118) **5** (0.0, 78, 119) **10** ($2.49e^{-2}$, 22, 118) **15** ($2.78e^{-15}$, 95, 90)	**2** (0.0, 187, 33) **5** (0.0, 197, 19) **10** (0.0, 232, 5) **15** (0.0, 240, 1)	**2** ($3.85e^{-4}$, 1, 150) **5** ($1.36e^{-8}$, 94, 74) **10** ($1.11e^{-16}$, 149, 39) **15** (0.0, 170, 26)
Lévy	See above	N/A	**2** (0.0, 0, 234) **5** (0.0, 1, 236) **10** (0.0, 1, 240) **15** (0.0, 2, 237)	**2** ($6.38e^{-9}$, 120, 49) **5** ($5.91e^{-10}$, 75, 97) **10** (0.0, 211, 1) **15** (0.0, 240, 0)	**2** (0.0, 0, 244) **5** (0.0, 0, 243) **10** (0.0, 0, 242) **15** (0.0, 0, 242)
Organized	See above	See above	N/A	**2** (0.0, 234, 5) **5** (0.0, 231, 9) **10** (0.0, 242, 1) **15** (0.0, 243, 1)	**2** (0.0, 77, 147) **5** (0.0, 148, 64) **10** (0.0, 202, 14) **15** (0.0, 200, 29)
Extended PSO	See above	See above	See above	N/A	**2** (0.0, 0, 232) **5** (0.0, 0, 242) **10** (0.0, 0, 243) **15** (0.0, 0, 243)

Table 4.5: Results for the general test comparing simulated time. Each strategy is compared against every other strategy. The bold number represents the number of agents compared. The following numbers are, the probability of the two strategies being equal, the number of times the strategy along the row performed better than the column and the last number is how many times the strategy along the column outperformed the row.

	Call-out with Random	Lévy	Organized	Extended PSO	Chemotaxis
Random	**2** (1.0, 19, 72) **5** ($1.41e^{-1}$, 16, 117) **10** ($3.76e^{-1}$, 25, 102) **15** ($6.74e^{-3}$, 54, 90)	**2** (0.0, 188, 5) **5** (0.0, 188, 10) **10** (0.0, 209, 0) **15** (0.0, 234, 0)	**2** (0.0, 47, 143) **5** ($2.85e^{-8}$, 24, 143) **10** (0.0, 12, 190) **15** ($1.83e^{-2}$, 37, 104)	**2** (1.0, 0, 0) **5** (0.0, 168, 49) **10** (0.0, 224, 2) **15** (0.0, 239, 1)	**2** (0.0, 7, 184) **5** ($7.76e^{-12}$, 45, 132) **10** (0.0, 98, 95) **15** ($6.32e^{-14}$, 123, 59)
Call-out with Random	N/A	**2** (0.0, 201, 2) **5** (0.0, 209, 8) **10** (0.0, 227, 5) **15** (0.0, 233, 2)	**2** ($1.11e^{-16}$, 70, 118) **5** (0.0, 78, 119) **10** ($2.49e^{-2}$, 22, 118) **15** ($8.88e^{-16}$, 96, 90)	**2** (1.0, 0, 0) **5** (0.0, 196, 27) **10** (0.0, 230, 5) **15** (0.0, 236, 2)	**2** ($1.79e^{-5}$, 1, 156) **5** ($6.38e^{-9}$, 87, 82) **10** (0.0, 138, 62) **15** (0.0, 157, 49)
Lévy	See above	N/A	**2** (0.0, 0, 234) **5** (0.0, 1, 236) **10** (0.0, 1, 240) **15** (0.0, 2, 237)	**2** (1.0, 0, 0) **5** ($2.78e^{-15}$, 74, 111) **10** (0.0, 202, 7) **15** (0.0, 236, 2)	**2** (0.0, 0, 244) **5** (0.0, 0, 244) **10** (0.0, 0, 243) **15** (0.0, 0, 244)
Organized	See above	See above	N/A	**2** (1.0, 0, 0) **5** (0.0, 228, 10) **10** (0.0, 242, 2) **15** (0.0, 242, 2)	**2** (0.0, 73, 151) **5** (0.0, 144, 78) **10** (0.0, 187, 33) **15** (0.0, 185, 44)
Extended PSO	See above	See above	See above	N/A	**2** (1.0, 0, 0) **5** (0.0, 0, 239) **10** (0.0, 0, 243) **15** (0.0, 0, 243)

Table 4.6: Results for the general test comparing length traveled where each strategy is compared against every other strategy. The bold number represents the number of agents compared. The following numbers are, the probability of the two strategies being equal, the number of times the strategy along the row traveled less than the column and the last number is how many times the strategy along the column outperformed the row.

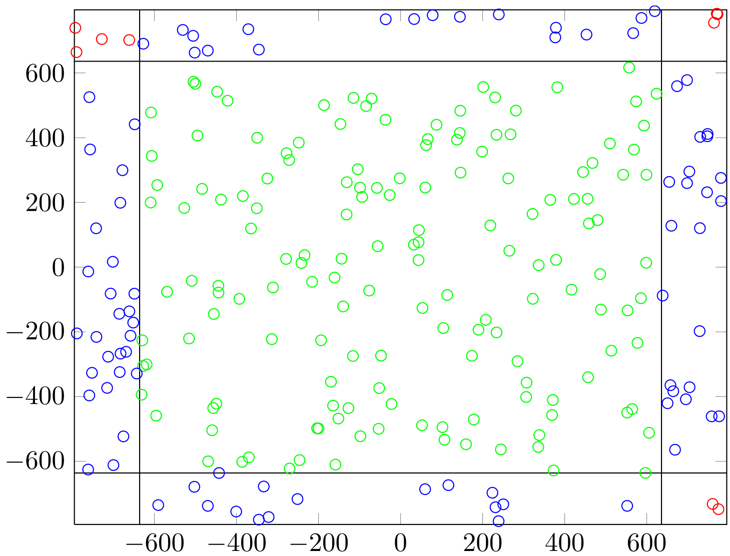

Figure 4.4: Transmitter positions divided into discrete areas. The blue circles represents the transmitter positions which are within 10% of the total distance to an edge. The red circles represents transmitter positions which are within a 10% distance to a corner. The green circles are the remaining circles which are not within range of any edge.

In addition to total time searching, used as a metric for comparison in table 4.5, the combined length traveled for all agents was also compared. The result of this comparison is found in table 4.6. The length was compared in the same way as time was compared, following appendix B.

To explore the results above, further analysis was performed on the data in table 4.5. By dividing the search area into some discrete smaller areas along the edges the comparison was performed again to see whether any of the agent performed better in specific circumstances regarding the search area. To illustrate this discretization figure 4.4 shows how the search area was divided.

The result of this discretization is only shown in brevity in this chapter and the full result is added in appendix D. As can be seen in figure 4.4 there are few transmitters placed in the corners. This means that the statistical basis is quite limited. For this reason the data concerning the corners will not be analysed any further.

As mentioned, the discretized results will not be shown in full here, below are the results which deviated from the general results in such a way that the outcome

Comparison	**Data**
Call-out with Random compared to Chemotaxis	**5** ($5.36e^{-5}$, 25, 34)
Call-out with Random compared with Organized	**15** ($1.33e^{-4}$, 6, 52)
Random compared to Call-out with Random	**15** ($2.23e^{-1}$, 42, 3)

Table 4.7: Data showing disparity between the general results shown in table 4.5 and the results for the edge analysis shown in table D.2.

was changed. Since the probability of the strategies being equal is shown, any change in the number of transmitters could give a change in probability. Because of this the limit in the following have been placed at 95% certainty in the results. This means that the probability of the agent being equal have to be less than 0.05 and subsequently this means that to deviate it is not enough to reduce the probability, it must move over this 0.05 boundary.

Data in table 4.7 are highlights of data in table D.2. As explained the data shown deviates from the general results and are therefore highlighted here. The highlighted data should be interpreted as the first strategy named is the second number in the triplet and the other strategy is the third number in the triplet.

In table 4.8 the highlighted results from table D.3 are shown. The interpretation used is the same as for table 4.7.

One of the goals of this thesis is to show that using more than one agent, i.e. a swarm, is beneficial to searching. In order to show this the statistics in figure 4.5 was created. The figure shows the relative increase in time when using more agents. The data is a comparison where each strategy is compared against itself with two agents. The statistics is produced by comparing each transmitter configuration with the corresponding transmitter configuration when more agents are added. When all relative times are calculated for each discrete number of agents an average and standard deviation is produced. Table 4.9 illustrates how the statistics is created. The statistics is produced from the same underlying data as table 4.5, but is analysed in a different way. It is important to notice that figure 4.5 does not compare the different strategies against each other. A relative increase for one strategy does not mean it is faster than another, it is only faster compared to itself.

Since an average taken from all the different transmitter positions is not possible, as explained above. One other option is to select a few positions and only calculate the average for those. This would give a comparable result, but the

Comparison	**Data**
Random compared to Call-out with Random	**5** ($2.8e^{-9}$, 0, 112) **10** ($3.43e^{-5}$, 0, 101)
Random compared to Chemotaxis	**10** ($5.24e^{-8}$, 52, 57)
Random compared to Organized	**15** ($5.95e^{-1}$, 13, 62)
Call-out with Random compared to Organized	**2** ($3.39e^{-10}$, 58, 56) **5** (0.0, 69, 58) **10** ($9.85e^{-1}$, 19, 44)

Table 4.8: Data showing disparity between the general results shown in table 4.5 and the results for the middle analysis shown in table D.3.

		Number of agents			
		2	5	10	15
Transmitter	T_1	(Reference)	$\frac{T_{1,5}}{T_{1,2}}$	$\frac{T_{1,10}}{T_{1,2}}$	$\frac{T_{1,15}}{T_{1,2}}$
	T_2	(Reference)	$\frac{T_{2,5}}{T_{2,2}}$	$\frac{T_{2,10}}{T_{2,2}}$	$\frac{T_{2,15}}{T_{2,2}}$
				$\vdots$	
	T_{250}	(Reference)	$\frac{T_{250,5}}{T_{250,2}}$	$\frac{T_{250,10}}{T_{250,2}}$	$\frac{T_{250,15}}{T_{250,2}}$
	Average	1.0	$\frac{1}{250}\sum_{i=0}^{250} T_{i,5}$	$\frac{1}{250}\sum_{i=1}^{250} T_{i,10}$	$\frac{1}{250}\sum_{i=1}^{250} T_{i,15}$

Table 4.9: An illustration of how the relative averaged data, shown in figure 4.5, is created.

