Delft University of Technology
Master's Thesis in Embedded Systems

Location-Free Gradient Navigation in Wireless Sensor Networks

Ioannis Protonotarios

embedded software

TU Delft Delft University of Technology

Location-Free Gradient Navigation in Wireless Sensor Networks

Master's Thesis in Embedded Systems

Embedded Software Group
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands

Ioannis Protonotarios
i.protonotarios@tudelft.nl

March 2014

Author
Ioannis Protonotarios (i.protonotarios@tudelft.nl)
Title
Location-Free Gradient Navigation in Wireless Sensor Networks
MSc presentation
March 19th, 2014

Graduation Committee

Prof. dr. K. G. Langedoen (Chair)	Delft University of Technology
Prof. dr. G. J. T. Leus	Delft University of Technology
Freek van Polen, MSc	Sense Observation Systems
Andreas Loukas, MSc	Delft University of Technology

Abstract

We consider the problem of navigating a mobile node to the extremum of a scalar field that is monitored by a wireless sensor network. Mobile nodes can perform tasks as repairing the network or providing coverage of an area where static nodes are not deployed. In particular, we are motivated by the scenario of a festival, where the task is to monitor and manage crowds of people. In this scenario, every participant is equipped with a sensor node in the form of a wrist band. The devices form a wireless network, which allows to monitor the density of the crowd. In cases of panic for example, we want to navigate each attendee away from areas of extreme crowd densities where dangerous phenomena may take place. Due to the high mobility, and low cost and power constraints of the wristbands, no location information can be used. We address this requirement by using gradient navigation methods.

In this thesis, we first detail the problem of location-free gradient navigation and present solutions from various research fields. Most approaches rely on taking unnatural paths for estimating the gradient. We demonstrate that even in simple scenarios estimating the angle of the gradient does not work. We then proceed to present a robust algorithm that always reaches the goal, although at the expense of a 50% increase in distance traveled compared to the euclidean distance. Finally, we present the results of an actual implementation of our solution on a testbed of 100+ nodes. The navigation algorithm is running on a smartphone, which communicates with the attached mobile wireless sensor node via bluetooth, and directs its user to the goal using the gradient broadcasted by the testbed nodes.

Preface

This thesis presents the work I've done in the Embedded Software group and Sense Observation Systems towards obtaining my master's degree. Both were involved in EWiDS, a project aiming at monitoring and managing crowds with wireless sensor networks. This collaboration was the incubator of the idea for this research. This thesis is not only important because of its practical aspect, but also because in the end, a working system was implemented.

I would like to thank a number of people for their help. First of all Andreas, who has been a great teacher and friend. Koen, Marco, Niels, and everyone else in the ES group for making me feel at home. Freek from Sense O.S. for hosting me, and providing everything I needed. My family for their never ending support.

Ioannis Protonotarios

Delft, The Netherlands
March 2014

Contents

Chapter 1

Introduction

Wireless sensor networks (WSNs) comprise of miniature computers with limited processing power and one or more sensors, communicating with each other wirelessly. These computers, often called *nodes* or *motes*, are inexpensive, draw little power, and usually have little or no infrastructure. Their use ranges form environmental monitoring [17] to smart buildings [20], and from battlefield surveillance [5] to disaster relief applications [6].

An important characteristic of WSNs is their ability to form ad-hoc networks, if they are randomly deployed. Different tasks are then delegated to various layers. The media access control (MAC) layer is responsible for when a node will use the wireless channel, and in turn when it is going to sleep for saving power. The nodes may be communicating in synchronous or asynchronous mode, while the routing layer takes care of forwarding messages to other nodes through the network. In cases of extreme node densities, gossip protocols are used. When gossiping, the information is passed on like a virus during an epidemic, with no need of a routing table.

WSN nodes have a lot of constraints. Their processing power is usually limited, the range of their radios is short, and the storage capacity is restricted. The most important constraint however, is that they run on a specific amount of energy, usually provided by a battery. In order to maximize their lifetime, they operate their components at a low duty cycle.

Nodes can be either static or mobile. Mobile nodes introduce new obstacles but also provide many research questions, like localizing and navigating mobile nodes within a wireless sensor network. Mobile nodes are actually static nodes with the added ability to reposition and organize themselves in the network.

In this thesis, we consider the problem of navigating a mobile node to the extremum of a scalar field that is monitored by a wireless sensor network. This scalar field may represent a sensed physical quantity such as temperature, or a network characteristic such as node density. Figure 1.1 shows an example of a navigation path in such a field.

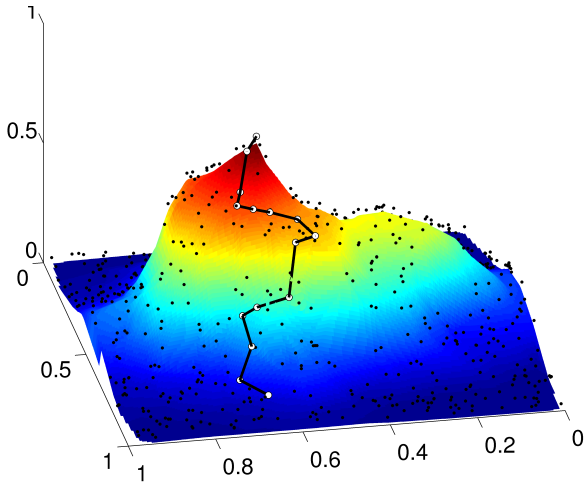


Figure 1.1: An example route to the peak of a scalar field. The vertical axis represents the value at each point of the field while the horizontal axes the coordinates of the nodes.

To better explain the motivation behind this work we provide three examples:

Environmental protection. Suppose that we want to detect the source of a dangerous chemical spill that has just occurred. We deploy a wireless sensor network in a uniform way, but without acquiring the location of each node. This network provides the field measurements for a mobile node that can autonomously navigate to the source and take appropriate action.

Crowd management. Another practical scenario is that of monitoring and managing crowds of people at concerts or festivals. In this scenario, every participant is equipped with a sensor node in the form of a wrist band. The devices form a wireless network, which allows to monitor the density of the crowd. In cases of panic for example, we want to navigate each attendee away from areas of extreme crowd densities where dangerous phenomena may take place.

Indoor navigation. Localization and navigation indoors, is a topic of high interest with many use cases. Imagine being navigated towards the items on sale in a store. Estimote[1] and iBeacon are two examples that localize you by roughly estimating your distance to the goal. These do not offer navigation and especially for larger spaces than a store they do not suffice. By deploying a wireless sensor network indoors, navigation can also be achieved in museums, offices, and malls.

[1] http://estimote.com/

The common characteristic of these scenarios is the absence of location information. In the first case we assume that we do not equip nodes with localization hardware for cost reasons, and in the other two that the nodes are either mobile or indoors, making location information very difficult to acquire. Therefore, an algorithm for navigating scalar fields must be location free. This in fact creates a major distinction between the topic of this thesis and the classical navigation/path planning problem.

1.1 Problem Statement

The core research question of this thesis is:

> *How to navigate to the extremum of a scalar field monitored by a wireless sensor network when no location information is available?*

Although this problem is general, its solution differs depending on the application requirements. For example, the gravity of choosing a safe navigation path is based on whether the mobile entity is a robot or a person. In this thesis, we focus on human entities because the research is supported by the EWiDS project that aims at better understanding the use of wireless sensor networks to monitor and manage crowds of people. This focus calls for some extra requirements:

The path taken must be "natural". Since we want to navigate humans, not robots or packets in a network, the steps of the algorithm have to result in a "natural" path. A system that would make a user go in circles, as many robotics algorithms do, would not be used by anyone.

Energy and cost constraints. The nodes provided to the attendees have to be *inexpensive* and run on *batteries* for the whole duration of the festival, which could span over more than three days. Using gps sensors for acquiring location would in turn drain the battery and increase the cost of the sensor node. High mobility is a factor that also plays a role in location estimation. That is why we choose not to use any location information for the navigation.

Nonlinear Fields. We want to be able to reach our goal in any given scalar field, independently of the characteristics of its gradient. This is in contrast to related work, which has considered only simple scalar fields with linear and concentric gradients, and fails when these constraints are lifted. In Chapter 4 we define the characteristics of the field's gradient and provide examples for each different type of field.

To summarize, the problem statement is:

> *How to navigate to the extremum of a scalar field monitored by a wireless sensor network when no location information is available, such that the steps leading to the goal form a natural path and success is independent of the field's gradient characteristics.*

This thesis addresses this problem in three steps:

- In Chapter 2, we detail the problem of location-free gradient navigation and present solutions from various research fields. Then, we identify two possible candidates that match our requirements.
- In Chapter 3, we examine algorithms from two categories that satisfy the requirements about the path's naturality and the gradient's characteristics. In Chapter 4, we show that algorithms based on estimating the direction of the gradient demonstrate poor performance on nonlinear fields. We therefore present a simple alternative algorithm that always reaches the goal at the expense of a 50% increase in distance traveled compared to the euclidean.
- Finally, we present the results of an actual implementation of our solution on a testbed of 100+ nodes in Chapter 5. The navigation algorithm is running on a smartphone, which communicates with the attached mobile wireless sensor node via bluetooth, and successfully directs its user to the goal.

Chapter 2

Problem Setup and Related Work

We begin by presenting the general navigation problem (Section 2.1), and then we focus on gradient navigation and the corresponding related work (Section 2.2).

2.1 Problem Setup

Let $G = (V, E)$ be a network of n nodes over a two dimensional space. Function $P : V \rightarrow \mathbb{R}^2$ maps each node to a point $x = (x_1, x_2)$, in $\mathbb{R}^2$. A mobile node m starts from $x^0 = (x_1^0, x_2^0)$ and at each step k, moves to a new position according to some algorithm $D(\{x^k\}) = x^{k+1}$. k is the discrete time of each position in the trajectory $\{x^k\} = \{x^0, x^1, .., x^k\}$ that the mobile node follows. This trajectory is comprised of discrete steps such that $||x^{k+1} - x^k|| = a$, where a is the step size.

Classic Navigation. The objective of classic navigation is to find the algorithm D that will navigate the mobile node m to some point $x^* = (x_1^*, x_2^*)$. One way of expressing this is

$$\min_D \sum_{k=0}^{+\infty} ||D(\{x^k\}) - x^*||. \tag{2.1}$$

In robotics, this usually translates to path planning. The feasibility of path planning is dependent on the ability to use or construct a map and localize itself on it. Therefore, sensor localization is necessary when we want to use classic path planning techniques such as automotive navigation systems. Localization is achieved by acquiring distance estimates from neighbors and then with either triangulation or trilateration calculating the position in space. This distance measurement can be obtained by using

various methods: time of arrival (ToA), time difference of arrival (TDoA), received signal strength (RSS), or frequency shift (Doppler effect). These are all known and well studied methods that we cannot use due to special hardware requirements (ToA, TDoA) or low accuracy (RSS). The most-known system that uses TDoA is GPS [13], however power and cost constraints prohibit its use.