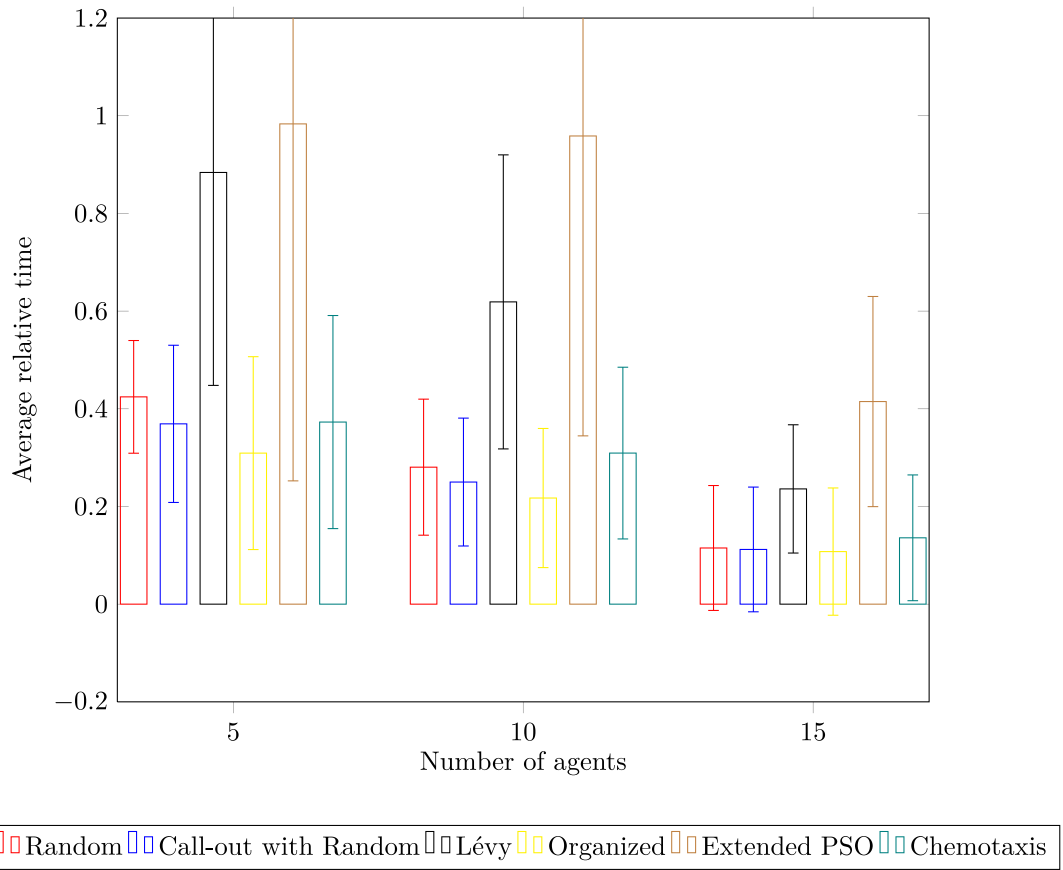

Figure 4.5: This figure shows the average relative time for each strategy. The relative time is a comparison between the average time it took for each strategy to find the transmitter with two agents. The figure shows that all strategies benefit from having more agents.

problem is that selecting a transmitter position would always be biased. For instance, one could select a transmitter position at random. If the transmitter is to far to the right, the Organized strategy would always be at a disadvantage. The same problem also arise for the other strategies in different context. As will be dicussed in the next chapter, each strategy have some places it has affinity for. Because of this such single position selection was not used in the comparison between the different strategies.

Chapter 5

Evaluation and Conclusion

In this last chapter an evaluation and discussion about the results from chapter 4 is presented. This chapter will go into detail about the results and will give both an explanation along with a discussion. The last part of this chapter will be devoted to the conclusion and future work.

5.1 Evaluation

The goal of this thesis was to *gain a deeper understanding of the potential for swarm intelligence to solve probelms related to detecting a hidden signal transmitter*. In order to show this, several experiments were run with different search strategies.

5.1.1 Time analysis

Table 4.5 contains a lot of information and thus require some interpretation. By comparing different strategies, from the simple random walk to the more complex PSO strategy, further insight into searching for a hidden transmitter is obtained. A point of note, for this work a confidence of 95% is required. That means that a superior strategy must have more victories and the difference between the values must be below 0.05.

Random

Starting with the Random strategy, it performs similar to the Call-out strategy. This is quite natural as the Call-out strategy acts random until some threshold. For 2, 5 and 10 agents the Call-out strategy does not outperform the Random strategy. With 15 agents, the Call-out strategy outperforms the Random strategy. The Call-out strategy, as implemented, should not perform any worse than the Random strategy. The small difference is down to the way in which the Call-out strategy utilizes the information in the environment. Further elaboration in future work should yield some interesting results.

The Random strategy perform very well compared to the Lévy strategy. This is most likely attributed to the long walks across the search area for the Random strategy. The Lévy strategy performs small jumps, contained in one area, followed by longer leaps, but this exploration is slower than stretching out. This result is supported by theory surrounding Lévy flight[45], which shows for many cooperative foragers, continually performing long leaps is superior. For this work it was decided that it would be interesting to compare the normal Lévy flight $\alpha = 2$[61]. This was decided because the Random strategy behaves as the cooperative Lévy flight $\alpha \sim 1$[45], meaning it would give little extra information comparing the Random strategy with the cooperative Lévy flight.

When comparing the Random strategy with the Organized strategy, it would seem like the Organized strategy is superior. The results are quite unanimous even when 15 simultaneous agents are used. This indicates that the random strategy is not effective enough. There are some indications in the results that with 15 agents the Random strategy is starting to become more effective, yet the Organized strategy is better. The reason for this result is most likely down to how the Organized strategy utilizes all the agents effectively. Because the Random strategy often will travel to places where other agents have explored before, some of the effectiveness of the strategy is lost. In this respect the Organized strategy performs quite a bit better, because it is able to evenly space the agents, thus never visiting the same spot twice.

Comparing the Random strategy with the Extended PSO strategy, it is clear that the Random strategy performs better. This indicates that the Extended PSO has problems separating noise from the signal and moves slowly towards the transmitter. This result can to some extent, be viewed when running the simulator in graphics mode. The reason for the performance advantage is most likely because the Random strategy is able to search in a much larger portion of the search area. When the Extended PSO agents is far away from the transmitter and standing still estimating the gradient towards the transmitter is difficult. While the Extended PSO agents stands still, or moves very little, the Random strategy is spreading out and moving to potentially new areas. This is a very interesting result because it indicates that when searching for a small object in a large area, covering as much of the search area as possible is very important. This is an indications that using the Random strategy as a metaheuristic to get closer to the transmitter could improve search times.

The Chemotaxis agents were another interesting comparison for the Random strategy. When using few agents, 5 and fewer, the chemotactic strategy performs well. Then a shift appears, and when using a large amount of agents the Random strategy performs well. This result is an interesting combination of the strengths and weaknesses of both strategies interacting. When there are few agents the Random strategy is not capable of spreading enough. This means that the agents must use more time to search the same amount of space. The Chemotaxis strategy on the other hand, tries to utilize the information in the environment to search for the transmitter and performs quite well. Compared to the result above, with the Extended PSO strategy, the chemotactic strategy has a good mix of random exploration and exploitation of the signal. If it can not sense the transmitter it will move randomly, but the strategy will try and exploit the signal when close. This combination makes the Chemotaxis strategy superior at low numbers, before the Random strategy has enough agents to spread out. This result indicates that when there are few agents cooperating, some heuristic must be employed to find the transmitter faster. At some point the Random strategy becomes saturated with enough agents and can effectively spread out and search the area. When this transition occurs the Random strategy becomes more effective compared to Chemotaxis.

It is important to point out that the Random strategy is heavily influenced by the initial random conditions. The exploration achieved by the Random strategy is predicated on the initial conditions and are not guaranteed. This is important because it means that predicting the outcome is hard. There is nothing in this strategy which prevents all the agents from moving to the same location, resulting in very poor performance. The results indicate that over a large sample size these random conditions are not deteriorating the performance too much. Because of this, adding some form of coordination in the initial spread can be important in real world experiments.

Call-out with Random

Because of the results above, comparing the Random strategy with different other strategies, it was observed that spreading out was important. The reason for this observation is most likely attributed to the increased probability of getting close to the transmitter, thus increasing the information available. The Call-out strategy was developed from this and to make it simple, it was paired with just a random approach to see how it performed. The Call-out strategy is a strategy shifting approach, trying to balance the need for exploration with the need for exploitation. The strategy shift occurs when the need for exploitation is larger than the need for exploration.

Against the Lévy strategy the Call-out performed well. This is reasonable as the Call-out strategy should not perform worse than the Random strategy.

Comparing the Call-out with the Organized does show some improvements, compared to the Random strategy. At few agents the Organized strategy performs better. However, as the amounts of agents increase the performance is not only equal, but at 15 agents the Call-out strategy performs better. This performance increase, compared to the Random strategy, is a result of the Call-out strategy using the information in the environment. Since the Call-out strategy calls all agents when a certain threshold is met, it can avoid situation where the Random strategy just barely misses the transmitter. This slight increase in performance is enough to improve compared to the Organized strategy. The result is a further indication that getting closer to the transmitter before changing strategy can be beneficial. It also shows that, by spreading out randomly, one can approach the efficiency of coordinating such separation. This is also interesting because of the simplicity of the Call-out strategy compared to what the Organized strategy would require in a real world scenario.

This strategy also seems to be quite novel within this field of research. Only Chemotaxis used any form of exploration/exploitation technique, and none of the RF related, show in section 2.3.3, used such a technique.

This strategy, as with the Random strategy, is very dependant on the initial random conditions. Because of the preliminary design of the Call-out strategy no further steps are incorporated to avoid this limitation. There is nothing in this strategy which would hinder such additions and it could be added later.

Lévy

The Lévy strategy performs almost as expected, most of its flaws has been explained above. Yet it is interesting to see that even the Lévy strategy perform better than the Extended PSO. The reason behind this is the same as the reason for the superior performance of the Random strategy. It shows that spreading out can be more important than trying to estimate a direction towards the transmitter. By spreading out and getting closer to the transmitter it is easier to estimate the signal gradient and not get trapped in local minima.

Organized

The Organized strategy performs quite well compared to all other strategies. However, it is interesting to note that with few agents, even this strategy performs worse than the chemotactic strategy. With only two agents it is quite clear that Chemotaxis manages to both spread out and utilizes the information in the environment maximally compared to the other strategies. This further shows that with few agents, some approximation must be done. It also shows that it can be easier to utilize the information in the environment than to search exhaustively.

For many agents the Organized strategy is outperformed by the Call-out strategy. This is likely attributed to the organizational work the strategy has to perform in the beginning. Both the Random and the Call-out strategy is able to move towards potential transmitters in the middle of the search area right away. During this time the Organized strategy has to spread out and move to the initial starting positions. The time spent doing this warm up is enough for the strategy to be outperformed by the Call-out strategy.

Extended PSO

The worst performer, the Extended PSO strategy, did not function as well as expected. Since the strategy tries to exploit all signals and coordinates movement according to these measurements, it was assumed that the strategy would be able to estimate the gradient towards the transmitter. The strategy was designed to overcome the difficulties of estimating the gradient towards the transmitter. The idea was to use several agents with several different samples to minimize noise and find the transmitter. The results here indicate that this strategy was not good enough. Since the agents potentially started quite some distance away from the transmitter trying to compensate for the noise is difficult. This strategy illustrate that some approximation is necessary before trying to estimate the gradient, when noise is present.