Gradient Navigation. Faced with the inability to acquire $P(m)$, we turn to gradient navigation. Its objective is defined with respect to some function $f : \mathbb{R}^2 \to \mathbb{R}$ that describes a scalar field over the two dimensional space. The scalar field described by f can be a physical process monitored by the network or one of its properties, such as node density or traffic load. The value of the scalar field at some node u is $f(P(u))$, for which we write in shorthand $f(u)$. We have to note that since the field is sampled at discrete points in space, the mobile node will estimate its field value by averaging the values of its neighbors

$$f(m) = \frac{\sum_{u \sim m} f(u)}{|u \sim m|}, \tag{2.2}$$

where $u \sim m$ are the nodes in the communication vicinity of the mobile node m. This problem is often referred to as source seeking/localizing or gradient climbing/descent. The objective of the navigation function D is

$$\min_{D} ||\nabla f(D(\{x^k\}))||, \tag{2.3}$$

where $\nabla f(x)$ is the gradient of f near x.

Note that in contrast to classic navigation, the goal is not a point in space but rather a node u that has the highest value across the scalar field. We terminate the algorithm D at time k, when the mobile node m enters the communication vicinity of the goal node u. The requirements can also be formulated in the following way:

- Node m is not aware of $P(u \sim m)$ or $\nabla f(m)$, but only of $f(m)$.
- $\{x^k\}$ must be perceived as "natural" by a human being.
- D is independent of the characteristics of the scalar field ($f''(m)$).

2.2 Related Work

In the following section, we present the related work on gradient navigation. Although a large body of work exists, mainly from the area of robotics, most of it fails to meet the requirements of path naturality and field linearity. Additionally, little emphasis has been given in implementing the methods that are proposed. However, we identify two possible candidates that serve as a basis for the solutions that we will present in Chapter 3.

2.2.1 On-line Gradient Estimation

If the gradient, $\nabla f(m)$ can be estimated, gradient descent can be used. Gradient descent methods require only local information, but can get stuck at local optima [10, 11, 26]. Finding the local extremum of f is only a matter of taking steps proportional to the negative of $\nabla f(m)$. Since our requirement is that we are unaware of the gradient, we look into the research that has been done on on-line gradient estimation. The common paradigm of these methods is to try to align the velocity vector of the mobile node with the field's gradient.

In order to successfully estimate the gradient by taking point measurements, these must be at a distance from each other. Wireless sensor networks can take advantage of the fact that nodes are at different locations to estimate the gradient of the field without changing their location [2, 4, 19]. However, the nodes must be aware of their neighbors' location. Such a requirement cannot be satisfied in our scenario.

Single node on-line estimation can be achieved by "dithering" the sensor position while maneuvering. Twigg et al. consider using both RSS average and gradient estimation to locate an RF source [23]. They propose to perturb the path of motion in order to achieve better RSS gradient estimates and compare different strategies to realize that. In GUIDE [12], a WiFi source is located using its RSSI and signal quality. In contrast to the previous scenario, the user is navigated by turning by constant angles based on the gradient it is estimating at every step. The chosen angles result in a path formed of triangles, which is not natural.

Sun et al. propose a *power gradient guide* that takes into account the last N measurements of the signal in order to estimate the correct direction of the signal power gradient. When N is small, the mobile node will turn faster but will be more susceptible to noise and when it is large it will be more reliable but change direction slowly [22]. In order to estimate the initial direction and whenever the gradient's estimation fails, a circle-like motion is employed, which is not natural.

The work in [21, 29] is inspired by geometric laws. The law of cosines is used to estimate the direction of the gradient. A model is used to estimate the distance between the mobile node and a radio source, based on the RSSI of the source's signal. This estimated distance is used to calculate the angle towards the radio source. This method actually estimates the direction of the local gradient and then navigates the mobile node along it. The first algorithm that we take into consideration is from this category. A more detailed explanation is presented in Section 3.1.

2.2.2 Biased Random Walk

The motion of bacteria in search for food, chemotaxis, has also inspired a lot of researchers for solving the source-seeking problem. These methods are very robust, but exhibit slow convergence and unnatural paths. In principle, the goal is to assign a probability density for the nodes' position whose maximum coincides with the maximum of the field. This is achieved by random biased walk, where nodes switch between two distinct motions, *tumble* and *run* [3].

In the *run phase*, the bacteria swims in the same direction at constant velocity while in the *tumble phase* it rotates in place, in a way that its next direction will be random. The duration of each run is not equal. It is dependent on the concentration of the chemical substance. For example, the chemical is food and the bacteria is attracted towards it. When the concentration increases, the probability of a tumble phase is lower. In this way, the lowest probability of a tumble phase will coincide with the maximum for the sensed quantity.

Mesquita et al. propose a technique where the probability density of the mobile node's position is controlled in a way that not only the extremum of the signal field is found, but its whole spacial profile [18]. In their work, *Optimotaxis*, convergence is proven and through simulations, the successful approach of both local and global extrema. Other positive points of this work are the simplicity of the method and its low cost implementation as well as its robustness to local extrema. However, the fact that the algorithm results in a random, unnatural path and its slow convergence deem it unfit for our scenario.

Wadhwa et al. propose an *RSS-adapted optimotaxis* to locate an RF source [24]. To overcome the high spatial variation of RSS, they propose various patterns of movement to better estimate the direction of the source. Although convergence is significantly faster, these unnatural patterns once again disqualify the use of this algorithm.

A memoryless robot, only taking point measurements has been proposed by Y. Elor et al. to navigate to the extremum of a scalar field [9]. The limitation of this work is that two robots that can sense their relative distance are needed. Other related work in this category can be found in [1, 7, 15, 28] although the common characteristic is the unnatural path not suitable for human navigation.

The second algorithm that we take under consideration is a hybrid of online gradient estimation and biased random walk (Section 3.2). The sign of the gradient is estimated by moving on a straight line for sake of naturality, and the direction is altered by turning by a fixed or random angle since the direction of the gradient is not being estimated.

2.2.3 Data Routing Based

Another important category is based on the extensive work done for data routing. The latter relies on navigating a mobile node m in the same way a data packet would be routed through the network. The mobile node follows a node-to-node path, until reaching the target node where f has an extremum. Only local information is used since the location of the extremum is not known. Node-to-node navigation is performed by estimating the distance to each target node using simple hop-count.

In [27], four implementations of ad-hoc routing protocols are presented in respect to driving mobile robots across the network. The focus of these is not exceeding a certain packet loss ratio while keeping a delay boundary. After a route has been found, the mobile node can be navigated node-by-node to the goal.

Deshpande et al. propose creating a pseudo gradient centered on the node closest to the target and then by using directional antennas navigate node-by-node to the goal [8]. This gradient is created by flooding the network in order for every node to establish a hop-count distance to the goal. The mobile node then navigates to the closest node with the highest "pseudo gradient" till it reaches the goal.

The approaches based on node-to-node navigation assume knowledge of P or the ability to estimate it, which is against one of our main requirements.

Chapter 3

Algorithms and Analysis

In the following chapter, we present three algorithms that meet the requirements of our problem. We begin with a geometry-inspired algorithm, which we call the Cosine Law algorithm. In Section 3.2 we present two algorithms that are based on a hybrid of biased random walk and on-line gradient estimation.

3.1 The Cosine Law Algorithm

The first algorithm that we take under consideration is inspired by the Law of Cosines. Variations of this algorithm have been proposed for navigating the gradient that an RF source creates [21, 29]. This algorithm is easily explained through the example in Figure 3.1. The field is depicted with the gray concentric disks. Their center, point N, is the RF source we want to navigate to. Imagine that the mobile node is moving along line AB and it is able to take measurements of this field at any given time. Without any other information, a map, a compass or a GPS module, it cannot recognize in which direction N lies.

The Law of Cosines relates the lengths of the sides of a triangle to the cosine of one of its angles. This means that if the lengths of the 3 sides of a triangle are known, we are capable of calculating its angles. In Figure 3.1, a triangle is formed between points A, B and N. The distance between A and B is known, because we know the number of steps that we took getting there. If we can somehow acquire the length of sides AN and BN, we will be able to calculate the angle θ that we have to turn in order to head straight to N. But we also do not know if we have to turn θ or θ', since these two locations are mirrored. This is referred to as the "flipping problem". That is why we need to first randomly select one of the two, and then make sure it was the correct one. If not, we step back and select its mirrored angle.

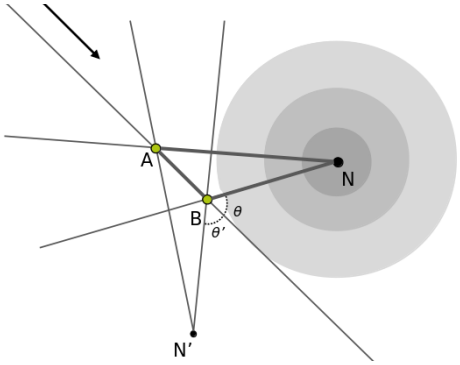


Figure 3.1: Cosine Law Algorithm angle calculation example. Imagine we are moving in the direction the arrow is pointing at. Notice the triangle being formed by points A,B and N. N is our goal in the center of the field depicted with the gray disks. We want to calculate angle θ that will lead us to N.

The angle θ is calculated with the following equation:

$$\theta = arccos(\frac{||x_A - x_N||^2 - ||x_B - x_N||^2 - ||x_A - x_B||^2}{2 \cdot ||x_A - x_B|| \cdot ||x_B - x_N||}) \tag{3.1}$$

These previous steps are summarized in the following pseudocode. They can be applied to any gradient, like the one created by a wireless network and its nodes in our case. In Figure 3.1, for example, A can be the position of the mobile node that wants to reach the node at N, which is outside of its communication range. The gradient that we use however is created over the whole network.

Algorithm 3.1 Cosine Law Algorithm pseudocode.

```
init
pick a random direction
measure signal field                          ▷ Equation 2.2

repeat
    step forward
    measure signal field
    calculate angle θ                         ▷ Equation 3.1
    add θ to direction of movement
    step forward
    measure signal field
    if values decrease then
        step back
        subtract 2θ from direction of movement
    end if
until goal reached
```

3.1.1 Distance Estimation

In order to calculate θ, we need to estimate the distance to the goal node. The method that we will choose has to use the field's value as the only input.