Chemotaxis

Chemotaxis performed quite well compared to most strategies. The strategy shows that some amount of randomness to aid exploration together with an attempt to estimate the direction of the transmitter works well. The strategy performed well with few agents which most likely indicates that the other strategies performed poorly with few agents. Compared with Random and Call-out this strategy shows that if massive parallel exploration is not possible, some other measure is necessary. Compared to the Extended PSO strategy it is clear that a balance between exploitation and exploration is necessary. Chemotaxis illustrates that using random exploration can work quite well.

5.1.2 Length analysis

The results shown in table 4.6 mirrors the results above quite closely. Using less time is also an indication that the strategy as a whole has traveled less. The only result that stand out is the Extended PSO strategy with two agents. Even though the strategy uses a lot more time, it is equal with all other strategies when it comes to length traveled. This indicates that with two agents the Extended PSO strategy moves very little in wrong directions, but it does so very slowly. The reason for this movement is that the two agents will always move towards the best agent. Because of the way the strategy is designed, the two agents will move very little unnecessarily, but they will also stand still much of the time. With more agents the strategy will have more random perturbation and the favourable travel length disappears.

5.1.3 Divided area analysis

Analyzing the results further lead to the separation of areas as shown in figure 4.4. The simplified results has some interesting items, and some further insight into the performance of the different strategies was obtained.

Edge

From the edge analysis it is clear that the Organized strategy performs better than the other strategies. Many of the results from the discussion above have been changed. This change is quite natural as most of the other strategies, especially the ones containing random movement, do not follow edges well. Since the Random strategy will turn whenever it hits an edge, wall following does not naturally occur. The same is true for Chemotaxis which will also try to turn 180° when faced with a wall. In addition, the Organized strategy will always be the specified distance away from the wall and thus is guaranteed, in the tests performed above, to find any transmitter along the edges. The same insight is also apparent in the comparison between Lévy and Extended PSO. Since the Lévy strategy has the same propensity to avoid edges as the other random strategies it is also at a disadvantage along the edges. The Extended PSO on the other hand tries to follow the signal, and will follow walls if that is necessary.

To illustrate the difference further, all strategies were run, without a transmitter, for an extended period of time. During this time, the movement of each strategy was recorded, producing figure 5.1. In the figure, the a priori probability of where each strategy will search is illustrated. The figure shows that the Organized strategy will search in perfectly rigid lines. The Random strategy has a much higher probability of finding anything in the corners. While the Lévy strategy has a much more uniform probability. Chemotaxis is quite evenly spread out, but prefers the corners. Extended PSO seems to move slightly towards the middle, but compared to the other strategies is very stationary.

The reason for the Random strategy preference for corners is due to how it selects the next direction to move in. Since the strategy will choose uniformly between all possible direction it has a higher chance of choosing a direction that keeps the strategy in the corner. However, this a priori probability is a bit misleading. The probability is collected over a long period, but the search for a transmitter is over relatively quickly compared to this period. It is therefore unlikely that this, a priori probability, has much effect on the propensity of the strategies to search where the probability is highest. However, the figures help illustrate that because of the task involved in the search relying on prior probability might not be the best match. The figure illustrates how the search would be conducted if no transmitter is present. A desired effect would be the even spread as is seen in the random inspired strategies as well as the Organized. Such an even spread indicates that the strategy searches the area completely which is important for exploration. As mentioned several times in the previous sections, exploration is very important to achieve lower search times. Figure 5.1 shows this exploratory property of each strategy.

Middle

When looking at the comparison with transmitters in the middle, more changes have occurred. The notable difference is that the Call-out agent performs better than the Organized strategy at even fewer agents. This is another sign of the good characteristics of spreading out in the search area. Since the Call-out strategy has a larger chance of getting close to the transmitter, when it is placed further away from the edges, it naturally performs better. Another change is the result for the Random strategy. The full results shows some improvements in the Random strategy, compared to the Organized, and for 15 agents the improvement is statistically significant. This shows that the strategies containing random movement is at quite a disadvantage along the edges. It also shows that combining the effective spread, inherent in the Random strategy, with some other means of utilizing the information in the environment result in a better outcome.

5.1.4 Research question I

One of the many interesting properties of SI is the ability for many simple agents to cooperate when solving a task. This property relates to the first research question, *do several agents perform better than a single agent.* It is therefore an important result to show that using several agents to search for a hidden transmitter is more beneficial than using few. By searching for a hidden transmitter with different amounts of agents this result could be obtained. In figure 4.5 a comparison was performed to see if this assumption was valid. The figure shows that for most strategies, adding more agents improved search times. In addition the figure shows that the relative gain increases as more agents are added indicating that using several agents together can decrease search time. One thing to note about the result is the standard deviation. For some of the agents the standard deviation is quite large which indicates that mean values might not be the best representation of the performance. Unfortunately because of limited time more experiments could not be run, but a natural progression from the result is to see when each strategy would become saturated with agents. One thing to notice about the result is that the organized strategy shows, for 15 agents, perfect

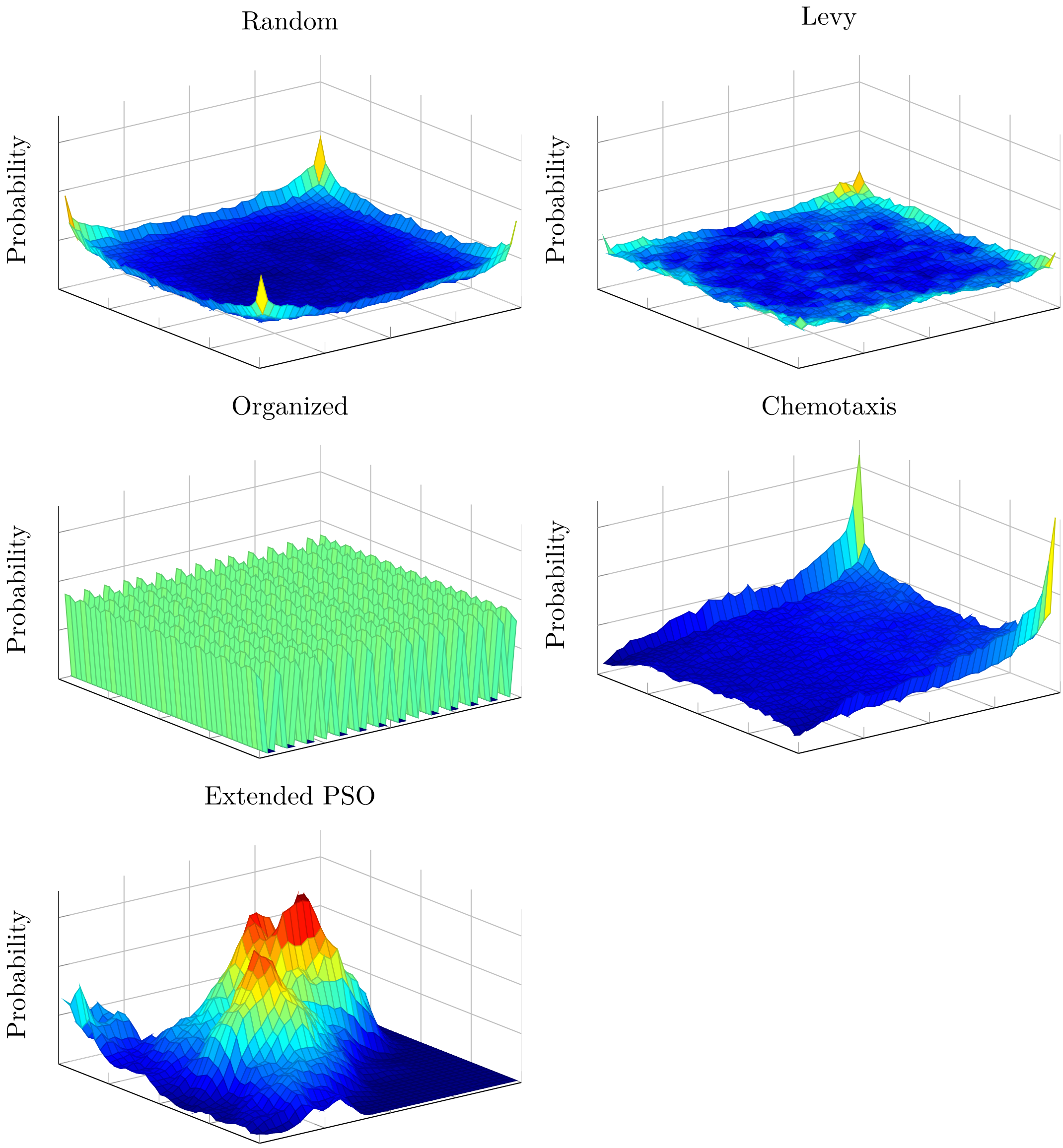

Figure 5.1: This figure displays the a priori probability of where the different strategies will search when no transmitter is present. This probability distribution illustrates the most likely places the different strategies will search. The probability is normalized and the axis have the same values for all strategies, making direct comparisons possible. The starting position is the corner pointing out towards the reader. For all strategies 15 agents was used, as can clearly be seen in the Organized figure.

utilization. Because of the size of the search area and the current transmitter radius, the organized strategy can not become more effective by adding more agents. The only strategy not to have a constant decline, in relative time spent searching, is the Extended PSO strategy. Since this strategy always tries to utilize the information in the environment it can easily be misguided by noise. As can be seen in the figure, this strategy has the highest variance which is a result of the noise. By adding more agents to the Extended PSO strategy there are more agents that can be influenced by noise. Because of this, when the transmitter is far away progress towards the transmitter can be slow. However, even for this strategy adding enough agents does increase performance, eventually. The results indicates that several cooperative searchers can perform a search quicker than few or single searchers. Thus answering the first research question.

5.1.5 Research question II

In regards to the second research question, *whether a random walk exploration could compare to an organized search*, several interesting points should be made. In this context, random walk applies to all strategies which uses some form of random process to explore the search area. From the evaluation above, the Random strategy performed worse than the Organized strategy. This would indicate that it is difficult for a random walk to outperform an organized search. However, several strategies have been included in this thesis and many of them contain random movement. It is therefore important to point out that using the Call-out agent, moving randomly did perform better when enough agents were added. In addition the chemotactic agent performed better with few agents, compared to the organized search. This indicates that, by using some of the information in the environment together with random exploration a good strategy can emerge. This is important because as explained in the background chapter, an organized search is both difficult to construct and vulnerable to faults within the system. It is difficult to organize, because each agent needs to coordinate with all other agents. The vulnerability comes from the fact that if an agent fails to perform, or is taken out, the system must take over the agents duty. In an organized search this means that the other agents must regroup and be certain that they search the positions where the malfunctioning agent was supposed to search. In all of the random strategies tested, none of these issues would present a challenge. If an agent failed in the Random strategy, the performance might drop, but the other agents would not have to take into consideration that another agent failed. In addition, as mentioned above, the Organized strategy as presented can not scale any further, with the given search area and transmitter radius. While the Call-out strategy would be able to scale almost indefinitely. In addition, if the transmitter radius is unknown the organized search would be difficult to use. The

Organized strategy, as implemented, is reliant on a given distance to space the agents. If the radius of the transmitter is smaller than this distance the agent would be at risk of not detecting the transmitter. The Random strategy, on the other hand, would not experience this problem and can function regardless of the radius. Because of these considerations, a random walk can compare relatively favourably to an organized search.

5.1.6 Research question III

The last research question, posed in section 1.2, was, *can a technique which does not use any information available in the environment compare to a technique which uses that information*. When evaluating the different strategies above many interesting points in regards to this was mentioned. Probably the most important indication that using the information in the environment is needed, is the comparison between the Random strategy and Call-out. The Call-out strategy moves randomly until it measures something above a given threshold, in other words using the information in the environment. If no transmitter is present, the Call-out and the Random strategy would move in the same fashion. Since the Call-out strategy performs slightly better than the Random strategy and performs better against other strategies, using the information in the environment is shown to be advantageous. In addition one can see that Chemotaxis performs quite well which also indicates that using the information in the environment is meaningful. However, it is important to point out that this exploitation of the transmitted signal must be balanced against exploration when the distance between agents and transmitter is large. This is evident in the poor performance of the Extended PSO strategy which almost performs worse than all the other strategies. This discussion should therefore be seen in light of the above discussion, about random movement. Both are important, but together they become even better. It is important to explore the search area, which can be done effectively with random movement, but at the same time it is important to use the information in the environment. How the information is used is important and should be considered in future work.

5.1.7 Using SI to detect a hidden transmitter

The goal of this thesis was to *gain a deeper understanding of the potential for SI to solve problems related to detecting a hidden transmitter*. In relation to this several points are worth making. The discussions above have to some degree gone into different areas where SI can often perform well. As mentioned in the introduction, SI is often associated with large systems containing several cooperating agents. These agents are often simple, yet together perform complex tasks. The answers to the research questions above, indicates that in some way, detecting a hidden transmitter can be done with methods inspired by SI. The Call-out agent is simple in its design and performs quite well. The number of Call-out agents searching simultaneously should be able to scale. Both of these traits are often found in other SI systems, which again should give some indication. Since this has only been an initial attempt to view a complex domain any definite conclusions are hard to draw, but using inspiration from SI within this domain can certainly reveal new interesting solutions and insight.

5.2 Threats Against Validity

Many of the algorithms described in section 2.3.3 are not included in this thesis. Some are explained in the text, but others were not included even though they could potentially work within this thesis framework. Because this is an initial foray into a complex domain, it was decided that the included strategies should be quite basic. This should give a deeper insight into some of the more pressing problems of the domain, yet still allow inspiration from other works to influence the direction taken.

Because of limited time and limited computational power some compromises have been made in order to produce results. This section will address these issues and explain why the different choices were made. To ease readability this section is separated into a RF section, an agent movement section and a simulator section.

5.2.1 RF model

As explained in section 2.1, simulating a RF signal is an extensive task. Because of this difficulty, the work done in this thesis has used a simplified propagation model. The assumptions being that there is line-of-sight to the transmitter, only the amplitude of the signal is used, that there is only the free space loss affecting the signal and that the antennas used are isotropic.

Any elevated platform will be limited in where it can operate. For this reason any UAV envisioned for the task of detecting and locating a RF signal will not be able fly in ideal conditions. This means that the platform will have to operate near terrains containing trees, buildings and other objects that will affect a propagating signal. In addition to this the transmitter will not always be placed in such a way that it will give a line-of-sight to the platform. Because of this, the assumption used in this work, is a simplification of many real world situations. Modelling the RF signal assuming line-of-sight will therefore result in conclusions that can not be generalized to all areas. However, because the assumption makes modelling a RF signal easier and can be valid when using an elevated platform, see section 2.1.1, it was decided that such a model would be suitable for this work.

A RF signal is quite complex. There are many components which can be extracted and, as explained in section 2.1, different locating techniques use different properties. For this work only the amplitude was used. This simplifies the simulator, but there are some problems. The reason for this choice was the desire to utilize PDOA in future work. The simplicity and the cheap real world sensors also makes the choice compelling. The drawback is that potentially more agents needs to be deployed to achieve good location estimates. By using SI techniques this drawback can be reduced.

Connected with the assumption, that there is line-of-sight, is the assumption that only the free space loss is affecting the signal. In most varied environments there will be several different effects modifying the propagation of a RF signal. In addition to the free space loss, there will be several effects, among them multi-path, which can strongly influence the signal. Multi-path is constructive and destructive interference between multiple waves reaching the receiver. It occurs when two or more radio signals reach the receiving antenna, due to for instance reflection. As shown in section 2.1.1 these effects are quite reduced when elevating the sensor platform, but the effect should still be taken into consideration. Because this work is intended to be used on UAVs it was decided that assuming very little multi-path is a good approximation to real world conditions.

Assuming that the antennas used are isotropic also limits the results. In the simulation run, both the receiver and the transmitter antenna is assumed to be isotropic, meaning it measures the same value in all directions. This simplification makes the signal easier to model. In a real world setting most antennas are not isotropic which should be considered. The assumption was made because this work comes as a preliminary before real world experiments. As such there is currently no selected antenna which means that there is no antenna diagram to emulate. However, the antenna is none the less important in search and must therefore be considered.

Another implicit assumption was that the frequency is known. As mentioned in 2.1.1, the frequency of a radio signal is bound to some given values. In addition to this, it is assumed that the agents are capable of detecting the transmitter, meaning they have to know which frequency it is operating in. In real systems the receivers are limited to only seeing a portion of the frequency spectrum. If the agents have to search a large portion of the frequency spectrum, they could do so by sweeping the spectrum. Alternatively the agents could collaborate by searching a portion of the spectrum each. By searching in this way, the agents would have to use a lot more time interpreting the spectrum analysis and the possibility of misinterpretation would be a large possibility. To simplify the RF model this spectrum analysis was not considered. This limits the scope to some extent, but if the frequency spectrum is limited this assumption could still hold.

Relating to the RF model is the assumption that the transmitter is continuously sending and that there is only one transmitter. Many real world transmitter only send a signal intermittently. This conserves power and since data is usually produced in bursts, e.g. a cellphone needs only transmit when a phone call is under way, continuously sending is rarely done. For these reasons, assuming that the transmitter is always sending makes the strategies simpler. If the transmitter was sending intermittently an organized search would be much harder to perform and any strategy not using the information in the environment would end up with a severe disadvantage. By incorporating an always-on transmitter this work becomes simpler, but intermittent sending must be considered at some point. Because of the complexity of the RF domain this trade-off was deemed necessary to be able to get some initial work done.

Another RF related assumption made was that there is only one transmitter. In a real world environment more than one transmitter is usually the norm. Most devices often have multiple receivers and transmitters, such as the Bluetooth and WiFi in a mobile phone. Any operation trying to detect and locate such a transmitter will be required to handle several transmitters. Since this work serves as an initial attempt at understanding the complexities of detecting a transmitter it was decided that including one transmitter should be the first goal. Once an understanding of the difficulties of detecting one transmitter had been undertaken, more transmitter could be added in later work.

In this work the transmitter was assumed to be stationary. Since the transmitter is standing still, one can feasibly calculate the gradient towards it and just move in that direction. With a moving transmitter this would not be possible and the strategy employed would have to take this into consideration. Many real world objects that one would be interested in, both detecting and locating, will most likely be moving. However, there are cases where the transmitter will not be moving. One example of such a stationary target could be detecting a mobile phone of a person trapped in an avalanche. To tackle one problem at a time, this work does not consider moving transmitters. This makes the strategies easier to design, but it limits the scope of the strategies.

Because of these assumptions the RF model is limited in scope. The assumptions will limit the generality of the result, but it can make this work serve as an initial entry to a large and complex domain.

5.2.2 Agent movement

In this work the agent movement has also been simplified. By assuming that the agents only move in two-dimensional space and that the agents do not collide, the resulting scope is limited. In addition the agents are assumed to be able to sense their position, they can move perfectly with no uncertainty and they have the same acceleration and speed in all directions.

By containing the movement to two-dimensional space the agent movement becomes easier than it would in three dimensions. Each strategy is therefore easier to implement and rapid testing of different strategies was enabled. However, by assuming that the movement is in two dimensions the results are not directly applicable to any airborne platform. As the work on this thesis progressed it became clear that a comparison between different search techniques was desirable. Because of this, the assumption was kept in place. The rational behind this is the difference between two and three dimensions should not impact the search algorithms. Since all algorithms had the same basic movement the difference between them would be inherent in the algorithm and not the movement. It is also worth noting that this can be viewed as UAVs flying at a specific altitude.

One of the assumptions made about the search area, that the agents work within, is that it is rectangular. This assumption is made because it is envisioned that this work could be used on UAVs which fly in an unobstructed area. For simple search scenarios this assumption might be correct, but for most real world situations this will not hold. In many settings there will be environmental obstacles which will hinder movement. If the area to search is a city, large buildings may stand in the way making it difficult to perform an organized search. In other situations some information may be available which restricts the area of interest to some complex polygon. In both of these situation it is not clear how the different strategies might perform. If one assumes a situation where there are two circular areas connected by a thin rectangle, strategies like random may search the just one circle for far too long before some agents moves to the other circle. In such a situation it is also difficult to perform an organized search and care would be needed to handle this. For these reasons, the rectangle search area severely limits the results. The assumption was deemed acceptable for this work because of the unknown nature of the problem. By starting with a simple known shape, for the search area, some of the basic difficulties could be worked on before embarking on different challenges. In addition early real world testing could be performed in a unobstructed rectangular airspace.

Assuming that the agents do not collide is a different matter. Collision can have a severe impact if the agents try to move on top of each other all the time and a strategy which tries to separate the agents would therefore have an advantage. This assumption was included to make the transition from abstract idea to implementation easier. To mitigate this, each agent could use some form of collision avoidance on a lower level than the operation of the search strategy. If such a technique was used the strategies could be kept as they are now and should still function. This design has often been used in robotic movement[7, 18, 67] and as such the assumption that there is no collision should be acceptable for early testing.

By assuming that the agents know their exact position and can move perfectly makes this extremely hard to transition to real world robotics. The assumption that the agents know where they are is a simplifying assumption and is a topic of active research. Since these are assumptions that all strategies would have to deal with in the same way, their removal should not impact the individual performance. None of the implemented strategies tries to reason about these subjects which should mean that none of them are at a disadvantage.

5.2.3 Simulator setup

Several assumptions are made about the simulator, relating to both the statistics produced and the simulation setup.

One assumption made about the simulator setup is that the agents always start in one corner. Since the area tested is quadratic, starting in the corner means the result should be the same no matter which corner is selected. One criticism of this however is that the agent could be started along one of the edges. In a real world scenario starting in a corner or along one edge of the search area is quite realistic. Therefore it is natural to allow the agents to start in a corner, but it should also be tested along an edge. In the current setup this was not done because of limited time. The simulator described in chapter 3 used several days to produce relevant results of statistical significance. This meant that full testing iterations took some time and it was decided that proper statistics, i.e. more repetitions, was more important than testing different starting locations.