The euclidean distance $||P(u) - P(v)||$ of any two nodes u and v is unknown. Nevertheless, we can approximate it using the first-order Taylor's expansion

$$f(v) = f(u) + \nabla f(u)||P(u) - P(v)||. \tag{3.2}$$

The gradient in the vicinity of u is approximated by

$$\nabla f(u) = \frac{\max\limits_{z \sim u} f(z) - f(u)}{R}, \tag{3.3}$$

where $z \sim u$ denotes that z and u are in communication proximity. The distance between z and u is R, and r is the transmission radius of the nodes. Given a monotonic field, we assume that $\mathbb{E}[R] = r$ because the node with the maximum value of f is most probably at the edge of u's transmission vicinity. Observe that $\nabla f(u)$ depends on R and therefore is also a random variable. Solving for $||P(u) - P(v)||$ and taking the expectation for R, we have the estimator

$$||P(u) - P(v)|| \approx \frac{f(v) - f(u)}{\max\limits_{z \sim u} f(z) - f(u)} r. \tag{3.4}$$

However, the expectation of $\mathbb{E}[R] = r$ has to be validated. For this purpose we have run simulations with various values of R to prove that our assumption holds. Furthermore, we want to know what the error of this estimation is and how it is affected by the field's characteristics. In addition, we want to know how this error affects the estimation of θ. These simulations and a more detailed discussion will be presented in Section 4.3.

3.2 Biased Random Walk

As will be shown in Section 4.3, the Cosine Law algorithm performs poorly. This is mainly due to the distance estimation Equation 3.4. In an effort to find a more robust solution we turn to the category of biased random walk algorithms.

The typical biased random walk comprises of two distinct phases. The first, "run", is when the bacteria swims in a fixed direction and the second, "tumble", when it changes direction. The rate at which these two phases alternate is proportional to the concentration of the chemical substance the bacteria wants to move towards to or away form. For example, if the bacteria is moving towards the right direction it will tumble less frequently.

In our case, we create a hybrid algorithm based on random walk and gradient estimation. Rather than trying to estimate the direction of the gradient, we only care about its sign. As long as the sign is positive, which means that we are moving towards our target, we are in the "run" phase and will not change our direction. When we detect that the gradient is negative, we "tumble" by choosing a random direction.

This algorithm was inspired by the natural steps a human being would take if she was only aware of the field's value. Maintain the direction while the field values increase and try a new one when they decrease. The steps are summarized in the following pseudocode:

Algorithm 3.2 Biased Random Walk pseudocode.

```
init
pick a random direction
measure signal field
repeat
    step forward
    measure signal field
    while values increase do
        forward step                          ▷ run phase
        measure signal field
    end while
    pick a random direction                   ▷ tumble phase
until goal reached
```

This algorithm, although more robust than the Cosine Law algorithm, has poor performance due to the random direction that we choose every time the polled field values start decreasing. A benchmark in Section 4.2 will demonstrate this poor performance. Furthermore, its naturality aspect is low; for example, turning 180° would not be a choice a human would take. However we will use this version as the baseline for a comparison with the algorithm presented next.

3.2.1 The Constant Angle Algorithm

This section enhances the performance of the previous algorithm by answering the following question: What if we use a constant angle when we choose to make a turn rather than a random one? What should this angle be?

A scenario of the Constant Angle algorithm, can be seen in Figure 3.2. If the magnitude of the field is increasing while moving from point A to B, the agent will continue in the same direction. After passing point B the magnitude will start decreasing, so a new random direction will be chosen. We can infer that if instead of a random direction we choose to make a 90° turn we will be moving in a direction leading straight to the goal. This is because

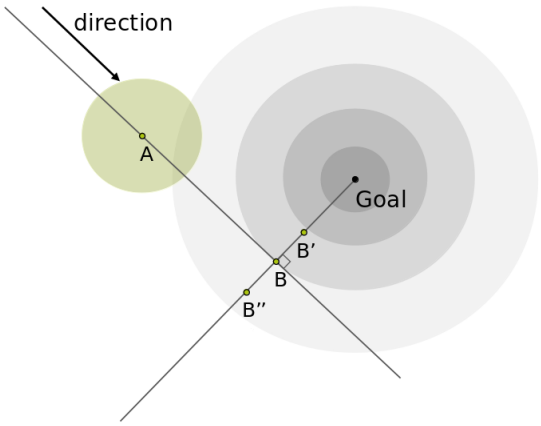


Figure 3.2: Constant angle algorithm example. While moving in the direction shown by the arrow, the field's value will increase till we reach point B. After that, we need to change our direction. Assuming concentric field iso-contours, any line perpendicular to those will intersect the goal. That is the reason why we choose to turn by 90°. Due to the flipping problem, the direction of the turn is random.

the magnitude will start decreasing at the point were the trajectory of the mobile node is perpendicular to the gradient's direction. If the magnitude is again decreasing, this means that we are moving towards B" instead of B', so by turning 180° we reach the goal.

The steps of the algorithm are summarized in the following pseudocode:

Algorithm 3.3 Constant Angle Algorithm pseudocode.

```
init
pick a random direction
measure signal field
repeat
    step forward
    measure signal field
    while values increase do
        forward step                          ▷ run phase
        measure signal field
    end while
    turn 90°
    step forward
    measure signal field                      ▷ tumble phase
    if values decrease then
        turn 180
    end if
until goal reached
```

We simulated the Constant Angle algorithm on fields of different concavities and non-concentric gradients. In Chapter 4, we will show that the Constant Angle algorithm outperforms the first two algorithms presented, both in convergence time and success rate.

Chapter 4

Algorithm Simulations

In this chapter, we present the validation and evaluation of the algorithms presented in Chapter 3 through a series of simulations. But first, we present the simulation tool and describe its setup in Section 4.1.

4.1 Simulation Environment

The simulation tool that was used is called Lapsang [1]. It is a tool built in JAVA by PhD students of the Embedded software group to enable fast prototyping of distributed algorithms for mobile ad-hoc networks. These algorithms usually scale up to thousands of nodes, but by using its very intuitive visualization (see Figure 4.1), we can quickly verify their correct implementation.

4.1.1 Properties and Setup

Motivated by the festival scenario, we have built our system and the algorithms that we will use. This is the same reason why we want our simulation environment to resemble the festival's network conditions as much as possible. In festivals, an approximate density is 3-4 persons per m^2. A low power device, such as the wristband that each person wears, has an average transmission radius of 1.5 - 2 meters. The average node degree for this combination of transmission radius and density is approximately 30.

To replicate these conditions we use a simulation area that represents an actual area of 100×100 meters, which we cover with 1000 nodes. Each node has a transmission radius of 10 meters. This combination results in the same average node degree of 30, as in the festival scenario.

[1] https://code.google.com/p/lapsang/

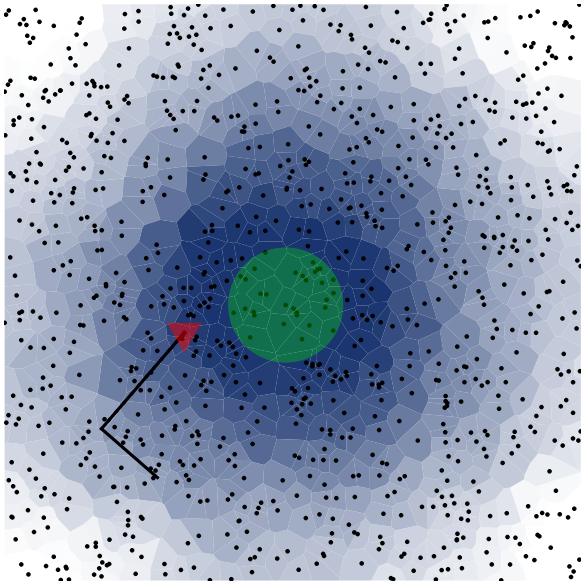


Figure 4.1: A frame of a sample simulation. The green disc represents the transmission vicinity of the goal node while the red triangle stands for the mobile agent. The black line is the path traveled so far. All other nodes are depicted using black dots and finally the signal field is visualized using blue colored Voronoi cells.

We choose to model the communication between nodes using a unit disk graph (UDG) and presume no packets are lost. These assumptions do not hold in practice [31, 32], but provide us with the necessary simplification to evaluate the given algorithms.

Our goal, as mentioned before, is to navigate a person to a predefined node. This goal is considered reached when the person's node is within the transmission radius of the target node. The performance metric that we use to evaluate the algorithms is the distance traveled until reaching the goal. And since the starting point of every simulation is random, the relative distance traveled is used.

$$\text{Relative Distance Traveled} = \frac{\text{Actual Distance Traveled}}{\text{Euclidean Distance to Goal}} \tag{4.1}$$

4.1.2 Gradient Field Creation

One of the main contributions of this thesis is that we simulate the algorithms on different kinds of signal fields. Therefore one of our objectives is to understand how the characteristics of a field's gradient are affecting the performance of our algorithms. It will be shown in Sections 4.3 and

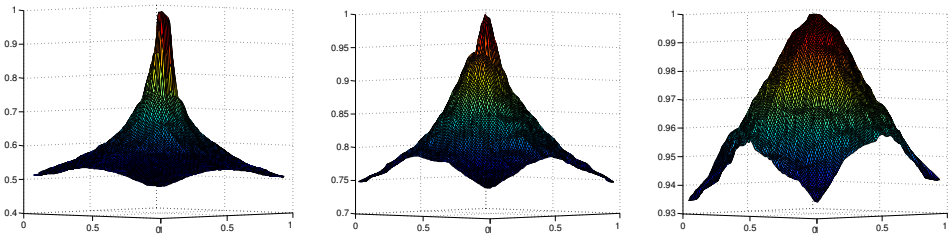
4.4, that the type of the field is of great importance. We consider two field properties; the concavity of the gradient and whether its iso-contours are concentric. We begin our analysis with the first property.

Field Concavity. The second derivative of a function defines its concavity. A function with a positive second derivative is called convex or concave up. If the second derivative is negative, the function is called concave down.

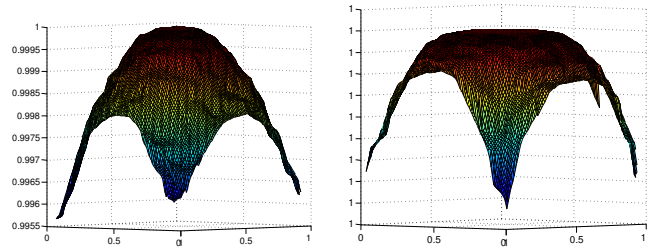
In order to create a concave field, we first create a simple hop count starting from the goal node, $hc : V \to \mathbb{R}$. This field is discretized in steps of transmission radius length. Therefore, we use a smoothing algorithm that basically assigns to each node the average value of its neighborhood. An example of a smoothing distributed mechanism like this can be seen in [16]. Apart from diffusing the information, the resulting field has one global extremum. In this way we obtain a linear field $l = smooth(hc) : V \to \mathbb{R}$. The final field we create is:

$$f = 1 - l^c, \tag{4.2}$$

where c is the concavity factor. The resulting fields for a concavity factor value of 0.25, 0.5, 1.0, 2.0 and 4.0 are presented in Figure 4.2. Before commenting on them, we have to indicate that all functions that have a maximum, are concave down in the area around that maximum. If we exclude this area, the first two fields are concave up, the third has zero concavity making it linear, and the last two are concave down.



(a) Concavity factor 0.25. (b) Concavity factor 0.5. (c) Concavity factor 1.



(d) Concavity factor 2.0. (e) Concavity factor 4.0.

Figure 4.2: Signal fields with varying concavity. The y axis is the field's value.

Concentricity. As stated in the beginning of this section there is another property of the signal field that we also take into consideration. All these fields so far had concentric iso-contours. This is easily seen in Figure 4.3 for the linear field.

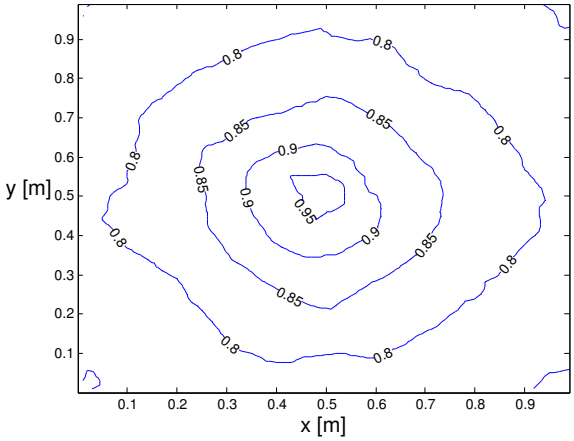


Figure 4.3: A field with concentric iso-contours.

The goal in gradient navigation is aligning the direction of movement with the direction of the field's gradient. All gradient estimation techniques estimate the direction of the local gradient. If this direction is pointing towards the goal in every point of the field, navigating to it is easier. In practice, we cannot rely on a field property such as this for our algorithm to work properly since naturally occurring fields usually have non-concentric gradients.

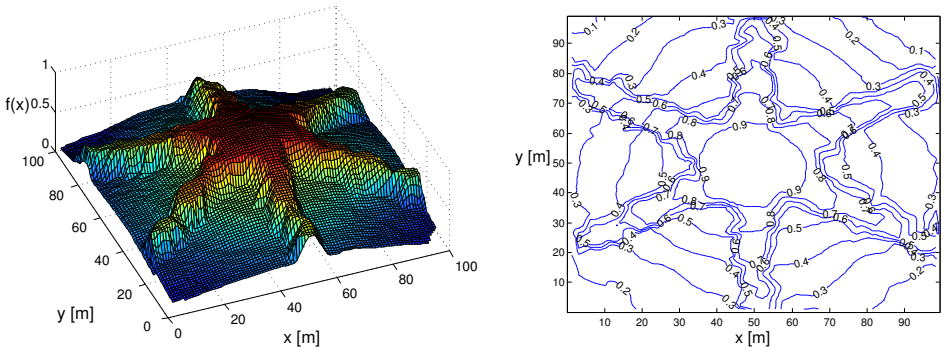
In order to test a more complex field where the direction of the gradient is not uniform, we use the superformula[2] (Equation 4.3) to create a mask for the linear field that will create steep ridges. The parameters used for the formula are shown in Table 4.1, and the field itself in Figure 4.4a.

$$r(\phi) = \left[\left| \frac{\cos\left(\frac{m\phi}{4}\right)}{a} \right|^{n_2} + \left| \frac{\sin\left(\frac{m\phi}{4}\right)}{b} \right|^{n_3} \right]^{-\frac{1}{n_1}} \tag{4.3}$$

a	b	m	n1	n2	n3
0.1	0.61	12	41.5	34	4

Table 4.1: Superformula parameters

[2]http://en.wikipedia.org/wiki/Superformula



(a) The field. (b) The iso-contours of the field.

Figure 4.4: Superformula mask applied on linear field. The ridges are evident in subfigure (b). An algorithm that follows the gradient would have to first "ascent" the ridge and then reach the center.

4.2 Algorithm Comparison

Before examining the algorithms in detail, we present a comparison between the performance of the three algorithms presented in Chapter 3. The objective is to provide a high-level intuition on how each of them performs.

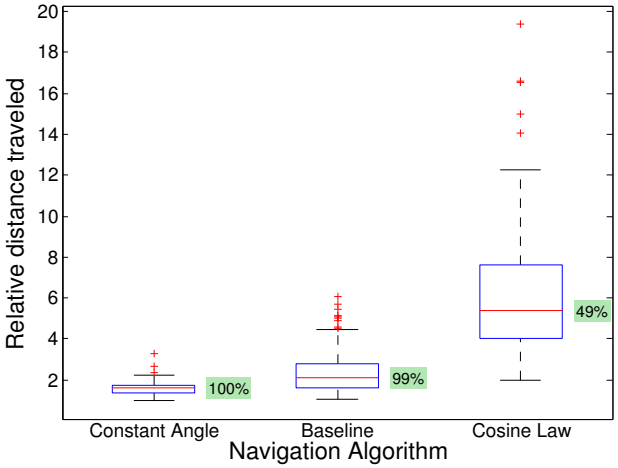


Figure 4.5: Success rate (in green) and relative path length for the three algorithms presented in Chapter 3. Using the linear field, ten runs of ten random networks were simulated for each algorithm.

The value highlighted in green indicates the success rate of each algorithm. As success rate we define the percentage of simulation runs where the mobile node reached its goal over the overall runs. An unsuccessful run is a run that resulted in navigating the mobile node out of bounds of the simulation space.

The Cosine Law algorithm has the worst performance, while demonstrating a very weak 49% success rate. In the following section we present and discuss the reasons behind this poor performance. On the other hand, turning by a constant angle of 90° reduced the relative path length of the Biased Random Walk algorithm by 25%. The last section of this chapter discusses the simulations that were run for these two algorithms. We show that they are robust, since field concavity or whether its iso-contours are concentric, do not affect their performance.

4.3 Cosine Law Algorithm Simulations

In the following, we present the verification of the distance estimation equation 3.4 as well as of the assumption that $\mathbb{E}[R] \approx r$. We show that even though this assumption is correct, the error of estimating the distance to the goal is too high.Motivated by this poor performance, we investigate whether an estimator exists for which the Cosine Law algorithm works. Our results show that *arrcos* is very sensitive to small distance errors. This suggests that it is unlikely that an estimator that would make the algorithm work exists.

Approximation of $\mathbb{E}[R]$. In order to verify that $\mathbb{E}[R] \approx r$, we run the following simulation: we measure the actual distance of each of the 1000 nodes to the distance of their neighbor with the maximum field value. We use the linear field and run the simulation ten times for six different transmission radii.

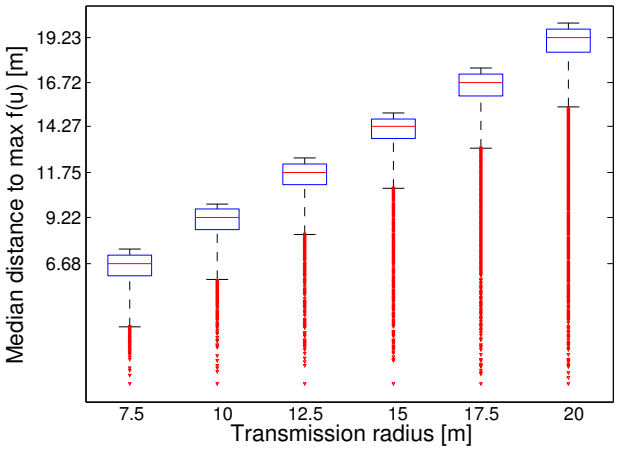


Figure 4.6: Median distance to neighbor with the maximum field value. Using the linear field, ten random networks were simulated.

We notice that by assuming that the node with the maximum value is at the edge of the transmission radius we are overestimating $\mathbb{E}[R]$. In order

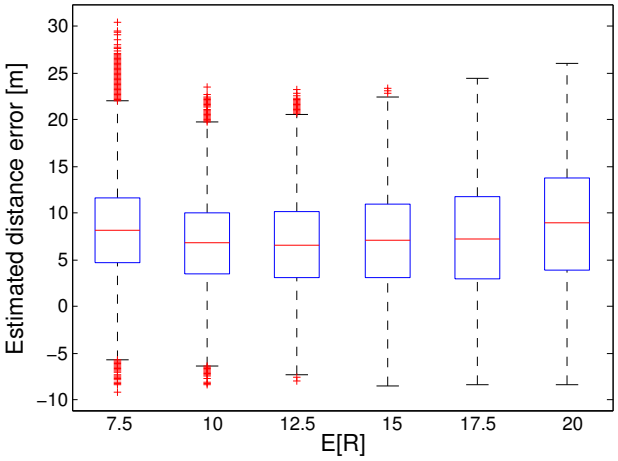


Figure 4.7: Estimated distance to goal node error. The error is minimal for $\mathbb{E}[R] = 12.5$, which is equal to the transmission error.

to understand how much this overestimation affects the estimation of the distance to the global extremum of the field we run the following simulation: we set the transmission radius at 12.5 meters and calculate the absolute error between the estimated and the euclidean distance from each of the 1000 nodes to the goal node. We use the linear field and approximate $\mathbb{E}[R]$ with the six values as before. For each value, we ran the simulation on 100 different networks.

Figure 4.7 shows that the minimum median of the error is 6.5 meters, when $\mathbb{E}[R] = r = 12.5$ meters. This error can be attributed to the fact that we use only the first order of the Taylor expansion and to the overestimation of $\mathbb{E}[R]$ as we saw in Figure 4.6.

Although in practice we cannot know the exact distance between the position of the mobile node and the node that has the maximum value in its neighborhood, we do know it during simulations. In order to reduce the error caused by the overestimation of $\mathbb{E}[R] = r$, we will use the real distance between the goal node and the mobile agent when using the distance estimation Equation 3.4 for R, for the rest of our simulations.