Another limiting factor of this work is the statistics produced. Because of limited time, only a small number of transmitters and repetitions per transmitter was tested. This has an effect on the conclusions made, since the limited sample size means that analysis are not as robust as they could be. As was noted in chapter 4, the corner analysis was not considered because of the small number of transmitter placements. The limited statistics is a result of time limitations, the simulator often needs to simulate several tens of thousands of steps in order for a strategy to find a transmitter. In addition the simulator must repeat each simulation, resulting in over a million simulations consisting of several thousands of simulations steps. On the hardware available this took around five days to complete.

5.3 Conclusion

There are many interesting aspects to conclude this work with. Below are some of the key insights discovered.

- Few agents should prefer exploitation over exploration. Many agents should prefer exploration over exploitation.

When few agents $N < 5$ are searching, exploitation of the information in the environment should be preferred. Chemotaxis was the best strategy with few agents. This strategy has a good combination of exploitation of the signal information and exploration. Since few agents are limited in their capability to search a large area quickly, exploitation is necessary. When the number of agents is increased organizing the search performs better. The Organized strategy had lower search times than the other strategies for $N >= 5$ to $N <= 10$. When the number of agents is increased further, organizing the search efficiently seems to become problematic. If many agents $N >= 15$ are searching, exploration becomes more important. This is shown by the Call-out strategy outperforming every other strategy for many agents.

- A novel search algorithm, the Call-out strategy, is proposed.

The Call-out strategy, developed in this thesis, showed the best performance for many agents. The Call-out strategy is a basic strategy shift, using different strategies at different parts of the search. It utilized random exploration to increase the probability of quickly getting close to the transmitter. While at the same time used the information in the environment when possible.

- Several SI agents decreases search time.

The results shows that several agents decreases search time. This increase in performance is the result of having several cooperating agents. However, it is important to point out that the performance does not need to come at an increased cost of organizational complexity. The most important conclusion to reach is that the task of searching for a hidden RF transmitter is suitable for many cooperating agents using simple, amplitude sensors. This means that many techniques derived from SI can be used in this setting.

The insight gained in this thesis builds a basic understanding of the key challenges within this field. The work is therefore a good starting point for further exploration.

5.4 Future Work

There are several aspects of this work that can, and should be, explored further. Several exciting directions in both the RF and the agent strategy domain could be explored.

Within the RF domain several problems relating to the transmitter could be studied. In this work the transmitter was stationary. Therefore introducing a moving transmitter could be quite interesting. Both movement during search and movement after the detection of the transmitter would be interesting to study. In many real world settings the transmitter is not continually sending a signal. For this reason exploring search relating to transmitter sending intermittently would be quite interesting. In addition to these two aspects, exploring the use of several transmitters would be highly beneficial. Since several transmitter are usually found in places of interest, creating a search strategy which can handle this is of high importance.

In this work it was assumed that the transmitter would transmit on a known frequency. Changing this assumption and using several agents to cooperate on discovering such a transmitter would be interesting. In many real world scenarios not knowing the frequency of the transmitter is quite usual. Having a system which could handle a large frequency spectrum would therefore be highly beneficial. One interesting direction to take this in, would be to distribute the task among the agents. If each agent would only have to search a small portion of the spectrum, it could work faster and the system could become more effective. This would however require care so that the system could handle failure when analysing the spectrum.

While the agents in this work behaved very predictable, such reliability is not always possible in the real world. It would therefore be interesting to see how each of the strategies would handle failure. By adding random failure to an incidental agent, the strategies would be tested in more realistic situations. This failure would also highlight which strategies copes better with problems surrounding the agents. This information would be valuable before transitioning to real world UAVs.

Designing the ultimate search strategy for any given problem will probably never happen. It is therefore important to explore as many options as possible before selecting the preferred strategy. Exploring different search strategies designed by hand can be a slow and time consuming process. For this reason it would be interesting to explore the creation of search strategies through the use of Evolutionary Algorithms (EAs). If the search problem and agents could be expressed in a way that an EA could work with, exploring different possibilities would become much quicker. This also opens up the possibility of using several objectives as optimization parameters, so called multi-objective optimization[10, 18]. Using several different objectives could lead to systems which can handle several different aspects of the search. In addition, EAs have a tendency to explore possibilities not envisioned by regular methods[18], meaning new strategies, hard for people to imagine, could emerge.

Another unexplored direction, is the incorporation of information about the search area. In a real world scenario the area of interest is usually not rectangular. For this reason it would be interesting to test different strategies and how they manage unusual search areas. Exploring different, irregular formed, search areas could lead to insights into both what information is relevant for the search and which properties of the search area most affect the agents.

In section 1.1, three different phases were proposed. Different aspects relating to phase 1 is described above, yet the two other phases are not mentioned. For phase 2 it would be interesting to see how many agents are needed to converge, which is connected with phase 3. In addition looking at how strategies would handle the transition from phase 1 to 2 would also be interesting. Phase 3 is necessary to research. In phase 3 it would be interesting to see how agents can cooperate and share information. As mentioned in section 2.1, all the location techniques need paired measurements to estimate the position. This information sharing should be as distributed as possible to increase fault tolerance, yet all agents must have it. A distributed calculation would also be an interesting direction to move towards, but problems such as redundancy, fault tolerance and performance are unknown. On top of this comes which method to use. PDOA requires relatively inexpensive equipment, but if the position estimates are greatly inferior to TDOA this trade-off might not be desirable.

The Call-out strategy was created for this work after inspiration from preliminary results. In this work, the Call-out strategy was only used in conjunction with random movement towards the call-out point. Further work into this strategy would therefore be quite interesting. It would be interesting to see which strategies one should shift to in addition to optimizing the strategy. Such optimization related to when a call-out is performed could give important insight, in addition to decreasing search time.

Appendix A

Understanding and generating the Lévy flight distribution

This appendix is included to aid in understanding how to generate Lévy flight and try and help avoiding some of the confusion around the difference between Lévy flight and the Lévy distribution.

This chapter and its content is thanks to Boye Annfelt Høverstad.

Lévy flight is a random walk with a distribution that is described as

$$P(l) \sim l^{-\mu} \tag{A.1}$$

for $P(l > 1)$ and it should not be confused with the Lévy distribution which is defined as

$$\sqrt{\frac{c}{2\pi}} \frac{e^{-\frac{c}{2(x-\mu)}}}{(x-\mu)^{3/2}} \tag{A.2}$$

for $x \geq \mu$. The Lévy distribution defines a tail heavy function which is centered around some value and trailing of on both sides. While Lévy flight is a simple tail heavy power law function. Both functions are shown in figure A.1.

To generate values for the Lévy flight one must convert the function into an inverse cumulative distribution function because of the relation between the uniform distribution and the other distribution.

The relation ship is characterized by

$$x = F^{-1}(u) \tag{A.3}$$

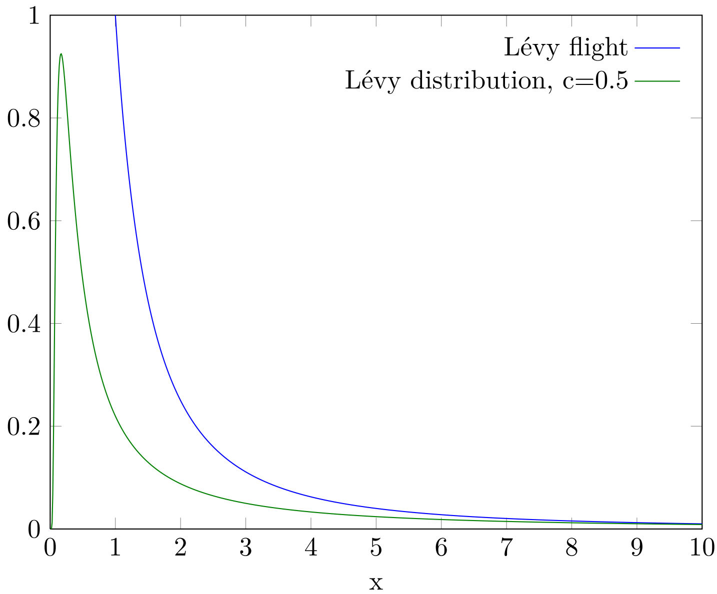

Figure A.1: A comparison of Lévy flight and the Lévy distribution

where F is the distribution to generate from and u is a uniformly distributed number between $[0, 1]$.

Since the Lévy flight is described for $P(l > 1)$ and it is need as a cumulative distribution function defined for $P(X \leq x)$ subtracting 1 from the Lévy flight in A.1 gives the needed function given as

$$P(l) = 1 - l^{-\mu} \tag{A.4}$$

The final step is to create the inverse cumulative distribution given as

$$P^{-1}(u) = (1 - u)^{-1/\mu} \tag{A.5}$$

which will create our Lévy flight using a random uniformly distributed number, u, as input.

Appendix B

Statistical comparison of search results

This appendix will explain in detail how the search results gathered from the experimental runs, when searching, are compared, how that comparison is calculated and the statistical background it is built upon.

As explain in section 3.2 the data collected when searching can be quite a lot. Because of the many repetitions for each individual transmitter configuration and the many repetitions for placing a transmitter randomly, simply calculating the mean and standard deviation does not give much information. In addition to this a large disparity occurs because of the random placement of the transmitter in a possibly very large search area which means that calculating an average for a single agent configuration results in a standard deviation as large as the mean. For this reason a different comparison was implemented, namely a Paired t -test[32]. This comparison not only gives a confidence interval, but it also enables a reduction in the disparity resulting from the random placement of the transmitters.

The table below, table B.1, illustrates how the result from the search experiments look. The data in table B.1 are random values and not actual result values, but the table illustrates how data looks after collection.

For each transmitter configuration many repetitions are performed for each agent configuration. The last line of the table illustrates how much the data can change from one transmitter configuration to the next showing that calculating a mean and standard deviation is not meaningful.

Since the goal is to compare *Agent configuration #1* with *Agent configuration #2* the first thing to compare is the results from each transmitter configuration. This is done by using the Paired t-test to produce a confidence interval for each

Transmitter configuration	Agent configuration #1	Agent configuration #2
1	[500, 600, 10000, 2000 ...]	[450, 700, 9000, 2200 ...]
2	[700, 200, 100, 3000 ...]	[850, 300, 90, 2000 ...]
3	[100, 300, 500, 7000 ...]	[180, 900, 2300, 8200 ...]
4	[3400, 1900, 1800, 40000 ...]	[2000, 2000, 1900, 32000 ...]
⋮		

Table B.1: Example output from the searching experiments

Transmitter configuration	Confidence interval(95%)
1	(-0.034, -0.002)
2	(-0.057, 0.013)
3	(0.78, 0.944)
4	(-0.15, 0.23)
⋮	

Table B.2: Result after applying the Paired t-test to the result of table B.1

transmitter configuration.

The first step is to subtract the agent configurations against the other agent configuration, e.g. take the array under *Agent configuration #1 Transmitter configuration 1* and subtract the array in *Agent configuration #2 Transmitter configuration 1*. This creates a new array $Z_i = A_i - B_i$. On this new array perform the Paired t-test given in equation B.1,

$$\overline{Z}(n) \pm t_{n-1,1-\alpha/2}\sqrt{\widehat{Var}[\overline{Z}(n)]} \tag{B.1}$$

where $\widehat{Var}$ is given in equation B.2, $\overline{Z}$ is the average of all Z_i values and $t_{n-1,1-\alpha/2}$ is the critical point for the t-distribution with n as the number of samples for the distribution and $1-\alpha/2$ is the degree of certainty or confidence level.

$$\widehat{Var}[\overline{Z}(n)] = \frac{\sum_{i=1}^{n}[Z_i - \overline{Z}(n)]^2}{n(n-1)} \tag{B.2}$$

The result is a table like B.2, note the confidence intervals are also random and are not calculated from table B.1.