Field concavity. A key requirement for the solution we seek is its indifference to the concavity of the gradient of the signal. For the next series of simulations, we estimate the distance of every node to the goal by using the Taylor approximation once again. This time we use all the fields that were presented in Subsection 4.1.2. Figure 4.8 displays the error for each of the five fields, which is the absolute difference between the estimated distance from each node and the actual to the goal node. Notice that for the field with concavity factor 0.25, the median of the error is ≈ 100 meters. The maximum possible distance from the goal node is equal to the radius of the

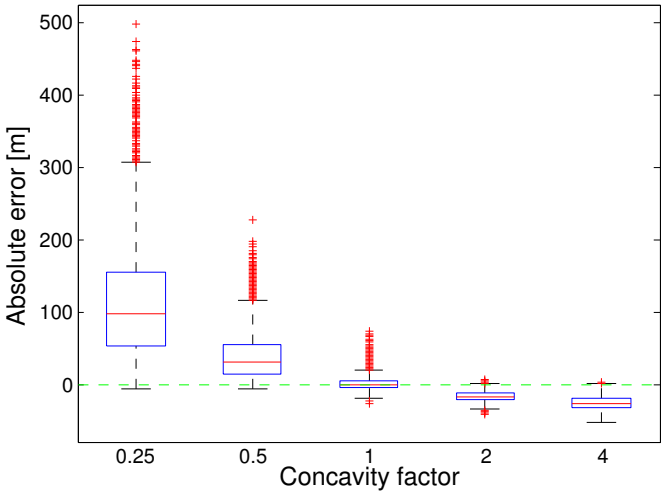


Figure 4.8: Distance estimation error for five fields of different concavity. We can see that for the nonlinear fields (Concavity factor $\neq 1$), the error increases significantly. For convex fields we overestimate the distance while we underestimate for concave fields.

circle that is circumscribed around the simulation space and has the goal node as its center. This radius is $\sqrt{2} * 100/2 \approx 70$ meters.

Angle estimation. Although the results of distance estimation are far from satisfactory, calculating the desired angle θ using Equation 3.1 could yield better results. That could be possible since the two erroneous distance estimations, used in calculating θ, have high spatial relevance. To further explain this, we need to understand how the Cosine Law algorithm actually works. The angle θ is an estimation of the direction of the field's local gradient. The two points from where we estimate the distance to the goal are more likely to have a similar local gradient and in turn a comparable distance estimation error.

To test this hypothesis, we ran the following simulation: we followed a random route that was comprised of steps of constant length taken at a random direction every time. In this way, we can use each set of consecutive points to estimate the angle using Equation 3.1. The distance between every two points is constant, so we only need to estimate the distance of the two points to the goal. As before, this simulation was run for the five fields of different concavities. The absolute error is the difference in degrees of the angle θ and the direction to the goal node.

Before commenting on the findings, we must note that due to the distance estimation error, the resulting arguments for the arccos function were greater than allowed. For this reason, they were clipped so that their absolute value was less than one. Also, an angle error of above 180° has no meaning since

we are randomly either adding or subtracting θ to our previous direction of movement, so these values were also clipped to be in the range of 0-180°.

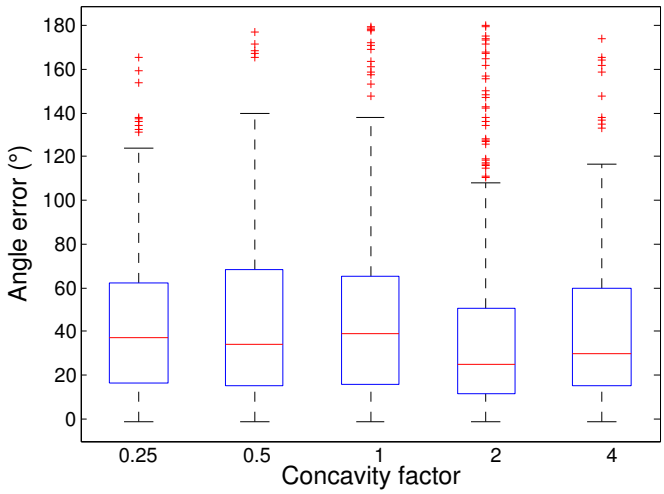
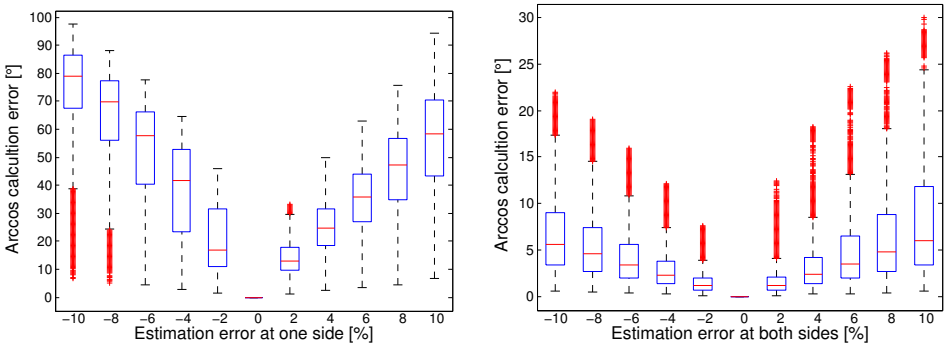


Figure 4.9: Angle θ estimation for five fields of different concavity. One thousand estimations were performed for each different field, in ten different networks. The error seems to not be affected by the concavity of the field but is higher than desired.

Nevertheless, even after these interventions the results are disappointing. Although the error seems not to be affected by the concavity of the field, in reality it is random. The median is $\approx 40°$, and due to the poor success rate of this algorithm (49%), shown in Figure 4.5, we deem the Cosine Law algorithm a failure.

These poor results pushed us to look into the relation of the *arccos* function and its inputs. We investigate whether an estimator exists for which the Cosine Law algorithm works. We will demonstrate with a simple Monte Carlo simulation, that even a small distance estimation error to the target N, as shown in Figure 3.1, can have significant effects on the calculation of θ. We simulated 10000 triangles with two sides being significantly longer than the third, representing the goal node being at a longer distance compared to the distance between two consecutive steps the algorithm took. We then added an estimation error on one of the two longer sides, expressed as a percentage of the initial length. The difference between the two angle calculations, one for the original triangle and one for the modified, is the error shown in axis y.

The results in Figure 4.10a display a massive angle error for the triangle where one of the long sides was just 10% longer or shorter. The spacial relation between the two consecutive points in the navigation path though, has a significant impact. In Figure 4.10b, we show that when the *same* distance estimation error is applied to both sides of the triangle, the error



(a) Distance estimation error at one side of the triangle.

(b) Distance estimation error at both sides of the triangle.

Figure 4.10: Angle calculation error caused by distance estimation error of either one or both sides of triangle of Figure 3.1.

is significantly lower. Our conclusion is that even when using a distance estimator with under 10% error, calculating the angle of the gradient is not possible using methods based on the *arccos* function and the law of the cosines. We have to stress that in a real-life scenario all conditions and field estimations are bound to be even worse due to assumptions that do not hold in practice.

The inability to estimate the distance to the goal pushed us to look into the algorithm presented next.

4.4 Constant Angle Algorithm Simulations

In the following, we present the results of the simulations run for the Constant Angle algorithm. We show that its performance is not affected by the field's concavity or on whether its iso-contours are concentric. We also demonstrate that a small step size results to less accumulated error. The metric used for this section is the relative length of the path traveled until entering the transmission radius of the goal node.

Field concavity. We start by evaluating the effect of field concavity on the Constant Angle algorithm. As it is seen in the Figure 4.11, the linearity of the gradient is of no importance. That is because the only important factor is monotonicity. The algorithm only evaluates whether the values of the field change, not the rate of change.

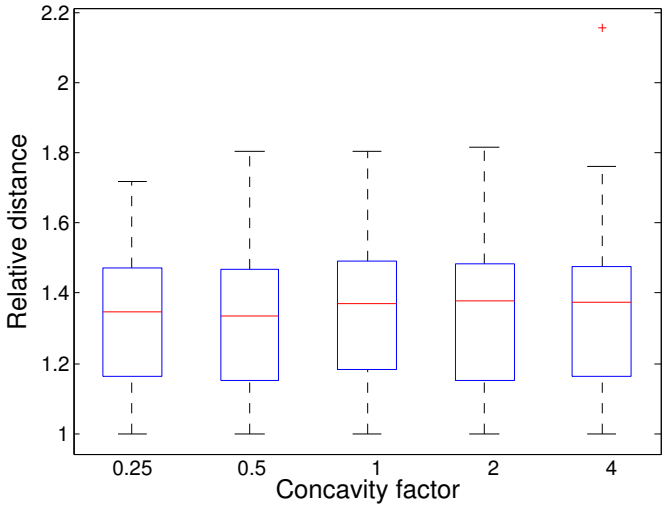


Figure 4.11: The relative path traveled length is not affected by the concavity of the field. We have run this simulation until reaching the goal 25 times for each field, starting from ten random locations in the simulation area.

Step size. The next simulation was performed to showcase the relation between step size and relative distance traveled to the goal. We chose the smallest step size based on an average human stride length in a crowded environment. This length is 30-40 cm, which translates to 2 m in the simulation environment in the same fashion as the 2 m actual transmission radius was translated to a 10 m transmission radius during simulations.

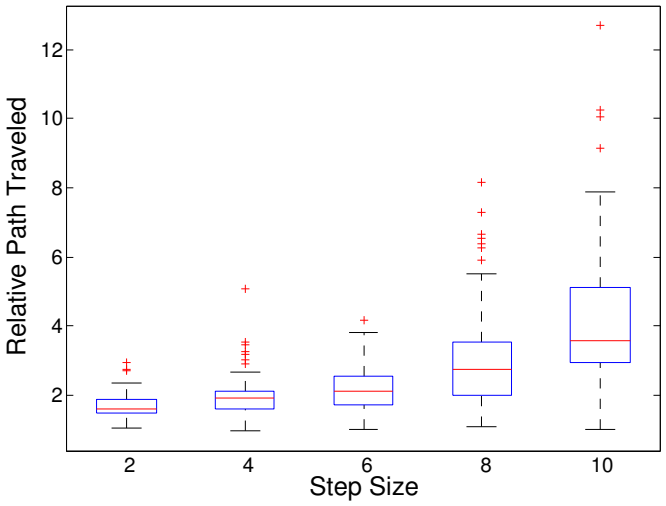


Figure 4.12: Relative path traveled for five different step sizes on the linear field. A smaller step size results in a shorter path to goal.

We can attribute the increasing error over the increase in step size presented in Figure 4.12, to two factors. The first is the penalty that occurs at every wrong decision. As stated in Subsection 3.2.1, whenever we choose to turn, we do that in a random fashion. If our decision turns out to be wrong, we need to step back and take a turn of 180°. This penalty is proportional to the step size.

The second factor is that a larger step size will often result in stepping further from the 90° ideal point that will lead the mobile node straight to the goal. Indeed, if the over-stepping is big enough, you might be forced to "circle" the goal without ever reaching it. An example of this phenomenon is shown in Figure 4.14.

Concentricity. As shown in Figure 4.3, the direction of the gradient of all the fields tested so far was always towards the end goal. With the following simulation, we show that the performance of the Constant Angle algorithm is not affected when the iso-contours of the field are not concentric.

We use the field that we created using the superformula mask on the linear field (see Figure 4.4a), and run the same simulation as before. A comparison between Figure 4.12 and 4.13, indicates that the Constant Angle algorithm is not affected by the superformula field. Due to randomness during the simulation space creation, we notice that the performance of the Constant Angle Algorithm is even better for some cases on the superformula field.

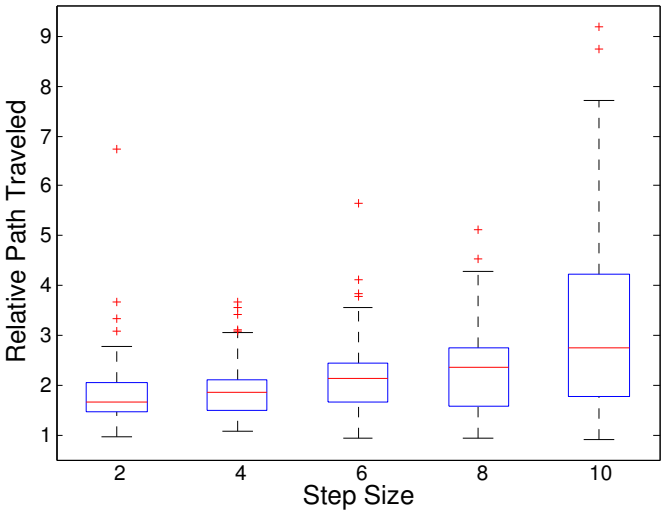


Figure 4.13: Relative path traveled for five different step sizes on the linear field and the superformula mask. The resulting relative path traveled is not affected by the non-concentric iso-contours of this field.

Overstep phenomenon. While running these simulations, an interesting property of the algorithm was discovered. Any step size equal or greater than the transmission radius may result in a route that will circle the goal and never reach it. As you can see in Figure 4.14, each new position has a smaller signal value than before so the decision taken is always taking a turn.

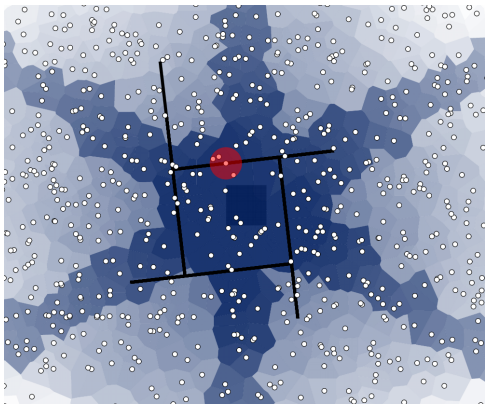


Figure 4.14: When the step size is equal to the transmission radius an interesting phenomenon may appear. The mobile node will circle the goal without reaching it.

A similar behavior can be noticed when we are on a steep ridge in the non-concentric field. If the width of the ridge is comparable to the step size, we can notice a similar infinite loop of the mobile node.

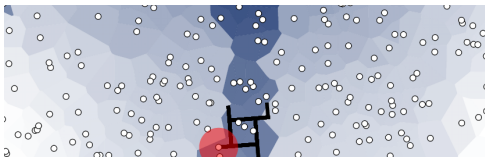


Figure 4.15: When the step size is comparable to the ridge width we may observe the same phenomenon as before. An infinite loop.

In both cases however, this problem can be easily corrected by adding randomness to each step or just by using an angle $\neq 90°$. This behavior can only be noticed while simulating the scenario, since in practice the field value of the mobile node varies over time. The mobile node calculates the field's value by averaging the values of its neighbors (Equation 2.2), and because of non-deterministic wireless links dropping packets at random, this value will be a function of time as well as space. Another factor is the error added by assuming that the step size is constant. In practice, we can measure the step size only by using the inertia measurement unit of the mobile phone. In addition, the angle the user has to turn is also dictated by the phone. These procedures are also estimations that entail error.

Chapter 5

Experimental Evaluation

This chapter presents an implementation of the Constant Angle algorithm on a testbed of 108 nodes. We start by explaining the system architecture in Section 5.1. Then, in Section 5.2, we describe the necessary engineering steps taken to ensure the algorithm's proper evaluation and present the results of the implementation.

5.1 System Architecture

A realistic experiment for the crowd management scenario would require all the nodes of the network to be mobile. However, constructing a controlled experiment with mobility is very difficult. For this reason, we used a testbed of 108 nodes as the sensor network hosting the field that the user has to navigate on.

5.1.1 Testbed

The testbed, which is referred to as *The Rack* [25], is a wireless sensor network consisting of 108 SOWNet GNode nodes. The GNode G301 is comprised of a TI MSP430F2418 microcontroller and a TI CC1101 packet radio chip operating at 868 MHz. Each GNode has a backbone network connection to a PC Engine Alix.1D. A testbed server connects to the Alix to collect serial output and debugging data, and to program the nodes.

As shown in Figure 5.1, *The Rack* is deployed on the ceiling of our office building at TU Delft. On this floor plan, we can see that each room has either four or eight nodes on its ceiling, indicated by the black dots. However their exact location was not known, which was a problem since we wanted to assign specific values to each one in order to create a signal field with precisely defined characteristics.

To solve this, we created a sniffer node that sensed the incoming packets from the nearby nodes. We then used the number of messages as well as the

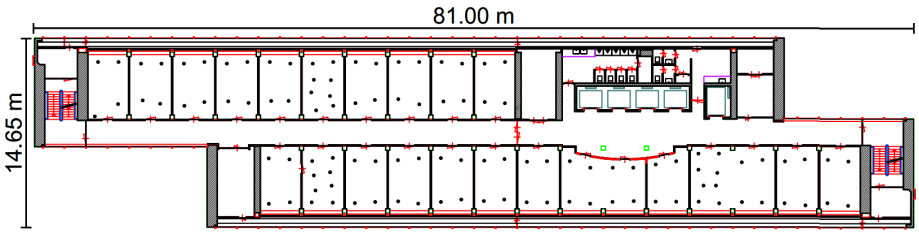


Figure 5.1: Floor plan of our offices at TU Delft. Black dots represent the nodes installed on the ceiling.

RSSI value of each message to localize each sensor node. This allowed us to visualize the connectivity graph as well as any field we created.

After roughly localizing the nodes at corners of the ceiling of each room, the connectivity graph for a transmission power of -20dBm was created as shown in Figure 5.2. We chose this tx power in order to achieve the same node density as our scenario and simulations. The average node degree was approximately 30. Note that each link was considered active when the packet reception rate was greater than 0.3.

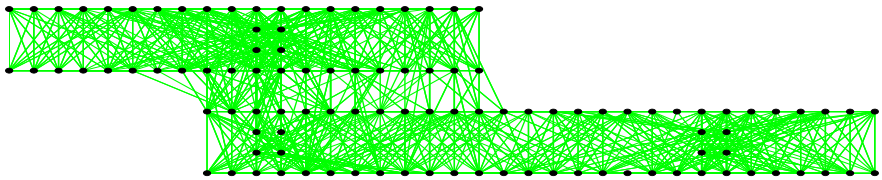


Figure 5.2: Active links for -20dBm transmission power and packet reception rate ≥ 0.3.

The nodes were programmed with the Contiki operating system to simply broadcast a preset value. In Figure 5.3 we show the field that we created using these preset values by varying the greyscale intensity.

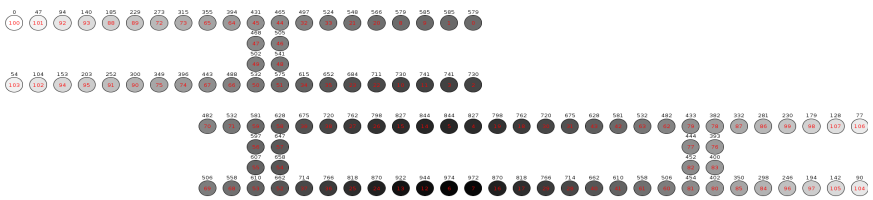


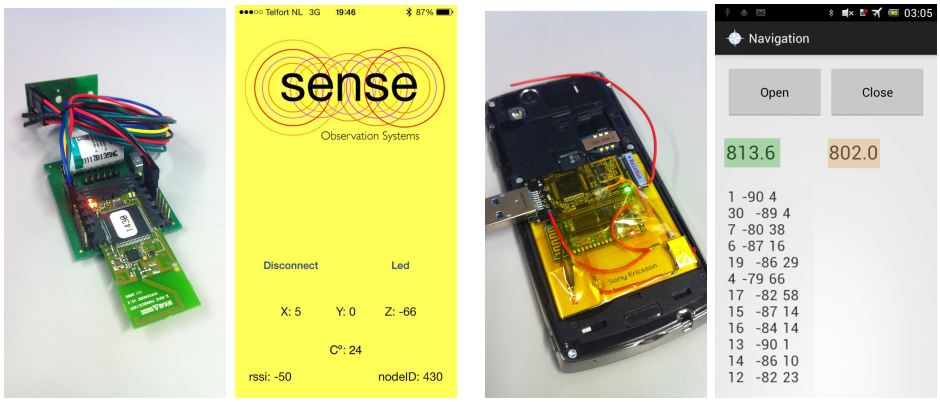
Figure 5.3: Value visualization on testbed nodes.

5.1.2 Navigation Algorithm Interface

In our experiment, a user is navigated to the goal by following the instructions on her smartphone. The latter is equipped with bluetooth for the communication needs with the sensor network.

In the beginning, we considered using MyriaNode v3.2 nodes to build the sensor network. In order to connect a smartphone with them we interfaced them with a nRF8001 Bluetooth Low Energy radio. An application to demonstrate this connection was developed for the iOS platform, which would connect to the node and receive data from an on-board accelerometer. Based on this data, the color of the background was changed in real time and other values such as temperature and RSSI were displayed (Figure 5.4b).