Since the data in table B.2 is still not that easy to interpret another conversion is performed. The confidence intervals can be interpreted to be able to tell if the null hypothesis can be rejected. In this setting the null hypothesis is the

Transmitter configuration	Hypothesis test result
1	Rejected, interval does not contain zero
2	The null hypothesis can not be rejected
3	Rejected, interval does not contain zero
4	The null hypothesis can not be rejected
⋮	

Table B.3: Result after testing the null hypothesis on the confidence intervals from table B.2

assumption that there is no difference between the different agent configurations. The null hypothesis can be rejected if the confidence interval does not contain zero and it can then be rejected with a confidence of 95%. Results are shown in table B.3 where the data from table B.2 is evaluated.

The final step is to perform a Binomial test[64] on the results in table B.3. Because similarities in performance means it is hard to separate two strategies, the binomial test is performed to give a confidence in the difference of the strategies. By assuming that it is equally probable for the null hypothesis to be rejected as not rejected one can use the Binomial distribution to calculated the probability of the results in table B.3. By summing the number of rejections, where rejections refer to data in table B.3, the probability of the results holding under the hypothesis that the difference test is inconclusive can be calculated as,

$$b(x;n,p) = \binom{n}{x} p^x (1-p)^{n-x} \tag{B.3}$$

where x is the number of rejections, n is the total number of hypothesis tests performed and p is the probability of rejection which, as stated above for the test performed here, is 0.5. In other words, the two hypothesis are expected to tie most of the time. To calculate the significance of the results obtained, $Pr < 0.05$, equation B.4 is needed.

$$F(x;n,p) = Pr(X \geq x) = \sum_{i=x}^{\lfloor n \rfloor} \binom{n}{i} p^i (1-p)^{n-i} \tag{B.4}$$

By applying equation B.4 to table B.3 an answer to whether or not the two agent configurations are equal or different in performance can be given. For the example data in table B.3 the calculated probability is $Pr = 0.3125$, using equation B.4, which means that the test is inconclusive, because $Pr > 0.05$, and the significance of the difference is not large enough.

To compare several agent configuration together the same procedure as shown above can be used, but the confidence level needs to be calculated differently depending upon the way the comparison is performed[32]. When comparing one and one the confidence level can be calculated as,

$$1 - \frac{\alpha}{2} \tag{B.5}$$

When comparing one against a list of others the level needs to be calculated as,

$$1 - \frac{\alpha}{k - 1} \tag{B.6}$$

where k is the number of other configurations to compare with. The last possibility is to compare every one against every one which needs a confidence level calculated according to,

$$1 - \frac{\alpha}{\frac{k(k-1)}{2}} \tag{B.7}$$

where again k is the number of configurations to compare.

The current implementation allows each of these possibilities, but if not otherwise state every strategy is compared against every other strategy.

Appendix C

Configuration

Below is the specific JSON configuration used in the experiments shown in chapter 4. The code is included so that the specific results could be recreated. The file shown below can be used together with the simulator to recreate the results. The meaning of the different variables is explained below.

Listing C.1: Configuration code used to replicate the results shown in table 4.5

```
{
        "seed" : 42,
        "simulations" : 250,
        "numberOfAgents" : [2, 5, 10, 15],
        "numberOfTransmitters" : 250,
        "agentSpeed" : 45,
```

In the code shown above, the simulator specific variables are shown. The variables are described in table 4.3.

The ”world” parameter describes the world. The following representation allows the search area to become rectangular and does not require it to be quadratic.

```
        "world" :
        {
                "worldX":1590,
                "worldY":1590
        },
```

The transmitter parameter describes the necessary parameters for the transmitter to be used. The ”effect” is the amplitude at the transmitter. The ”alpha” characterizes the propagation model. ”Sigma” is the variance used to generate the noise. The ”threshold” variable describes the probability of a detected signal being an actual signal and not random noise.

```
        "transmitter" :
        {
                "type":"AlphaTransmitter",
                "effect": 500,
                "alpha":2,
                "sigma":1,
                "threshold":0.999999
        },
```

The different agent variables are described in table 4.4.

```
        "agents" : [
                {
                        "type" : "RandomAgent"
                },
                {
                        "type" : "RandomDirAgent",
                        "numMeasurements" : 30,
                        "ticks" : 200,
                        "threshold" : 3.9,
                        "radians" : 1.0
                },
                {
                        "type" : "LevyAgent",
                        "alpha": 2.0,
                        "scale": 100
                },
                {
                        "type": "OrganizedAgent",
                        "distance": 150
                },
                {
                        "type": "ModPSO",
                        "numMeasurements" : 30,
                        "threshold": 5.2565,
                        "moveMin": 25,
                        "moveMax": 50
                },
                {
```

```
                        "type": "ChemoTaxisAgent",
                        "numMeasurements" : 30,
                        "stride" : 40,
                        "school" : false
                }
        ],
```

Lastly, the ”output” describes how output is produced. When wanting to recalculate some statistics, the ”output” variable is the only thing that needs to be changed. In other words, when running the simulator with the ”-s” option, this is the place to make changes.

The ”name” variable is the name appended on the output folder. To be certain that folders with similar names are not overwritten the simulator will prepend the date and time of creation. The ”confidence” variable is described in table 4.3. ”outputTypes” is also described in table 4.3 along with ”outputMethod”. To generate the statistics shown in table 4.5, use the code shown below. To generate the statistics shown in table 4.6, change the ”outputTypes” variable with ”LengthTraveled”.

To generate the data behind figure 4.5, have the ”outputTypes” as ”SimulationSteps” and change the ”outputMethod” variable to ”RelativeMean”. These are shown below.

```
        "output" :
        {
                "name":"Nonfixed_transmitter_large",
                "confidence":0.95,
                "outputTypes": [{"type":"SimulationSteps"}
                   ],
                "outputMethod":{"type":"RoundRobin"}
        }
}
```

To recreate the data shown in figure 4.6, retain all above, but change the following lines.

```
        "output" :
        {
                "name":"Nonfixed_transmitter_large",
                "confidence":0.95,
                "outputTypes": [{"type":"LengthTraveled"}]
                   ,
                "outputMethod":{"type":"RoundRobin"}
        }
```

```
}
```

To recreate the data shown in figure 4.5, change to the following.

```
        "output" :
        {
                "name":"Nonfixed_transmitter_large",
                "confidence":0.95,
                "outputTypes": [{"type":"SimulationSteps"}
                   ],
                "outputMethod":{"type":"RelativeMean"}
        }
}
```

Appendix D

Extended results

This appendix contains the full results of the edge analysis. The following results contains the data obtained after the discretization shown in figure 4.4.

	Call-out with Random	Lévy	Organized	Extended PSO	Chemotaxis
Random	**2** (0.0, 9, 0) **5** ($1.95e^{-3}$, 8, 0) **10** (0.0, 9, 0) **15** (0.0, 9, 0)	**2** (0.0, 9, 0) **5** (0.0, 9, 0) **10** (0.0, 9, 0) **15** (0.0, 9, 0)	**2** ($5.00e^{-1}$, 0, 4) **5** ($8.98e^{-2}$, 0, 6) **10** ($8.98e^{-2}$, 0, 6) **15** ($1.95e^{-3}$, 2, 6)	**2** (0.0, 9, 0) **5** (0.0, 9, 0) **10** (0.0, 9, 0) **15** (0.0, 9, 0)	**2** ($2.54e^{-1}$, 3, 2) **5** ($1.95e^{-2}$, 7, 0) **10** ($1.95e^{-3}$, 8, 0) **15** (0.0, 9, 0)
Call-out with Random	N/A	**2** (0.0, 9, 0) **5** (0.0, 9, 0) **10** (0.0, 9, 0) **15** (0.0, 9, 0)	**2** (0.0, 0, 9) **5** (0.0, 0, 9) **10** (0.0, 0, 9) **15** (0.0, 0, 9)	**2** (0.0, 9, 0) **5** (0.0, 9, 0) **10** (0.0, 9, 0) **15** (0.0, 9, 0)	**2** ($2.54e^{-1}$, 0, 5) **5** ($7.46e^{-1}$, 1, 2) **10** ($1.95e^{-3}$, 6, 2) **15** ($8.98e^{-2}$, 5, 1)
Lévy	See above	N/A	**2** (0.0, 0, 9) **5** (0.0, 0, 9) **10** (0.0, 0, 9) **15** (0.0, 0, 9)	**2** ($7.46e^{-1}$, 3, 0) **5** ($8.98e^{-2}$, 3, 3) **10** (0.0, 9, 0) **15** (0.0, 9, 0)	**2** (0.0, 0, 9) **5** (0.0, 0, 9) **10** (0.0, 0, 9) **15** (0.0, 0, 9)
Organized	See above	See above	N/A	**2** (0.0, 9, 0) **5** (0.0, 9, 0) **10** (0.0, 9, 0) **15** (0.0, 9, 0)	**2** ($8.98e^{-2}$, 4, 2) **5** (0.0, 9, 0) **10** (0.0, 9, 0) **15** (0.0, 9, 0)
Extended PSO	See above	See above	See above	N/A	**2** (0.0, 0, 9) **5** (0.0, 0, 9) **10** (0.0, 0, 9) **15** (0.0, 0, 9)

Table D.1: This table shows the results after discretization of the general results shown in table 4.5. This table contains the corner results, shown in red in figure 4.4. The meaning of the numbers in this table mirrors the meaning in the result table 4.5.