Due to lack of time, we decided to use the already existing testbed at TU Delft instead of setting up a new one with the MyriaNed nodes. Since equipping 108 nodes with BLE radios would be neither fast nor cheap, we used an extra GNode as the mobile node. The connection to the smarthpone was realized using a legacy bluetooth radio. However, iOS does not provide access to the legacy bluetooth stack, so we had to create our GUI on the android platform. For practicality and a power source, the node and the bluetooth-to-serial board were attached to the phone's battery. The number highlighted in green in Figure 5.4d, shows the value of the field using a weighted moving average. The value on its right is the unfiltered one. Finally, the list below them displays packet information. In particular, for each sender, the id, RSSI and number of packets received during the last second.



(a) MyriaNode V3.2 & nrf8001. (b) iPhone demo application. (c) Gnode and bluetooth radio. (d) Graphical interface on Android.

Figure 5.4: Two testbed-smartphone interfaces were built. On the left, one using Bluetooth Low energy and iOS and on the right, one using Legacy bluetooth and Android.

5.2 Experiments

The goal of the experiments is to prove that the Constant Angle Algorithm can navigate a user to the peak of an artificial gradient. Unfortunately, we are confined by the experiment setup: due to the existence of walls between the rooms, one cannot move about freely. The only possible pathways are along the 80 meter corridor as seen in Figure 5.1 and going in and out of rooms. *Therefore, the goal of each experiment is to navigate a person that is walking along the corridor to a predefined room.*

Field creation. The goal so far was to reach the communication radius of the target node. Since that would be achievable just by walking in the corridor, we created a static field with the maximum being on the edge of a room as shown in Figure 5.5.

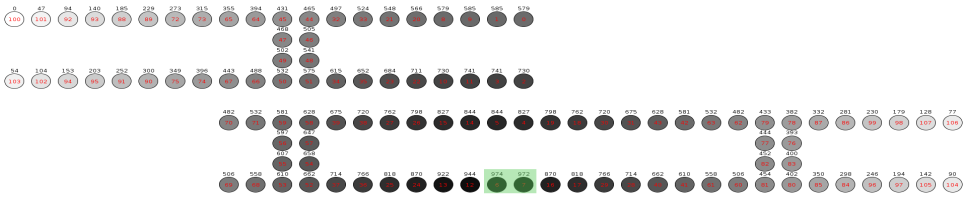


Figure 5.5: Visualization of the final field used for the experiment. Notice that the peak is at the outer side of the room highlighted in green.

Field sensing. The first noticeable phenomenon is that f is not a function of only space, but also of time. This is because in practice the communication model on sensor nodes reveals various spatial and temporal radio irregularities [31, 32]. This subsequently causes the delivery rate to drop to about 68% on a real testbed [14]. Furthermore, the relative position of the sensor node in regard with the smartphone as well as its orientation significantly alters its packet reception rate. We noticed this phenomenon during the experiments. Its magnitude depends on the relative angle between the mobile node's antenna and the static ones'. To mitigate this problem, we carried the experiments in a way the mobile node had a constant direction.

We perform the following experiment to demonstrate the difficulty in determining the field value: we walk the floor's corridor lenghtwise at a constant speed of $1m/s$ while logging all received packets. The received packets comprise the raw signal, which is plotted against time in Figure 5.3 together with the output of two filters. The first is a moving average filter that has a window of 5 seconds that averages one packet for every unique packet id. The second is a weighted moving average filter that averages all packets in the same 5 second window, regardless of the number of occurrences of the same packet id. The mobile node receives packets from nodes with different values sequentially, which appears as noise on the plot.

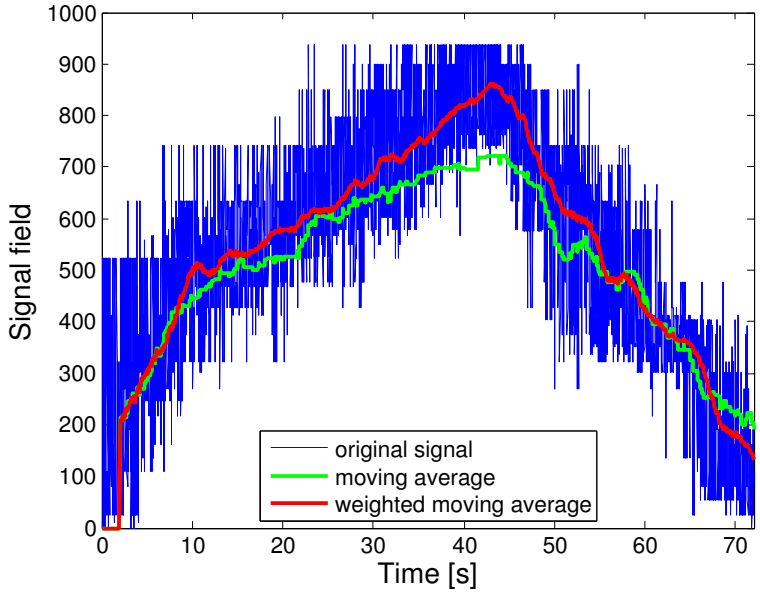


Figure 5.6: Raw and filtered signal. The take away message here is that we are interested in acquiring a signal with only two monotonic parts. Since the mobile node's speed is $1m/s$, the x axis is also the position in meters on the map in Figure 5.1.

What we are actually interested in, is the monotonicity of the field. Figure 5.6 shows that the output of the moving average, and the weighted moving average has fewer higher frequency components. The weight of each value is the number of packets received in a window frame. How big should the averaging window be though? If we increase it, we risk adding a lot of delay. This will mean that we might miss the ideal point of turning. However, we have to note that, since the UDG model is a crude approximation, a 90° turn does not guarantee a direct route to the goal anyway.

In the Constant Angle algorithm, the decision to make a turn is solely dependent on the monotonicity of the field. When we are traversing a scalar field that has one global maximum, the value of the field increases monotonically until reaching the maximum point of this trajectory and then decreases monotonically. Take for example the experiment in Figure 5.6. In the absence of noise, the signal has only maximum, which is the global maximum. We define as a false positive any local maxima in the signal, since they would trigger a turn action. In Figure 5.7, we plot the false positives over window size for 5 runs of the previous experiment. The filter used is the weighted moving average. We can deduce that a window size of 4 seconds is enough to filter out most of the false positives without adding a lot of delay. This however is true for only the particular walking speed and node transmission power.

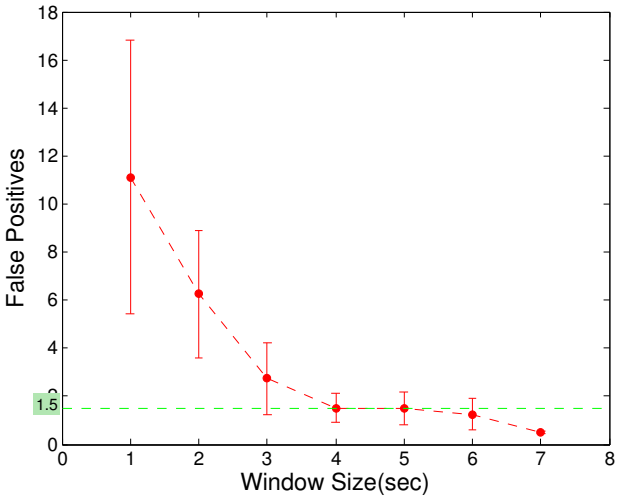
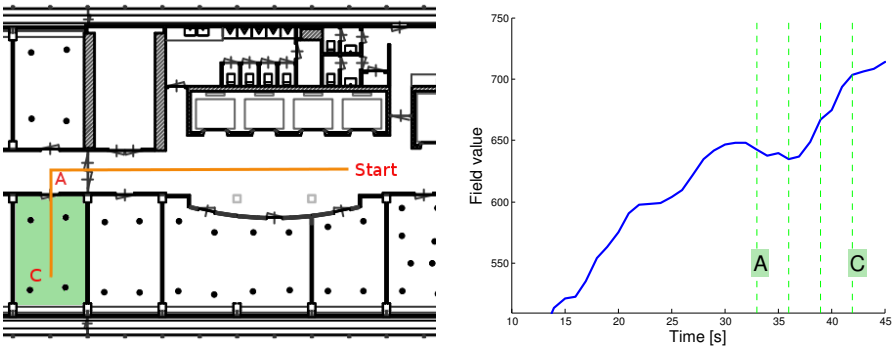


Figure 5.7: False Positives vs Window Size. A window size of 4 seconds is enough to filter out most of the false positives.

Results. The goal of the experiment is to navigate to the room highlighted in green, as shown in Figure 5.8a. The signal that is being plotted next to it is the filtered field measurements. The weighted moving average filter was used, with a window of 5 seconds. In order to simulate the same step size as before, we do the same calculation as in Section 4.4. To achieve an average node degree of 30, the transmission radius of the nodes was set to approximately 20 meters. The minimum step size of the algorithm for this transmission radius is 3 meters. We have to note here, that the termination criterion of the original algorithm was for the mobile node to enter the communication vicinity of the goal node. This is achievable by just walking along the corridor. The new termination criterion is thus more demanding.

We run the same experiment 10 times, always starting from the area in front of the elevators, as shown in Figure 5.8a. We reached the target 8 times, the other 2 overshot and entered the next room. The average false positives for these runs was 3.2. Next, we present two characteristic examples of navigation to the target room, one where the first turn decision was correct and one that was wrong. In both cases, the goal was reached successfully.

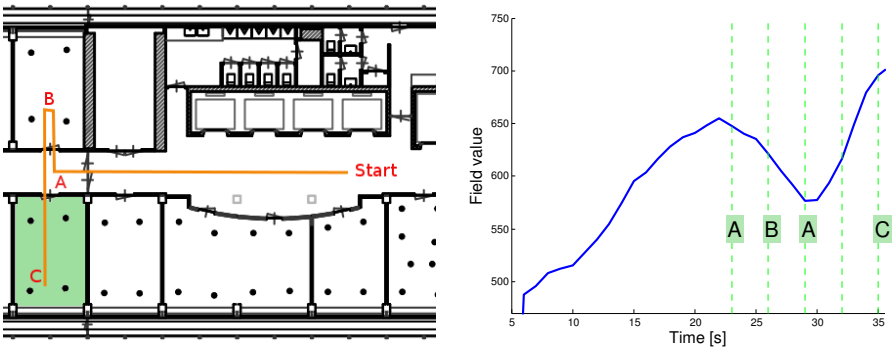
In the first example, shown in Figure 5.8, the turn action was decided at point A, where the averaged field value started decreasing. Apart from the delay added by the filter, stopping and turning in place is slower than walking at $1m/s$. That is why although the user chose the right direction, the field values started increasing only after two steps of the algorithm as is shown in Figure 5.8. Point C is then reached after another step.