	Call-out with Random	Lévy	Organized	Extended PSO	Chemotaxis
Random	**2** (1.0, 10, 0) **5** (1.0, 8, 5) **10** (1.0, 17, 1) **15** ($2.23e^{-1}$, 42, 3)	**2** ($7.33e^{-6}$, 59, 2) **5** ($2.04e^{-5}$, 56, 4) **10** ($2.04e^{-5}$, 60, 0) **15** ($1.12e^{-12}$, 72, 0)	**2** ($3.93e^{-11}$, 22, 48) **5** ($7.33e^{-6}$, 13, 48) **10** ($7.33e^{-6}$, 8, 53) **15** ($1.33e^{-4}$, 22, 36)	**2** (0.0, 51, 28) **5** ($1.64e^{-13}$, 49, 24) **10** ($3.93e^{-11}$, 69, 1) **15** (0.0, 77, 0)	**2** ($2.93e^{-1}$, 5, 39) **5** ($1.75e^{-8}$, 30, 36) **10** ($3.93e^{-11}$, 43, 27) **15** ($3.14e^{-4}$, 46, 11)
Call-out with Random	N/A	**2** ($5.36e^{-5}$, 57, 2) **5** ($7.92e^{-7}$, 55, 8) **10** ($2.03e^{-10}$, 64, 5) **15** ($6.97e^{-12}$, 69, 2)	**2** ($6.66e^{-8}$, 12, 53) **5** ($7.33e^{-6}$, 9, 52) **10** ($9.69e^{-10}$, 3, 65) **15** ($1.33e^{-4}$, 6, 52)	**2** ($2.55e^{-15}$, 47, 28) **5** ($6.66e^{-8}$, 46, 19) **10** ($2.15e^{-14}$, 69, 5) **15** (0.0, 77, 1)	**2** ($1.07e^{-2}$, 1, 51) **5** ($5.36e^{-5}$, 25, 34) **10** ($3.93e^{-11}$, 41, 29) **15** ($9.69e^{-10}$, 45, 23)
Lévy	See above	N/A	**2** (0.0, 0, 77) **5** (0.0, 0, 80) **10** (0.0, 1, 80) **15** (0.0, 1, 78)	**2** ($2.55e^{-15}$, 41, 34) **5** ($1.75e^{-8}$, 32, 34) **10** ($1.49e^{-3}$, 54, 1) **15** (0.0, 77, 0)	**2** (0.0, 0, 81) **5** (0.0, 0, 80) **10** (0.0, 0, 79) **15** (0.0, 0, 79)
Organized	See above	See above	N/A	**2** (0.0, 74, 5) **5** (0.0, 79, 1) **10** (0.0, 80, 1) **15** (0.0, 80, 1)	**2** (0.0, 40, 39) **5** ($1.64e^{-13}$, 62, 11) **10** ($1.64e^{-13}$, 71, 2) **15** (0.0, 71, 8)
Extended PSO	See above	See above	See above	N/A	**2** ($2.03e^{-10}$, 0, 69) **5** (0.0, 0, 79) **10** (0.0, 0, 80) **15** (0.0, 0, 80)

Table D.2: This table shows the results after discretization of the general results shown in table 4.5. This table contains the edge results. The transmitter positions used in this table are labeled as blue in figure 4.4. The meaning of the numbers in this table mirrors the meaning in the result table 4.5.

	Call-out with Random	Lévy	Organized	Extended PSO	Chemotaxis
Random	**2** $(7.66e^{-1}, 0, 72)$ **5** $(2.80e^{-9}, 0, 112)$ **10** $(3.43e^{-5}, 0, 101)$ **15** $(1.46e^{-2}, 4, 86)$	**2** $(4.44e^{-15}, 120, 3)$ **5** (0.0, 123, 6) **10** (0.0, 140, 0) **15** (0.0, 153, 0)	**2** $(3.58e^{-11}, 25, 91)$ **5** $(6.79e^{-5}, 11, 89)$ **10** (0.0, 4, 131) **15** $(5.95e^{-1}, 13, 62)$	**2** (0.0, 119, 12) **5** (0.0, 111, 20) **10** (0.0, 151, 0) **15** (0.0, 154, 0)	**2** (0.0, 1, 133) **5** $(2.04e^{-8}, 15, 95)$ **10** $(5.24e^{-8}, 52, 57)$ **15** $(2.18e^{-2}, 75, 14)$
Call-out with Random	N/A	**2** (0.0, 134, 0) **5** (0.0, 145, 0) **10** (0.0, 154, 0) **15** (0.0, 154, 0)	**2** $(3.39e^{-10}, 58, 56)$ **5** (0.0, 69, 58) **10** $(9.85e^{-1}, 19, 44)$ **15** $(3.30e^{-12}, 89, 29)$	**2** (0.0, 131, 5) **5** (0.0, 142, 0) **10** (0.0, 154, 0) **15** (0.0, 154, 0)	**2** $(2.32e^{-3}, 0, 94)$ **5** $(7.43e^{-7}, 68, 38)$ **10** $(2.04e^{-8}, 102, 8)$ **15** $(1.80e^{-14}, 120, 2)$
Lévy	See above	N/A	**2** (0.0, 0, 148) **5** (0.0, 1, 147) **10** (0.0, 0, 151) **15** (0.0, 1, 150)	**2** $(9.58e^{-3}, 76, 15)$ **5** $(6.79e^{-5}, 40, 60)$ **10** (0.0, 148, 0) **15** (0.0, 154, 0)	**2** (0.0, 0, 154) **5** (0.0, 0, 154) **10** (0.0, 0, 154) **15** (0.0, 0, 154)
Organized	See above	See above	N/A	**2** (0.0, 151, 0) **5** (0.0, 143, 8) **10** (0.0, 153, 0) **15** (0.0, 154, 0)	**2** (0.0, 33, 106) **5** (0.0, 77, 53) **10** (0.0, 122, 12) **15** (0.0, 120, 21)
Extended PSO	See above	See above	See above	N/A	**2** (0.0, 0, 154) **5** (0.0, 0, 154) **10** (0.0, 0, 154) **15** (0.0, 0, 154)

Table D.3: This table shows the results after discretization of the general results shown in table 4.5. The transmitter positions used in this table are labeled as green in figure 4.4 and is contained in the middle of the search area. The meaning of the numbers in this table mirrors the meaning in the result table 4.5.

Appendix E

Running the simulator

This appendix contains the necessary information to both setup and run the simulator used in this thesis.

E.1 Setup

In order to run the simulator, created for this thesis, there are a few necessary setup steps.

The first thing that needs to be done is to install the latest version of the Haskell Platform[1]. The latest version can be found at `http://www.haskell.org/platform/`.

Once the Haskell Platform is installed, the next step is to download and install the necessary requirements for the simulator. Move to the directory where the simulator code is found, the directory will contain a file ending in a ".cabal". When this is done perform the following command[2]:

```
$ cabal install --only-dependencies
```

This action will likely take some time, the Haskell system is now downloading and compiling dependencies for the simulator.

The last step is to build the simulator. This is done with the command:

```
$ cabal build
```

[1] On most Linux distributions one can install GHC and Cabal through the package manager instead of installing the Haskell Platform

[2] Note that, in the commands shown in this appendix "$" represents the command-line.

this is done in the same directory as above. Note that if cabal complains during any step, make sure that the current directory is correct and that it contains the "SearchSimulation.cabal" file.

Once all these steps are done, the simulator can be run.

E.2 Running

The simulator is run with the executable created after the successful completion of the previous section. To run try:

```
$ ./dist/bin/SearchSimulation/SearchSimulation
```

The executable has three different options to run with.

-f The configuration file to use. Always needed. An example with an explanation is given in appendix C.

-g Run simulation in graphical mode. This is used in conjunction with the "-f" option to view the implemented strategies. If several options are present the simulator will ask which alternative should be used.

-s Raw result file to recalculate statistics. Since the simulator uses some time when running it is often handy to be able to recalculate some statistics without waiting for another run. By using the "-s" flag this can be accomplised quite easy.

Below are some examples of how to run and what will be accomplished.

```
$ ./dist/bin/SearchSimulation/SearchSimulation -f config.
    json
```

The above code, will make the simulator run with the parameters specified in the file "config.json". Because the simulator is capable of running on multiple processors this run can be further improved by adding:

```
$ ./dist/bin/SearchSimulation/SearchSimulation +RTS -N -
    RTS -f config.json
```

this will tell Haskell to use all available processors to run the simulation.

To run the simulator in graphics mode, just add the "-g" option, as shown below.

```
$ ./dist/bin/SearchSimulation/SearchSimulation -g -s
    config.json
```

To recalculate some statistics, just call the following

```
$ ./dist/bin/SearchSimulation/SearchSimulation -f
config_with_updated_output.json -s /path/to/old_raw_file.
    txt
```

This will create a new output folder, but will not run the full simulator.

Bibliography

[1] Lara Brinon Arranz. *Cooperative control design for a fleet of AUVs under communication constraints*. PhD thesis, INIRA, 2011.

[2] Ralf Bachmayer and Naomi Ehrich Leonard. Vehicle networks for gradient descent in a sampled environment. In *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, volume 1, pages 112–117. IEEE, 2002.

[3] Howard C Berg, Douglas A Brown, et al. Chemotaxis in escherichia coli analysed by three-dimensional tracking. *Nature*, 239(5374):500–504, 1972.

[4] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm intelligence: from natural to artificial systems*, volume 4. Oxford university press New York, 1999.

[5] Johann Borenstein and Yorem Koren. Real-time obstacle avoidance for fast mobile robots. *Systems, Man and Cybernetics, IEEE Transactions on*, 19 (5):1179–1187, 1989.

[6] Dan Bratton and Tim Blackwell. A simplified recombinant pso. *Journal of Artificial Evolution and Applications*, 2008:14, 2008.

[7] Rodney Allen Brooks. A robust layered control system for a mobile robot. *Robotics and Automation, IEEE Journal of*, 2(1):14–23, 1986.

[8] Elizabeth A. Capaldi, Alan D. Smith, Juliet L. Osborne, Susan E. Fahrbach, Sarah M. Farris, Donald R. Reynolds, Ann S. Edwards, Andrew Martin, Gene E. Robinson, Guy M. Poppy, et al. Ontogeny of orientation flight in the honeybee revealed by harmonic radar. *Nature*, 403(6769):537–540, 2000.

[9] Maurice Clerc and James Kennedy. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *Evolutionary Computation, IEEE Transactions on*, 6(1):58–73, 2002.

[10] Carlos A. Coello Coello and Maximino Salazar Lechuga. Mopso: A proposal for multiple objective particle swarm optimization. In *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 2, pages 1051–1056. IEEE, 2002.

[11] Kurt Derr and Milos Manic. Multi-robot, multi-target particle swarm optimization search in noisy wireless environments. In *Human System Interactions, 2009. HSI'09. 2nd Conference on*, pages 81–86. IEEE, 2009.

[12] Sheetal Doctor, Ganesh K. Venayagamoorthy, and Venu G. Gudise. Optimal pso for collective robotic search applications. In *Evolutionary Computation, 2004. CEC2004. Congress on*, volume 2, pages 1390–1395. IEEE, 2004.

[13] Anna Dornhaus and Lars Chittka. Why do honey bees dance? *Behavioral Ecology and Sociobiology*, 55(4):395–401, 2004.

[14] Russell Eberhart and James Kennedy. A new optimizer using particle swarm theory. In *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*, pages 39–43. IEEE, 1995.

[15] Russell Eberhart and James Kennedy. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, pages 1942–1948. IEEE, 1995.

[16] Sondre Andreas Engebråten. Using a genetic algorithm to optimize a uav formation. Technical report, NTNU, 2013.

[17] Gabriele Ferri, Michael V Jakuba, Emanuele Caselli, Virgilio Mattoli, Barbara Mazzolai, Dana R Yoerger, and Paolo Dario. Localizing multiple gas/odor sources in an indoor environment using bayesian occupancy grid mapping. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 566–571. IEEE, 2007.

[18] Dario Floreano and Claudio Mattiussi. *Bio-inspired artificial intelligence: theories, methods, and technologies.* MIT press, 2008.

[19] Douglas W. Gage. Randomized search strategies with imperfect sensors. In *Optical Tools for Manufacturing and Advanced Automation*, pages 270–279. International Society for Optics and Photonics, 1994.

[20] Adam T. Hayes, Alcherio Martinoli, and Rodney M. Goodman. Distributed odor source localization. *Sensors Journal, IEEE*, 2(3):260–271, 2002.

[21] Andrew Howard, Maja J Matarić, and Gaurav S Sukhatme. Mobile sensor network deployment using potential fields: A distributed, scalable solution

to the area coverage problem. In *Distributed autonomous robotic systems 5*, pages 299–308. Springer, 2002.

[22] Hiroshi Ishida, Yukihiko Kagawa, Takamichi Nakamoto, and Toyosaka Morizumi. Odor-source localization in the clean room by an autonomous mobile sensing system. *Sensors and Actuators B: Chemical*, 33(1):115–121, 1996.

[23] B. R. Jackson, S. Wang, and R. Inkol. Emitter geolocation estimation using power difference of arrival. Technical report, Defence R&D Canada, 2011.

[24] Wisnu Jatmiko, Yusuke Ikemoto, Takayuki Matsuno, Toshio Fukuda, and K. Sekiyama. Distributed odor source localization in dynamic environment. In *Sensors, 2005 IEEE*, pages 254–257. IEEE, 2005.

[25] James Kennedy. Bare bones particle swarms. In *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*, pages 80–87. IEEE, 2003.

[26] James Kennedy. Particle swarm optimization. In *Encyclopedia of Machine Learning*, pages 760–766. Springer, 2010.

[27] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The international journal of robotics research*, 5(1):90–98, 1986.

[28] Scott Kirkpatrick, D. Gelatt Jr., and Mario P. Vecchi. Optimization by simmulated annealing. *science*, 220(4598):671–680, 1983.

[29] Yoram Koren and Johann Borenstein. Potential field methods and their inherent limitations for mobile robot navigation. In *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pages 1398–1404. IEEE, 1991.

[30] G. Kowadlo and R. a. Russell. Robot odor localization: a taxonomy and survey. *The International Journal of Robotics Research*, 27(8):869–894, August 2008. ISSN 0278-3649. doi: 10.1177/0278364908095118. URL `http://ijr.sagepub.com/cgi/doi/10.1177/0278364908095118`.

[31] Renato A. Krohling. Gaussian swarm: a novel particle swarm optimization algorithm. In *Cybernetics and Intelligent Systems, 2004 IEEE Conference on*, volume 1, pages 372–376. IEEE, 2004.

[32] Averill M. Law. *Simulation Modeling & Analysis*. McGraw-Hill, 4th edition, 2007.

[33] Chang-Yong Lee and Xin Yao. Evolutionary programming using mutations based on the lévy probability distribution. *Evolutionary Computation, IEEE Transactions on*, 8(1):1–13, 2004.

[34] Min Cheol Lee and Min Gyu Park. Artificial potential field based path planning for mobile robots using a virtual obstacle concept. In *Advanced Intelligent Mechatronics, 2003. AIM 2003. Proceedings. 2003 IEEE/ASME International Conference on*, volume 2, pages 735–740. IEEE, 2003.

[35] Sukhan Lee and Jun Park. Cellular robotic collision-free path planning. In *Advanced Robotics, 1991.'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on*, pages 539–544. IEEE, 1991.

[36] Nadav Levanon. *Radar Prociples.* Johan Wiley & Sons, Inc., 1988.

[37] Robert M. Macnab and D.E. Koshland. The gradient-sensing mechanism in bacterial chemotaxis. *Proceedings of the National Academy of Sciences*, 69 (9):2509–2512, 1972.

[38] Marco Mamei, Ronaldo Menezes, Robert Tolksdorf, and Franco Zambonelli. Case studies for self-organization in computer science. *Journal of Systems Architecture*, 52(8):443–460, 2006.

[39] Benoit B. Mandelbrot. The fractal geometry of nature. *San Francisco, CA*, 1982.

[40] Hans P. Moravec. Sensor fusion in certainty grids for mobile robots. *AI magazine*, 9(2):61, 1988.

[41] Petter Ogren, Edward Fiorelli, and Naomi Ehrich Leonard. Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment. *Automatic Control, IEEE Transactions on*, 49(8):1292–1302, 2004.

[42] L Yu Paul, Jeffrey N Twigg, and Brian M Sadler. Radio signal strength tracking and control for robotic networks. In *SPIE Defense, Security, and Sensing*, pages 803116–803116. International Society for Optics and Photonics, 2011.

[43] Karl Pearson. The problem of the random walk. *Nature*, (72), 1905.

[44] David M. Pozar. *Microwave and RF design of wireless systems.* John Wiley & Sons, Inc., 2000.

[45] A.M. Reynolds. Cooperative random lévy flight searches and the flight patterns of honeybees. *Physics letters A*, 354(5):384–388, 2006.

[46] Toby J. Richer and T.M. Blackwell. The lévy particle swarm. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 808–815. IEEE, 2006.

[47] R. Andrew Russell, David Thiel, Reimundo Deveza, and Alan Mackay-Sim. A robotic system to locate hazardous chemical leaks. In *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, volume 1, pages 556–561. IEEE, 1995.

[48] R Andrew Russell, Alireza Bab-Hadiashar, Rod L Shepherd, and Gordon G Wallace. A comparison of reactive robot chemotaxis algorithms. *Robotics and Autonomous Systems*, 45(2):83–97, 2003.

[49] Simon R. Saunders and Alejandro Aragón-Zavala. *Antennas and Propagation for Wireless Communication Systems.* Johan Wiley & Sons, Inc., 2nd edition, 2007.

[50] Paul Scerri, Robin Glinton, Sean Owens, David Scerri, and Katia Sycara. Geolocation of rf emitters by many uavs. In *AIAA Infotech@ Aerospace 2007 Conference and Exhibit*, 2007.

[51] Dezhen Song, Chang-Young Kim, and Jingang Yi. Monte carlo simultaneous localization of multiple unknown transient radio sources using a mobile robot with a directional antenna. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 3154–3159. IEEE, 2009.

[52] William M. Spears and Diana F. Gordon. Using artificial physics to control agents. In *Information Intelligence and Systems, 1999. Proceedings. 1999 International Conference on*, pages 281–288. IEEE, 1999.

[53] William M. Spears, Diana F. Spears, Jerry C. Hamann, and Rodney Heil. Distributed, physics-based control of swarms of vehicles. *Autonomous Robots*, 17(2-3):137–162, 2004.

[54] William M. Spears, Diana F. Spears, Rodney Heil, Wesley Kerr, and Suranga Hettiarachchi. *An overview of physicomimetics.* Springer, 2005.

[55] Yi Sun, Jizhong Xiao, Xiaohai Li, and Flavio Cabrera-Mora. Adaptive source localization by a mobile robot using signal power gradient in sensor networks. In *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*, pages 1–5. IEEE, 2008.

[56] Donny K. Sutantyo, Serge Kernbach, Paul Levi, and Valentin A. Nepomnyashchikh. Multi-robot searching algorithm using lévy flight and artificial potential field. In *Safety Security and Rescue Robotics (SSRR), 2010 IEEE International Workshop on*, pages 1–6. IEEE, 2010.

[57] Thomas Thoresen, Jonas Moen, Sondre A. Engebråten, Lars B. Kristiansen, Jørgen H. Nordmoen, Håkon K. Olafsen, Håvard Gullbekk, Øistein T. Hoelsæter, and Lorns H. Bakstad. FFI-rapport 14/00958 - Distribuerte COTS UAS for PDOA WiFi geolokalisering med Android smarttelefoner. Technical report, Forsvarets forskningsinstitutt, 2014.

[58] Jeffrey N Twigg, Jonathan R Fink, PL Yu, and Brian M Sadler. Rss gradient-assisted frontier exploration and radio source localization. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 889–895. IEEE, 2012.

[59] G.M. Viswanathan and V. Afanasyevt. Lévy flight search patterns of wandering albatrosses. *Nature*, 381:30, 1996.

[60] G.M. Viswanathan, Sergey V. Buldyrev, Shlomo Havlin, M.G.E. Da Luz, E.P. Raposo, and H. Eugene Stanley. Optimizing the success of random searches. *Nature*, 401(6756):911–914, 1999.

[61] G.M. Viswanathan, Frederic Bartumeus, Sergey V. Buldyrev, Jordi Catalan, U.L. Fulco, Shlomo Havlin, M.G.E. Da Luz, M.L. Lyra, E.P. Raposo, and H. Eugene Stanley. Lévy flight random searches in biological phenomena. *Physica A: Statistical Mechanics and Its Applications*, 314(1):208–213, 2002.

[62] Karl Von Frisch. The dance language and orientation of bees. 1967.

[63] Aseem Wadhwa, Upamanyu Madhow, Joao Hespanha, and Brian M Sadler. Following an rf trail to its source. In *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*, pages 580–587. IEEE, 2011.

[64] Ronald E. Walpole, Raymond H. Myers, Sharon L. Myers, and Keying Ye. *Probability & Statistics for Engineers & Scientists*. Prentice-Hall, Inc., 7th edition, 2002.

[65] Charles W. Warren. Global path planning using artificial potential fields. In *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*, pages 316–321. IEEE, 1989.

[66] Charles W. Warren. Multiple robot path coordination using artificial potential fields. In *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pages 500–505. IEEE, 1990.

[67] Michael Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009.

[68] Wencen Wu. Bio-inspired cooperative exploration of noisy scalar fields. 2013.

[69] PL Yu and Brain M Sadler. Received signal strength gradient estimation for mobile networks. In *MILITARY COMMUNICATIONS CONFERENCE, 2010-MILCOM 2010*, pages 519–523. IEEE, 2010.

[70] Dimitri Zarzhitsky, Diana F. Spears, and William M. Spears. Swarms for chemical plume tracing. In *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE*, pages 249–256. IEEE, 2005.

Acronyms

ACO Ant Colony Optimization. 2

AI Artificial Intelligence. 2, 21

AOA Angle of Arrival. 17

APF Artificial Potential Field. 5, 28–30, 32, 34

CDF Cumulative Distribution Function. 16

EA Evolutionary Algorithm. 85

EM Electromagnetic. 3, 7, 8, 33

FDOA Frequency Difference of Arrival. 17

FFI Norwegian Defence Research Institute. 12

GA Genetic Algorithm. 27, 32

NLS Non-Linear Least Square. 18

PDOA Power Difference of Arrival. 1, 17, 18, 80, 86

PSO Particle Swarm Optimization. 3, 5, 6, 26–28, 34, 46–48, 59, 62, 63, 69–74, 77, 78, 98–100

RF Radio Frequency. 1, 3–5, 7, 8, 13, 19, 20, 26, 37–39, 72, 79–81, 84, 85

RSS Received Signal Strength. 12, 17, 18, 33–35, 42, 52, 59

SA Simulated Annealing. 49

SI Swarm Intelligence. 1–4, 20, 22, 23, 25, 26, 37, 46, 75, 78–80, 84

SNR Signal-to-Noise-Ratio. 5, 13, 15

SO Self-Organization. 23

TDOA Time Difference of Arrival. 17, 86

UAV Unmanned Aerial Vehicle. 1, 2, 4–7, 12, 18–20, 34, 35, 38, 52, 55, 79, 80, 82, 85