(a) Path traveled. (b) Filtered field.

Figure 5.8: First experiment.

We noticed the same delay when the user chose the wrong room in the second example. In Figure 5.9, the user first turned towards the room opposite the target, but as soon as the field value decreased even more, she turned again towards the target room. The field value started increasing after two steps as before due to the same delay.



(a) Path traveled. (b) Filtered field.

Figure 5.9: Second experiment, wrong initial decision.

Although we demonstrate that the Constant Angle algorithm achieves its goal, we have to point out that this scenario was different from the target scenario of the festival in various ways. The inability to choose a random direction and walk about freely is restricting, as well as the fact that the field and the WSN were static. On the other hand the termination criterion was a lot more demanding than the one during simulations, and the irregularities of wireless links indoors are far greater than outdoors.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The core research question of this thesis was how to navigate to the extremum of a scalar field monitored by a wireless sensor network when no location information is available. Additionally, we identified two necessary requirements for out motivating scenario of crowd management at festivals: the steps that lead an attendee to the goal have to form a natural path, and the algorithm has to be independent of the field's gradient characteristics.

Although a large body of work exists, mainly from the area of robotics, none of these solutions can handle the nonlinear fields following a natural path. Demonstrating this was our first contribution. We proposed a simple algorithm for that, which we call Cosine Law algorithm, that uses a Taylor's expansion to estimate the distance to the goal. Through simulations we performed, we showed that estimating distance is possible, but exhibits poor performance on nonlinear fields. Estimating the angle of the gradient was shown to be very difficult even under ideal conditions.

Our second contribution was tackling the problem using an even simpler algorithm, which we call the Constant Angle algorithm. This algorithm does not try to estimate the angle of the gradient, rather only its sign. Whenever this becomes negative, a constant angle will be added to the direction of movement to correct it. Through simulations we demonstrated that it always reaches the goal while displaying only a 50% increase in distance traveled to the goal in comparison to the euclidean.

However, simulations and real life differ greatly. Our third and most important contribution was the implementation of the Constant Angle algorithm on real hardware. We designed a demonstrator, where a person had to be navigated to a room of our floor, where a wireless sensor network testbed exists. This experiment exhibited mobility, and the same network

density as in the simulations. We showed that in practice, due to radio irregularities, the scalar field is described by a function of not only space, but also time. Before following any navigation steps, noise caused by these irregularities had to be filtered. We used a weighted moving average filter taking the number of recent packets from each node as its weight. This way we managed to remove the noise, and use the signal to navigate to the correct room.

The take away message of this thesis is that estimating the gradient's angle on nonlinear fields while following natural paths is *unfeasible*. However, navigating to the extremum of a scalar field in a location-free way, following a natural path, independently of the characteristics of the field is possible, while only adding a small distance overhead.

6.2 Future Work

The problem we investigated is complex and can be approached in many different ways. We have identified numerous points that need improvement.

- Noise filtering at the smartphone side. State of the art filter that exploit the movement of the user can improve the accuracy. We can use other metrics *if* they are available from the hardware such as the RSSI of packets to choose the best links.

- Added noise during simulations. In order to clearly understand the effects of the field's characteristics on the performance of the algorithms we chose not to add any noise to the field during simulations. This is a required step towards improving the quality of this work.

- Experiment setup. An experimental evaluation using a testbed that would not interfere with the motion of the mobile node will allow for a better understanding of the algorithm in a scenario similar to the festival. If the experiment took place outdoors, the localization hardware of the phone could be used to provide the ground truth for the motion of the mobile node.

- Antenna directionality. During the experiments, we noticed large variations of packet reception rate depending on the angle of the mobile node in relation to the static ones. In the festival scenario, the antenna on the nodes should be as omni-directional as possible.

- Signal attenuation. Although the radio we used operates at 868 MHz, there was some signal attenuation due to the user's body. This can be exploited for locating an RF source using one's body as a directional antenna as in [30]. For higher frequency radios, this phenomenon will have a greater impact.

- Mobility. Due to radio irregularities we had to filter a static signal field measured by a static network. How well will the navigation algorithm we propose perform on scalar fields created by highly mobile sensor networks? We want to investigate this by testing on either a mobile testbed or a time-varying field.

Bibliography

[1] Kartik B Ariyur and Miroslav Krstic. *Real-time optimization by extremum-seeking control*. John Wiley & Sons, 2003.

[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] Emrah Bıyık and Murat Arcak. Gradient climbing in formation via extremum seeking and passivity-based coordination rules. *Asian Journal of Control*, 10(2):201–211, 2008.

[5] Tatiana Bokareva, Wen Hu, Salil Kanhere, Branko Ristic, Neil Gordon, Travis Bessell, Mark Rutten, and Sanjay Jha. Wireless sensor networks for battlefield surveillance. In *Proceedings of the land warfare conference*, 2006.

[6] M Castillo-Effer, Daniel H Quintela, W Moreno, R Jordan, and W Westhoff. Wireless sensor networks for flash-flood alerting. In *Devices, Circuits and Systems, 2004. Proceedings of the Fifth IEEE International Caracas Conference on*, volume 1, pages 142–146. IEEE, 2004.

[7] Jennie Cochran and Miroslav Krstic. Nonholonomic source seeking with tuning of angular velocity. *Automatic Control, IEEE Transactions on*, 54(4):717–731, 2009.

[8] Nikhil Deshpande, Edward Grant, and Thomas C Henderson. Experiments with a pseudo-gradient algorithm for target localization using wireless sensor networks. In *Multisensor Fusion and Integration for Intelligent Systems (MFI), 2010 IEEE Conference on*, pages 74–79. IEEE, 2010.

[9] Yotam Elor and Alfred M. Bruckstein. Two-robot source seeking with point measurements. *Theoretical Computer Science*, 457(0):76 – 85, 2012.

[10] Jabed Faruque and Ahmed Helmy. Rugged: Routing on fingerprint gradients in sensor networks. In *Pervasive Services, 2004. ICPS 2004. IEEE/ACS International Conference on*, pages 179–188. IEEE, 2004.

[11] Jabed Faruque, Konstantinos Psounis, and Ahmed Helmy. Analysis of gradient-based routing protocols in sensor networks. In *Distributed Computing in Sensor Systems*, pages 258–275. Springer, 2005.

[12] Marco A Gonzalez, Javier Gomez, Miguel Lopez-Guerrero, Victor Rangel, and Martha M Montes de Oca. Guide-gradient: A guiding algorithm for mobile nodes in wlan and ad-hoc networks. *Wireless Personal Communications*, 57(4):629–653, 2011.

[13] Bernhard Hofmann-Wellenhof and Herbert Lichtenegger. *Global positioning system: theory and practice*. DIANE Publishing Inc., 1993.

[14] Young-Jin Kim, Ramesh Govindan, Brad Karp, and Scott Shenker. On the pitfalls of geographic face routing. In *Proceedings of the 2005 Joint Workshop on Foundations of Mobile Computing*, DIALM-POMC '05, pages 34–43, New York, NY, USA, 2005. ACM.

[15] Gideon Kowadlo and R Andrew Russell. Robot odor localization: a taxonomy and survey. *The International Journal of Robotics Research*, 27(8):869–894, 2008.

[16] Andreas Loukas, Marco Zuniga, Matthias Woehrle, Marco Cattani, and Koen Langendoen. Think globally, act locally: on the reshaping of information landscapes. In *Proceedings of the 12th international conference on Information processing in sensor networks*, pages 265–276. ACM, 2013.

[17] Alan Mainwaring, David Culler, Joseph Polastre, Robert Szewczyk, and John Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 88–97. ACM, 2002.

[18] Alexandre R Mesquita, Joao P Hespanha, and Karl Åström. Optimotaxis: A stochastic multi-agent optimization procedure with point measurements. In *Hybrid Systems: Computation and Control*, pages 358–371. Springer, 2008.

[19] Brandon J Moore and Carlos Canudas-de Wit. Source seeking via collaborative measurements by a circular formation of agents. In *American Control Conference (ACC), 2010*, pages 6417–6422. IEEE, 2010.

[20] Jaspal S Sandhu, Alice M Agogino, Adrian K Agogino, et al. Wireless sensor networks for commercial lighting control: decision making with multi-agent systems. In *AAAI workshop on sensor networks*, volume 10, pages 131–140, 2004.

[21] Xiao Sun, Xiaoguang Zhao, En Li, Hongchun Li, and Zize Liang. Mobile robot navigation using rssi potential field in wireless sensor network. *Journal of Computational Information Systems*, 6(14):4751–4759, 2010.

[22] 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.

[23] 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.

[24] 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.

[25] M. Woehrle, M.C. Bor, and K.G. Langendoen. 868 Mhz: a noiseless environment, but no free lunch for protocol design. In *9th int. conf. on Networked Sensing Systems*, INSS, pages 1–8, jun 2012.

[26] Fan Ye, Gary Zhong, Songwu Lu, and Lixia Zhang. Gradient broadcast: A robust data delivery protocol for large scale sensor networks. *Wireless Networks*, 11(3):285–298, 2005.

[27] F. Zeiger, N. Krämer, and K. Schilling. Commanding mobile robots via wireless ad-hoc networks - a comparison of four ad-hoc routing protocol implementations. In *IEEE International Conference on Robotics and Automation (ICRA), 2008, Pasadena, USA*, pages 590–595, 2008.

[28] Chunlei Zhang, Daniel Arnold, Nima Ghods, Antranik Siranosian, and Miroslav Krstic. Source seeking with non-holonomic unicycle without position measurement and with tuning of forward velocity. *Systems & control letters*, 56(3):245–252, 2007.

[29] Zheng Zhang, Zhenbo Li, Dawei Zhang, and Jiapin Chen. Path planning and navigation for mobile robots in a hybrid sensor network without prior location information. *International Journal of Advanced Robotic Systems*, 10:1–12, 2013.

[30] Zengbin Zhang, Xia Zhou, Weile Zhang, Yuanyang Zhang, Gang Wang, Ben Y Zhao, and Haitao Zheng. I am the antenna: accurate outdoor ap location using smartphones. In *Proceedings of the 17th annual international conference on Mobile computing and networking*, pages 109–120. ACM, 2011.

[31] Jerry Zhao and Ramesh Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 1–13. ACM, 2003.

[32] Gang Zhou, Tian He, Sudha Krishnamurthy, and John A. Stankovic. Impact of radio irregularity on wireless sensor networks. In *Proceedings of the 2Nd International Conference on Mobile Systems, Applications, and Services*, MobiSys '04, pages 125–138, New York, NY, USA, 2004. ACM.