Evading sensor detection by using continuum approximation to generate optimal path

Arafat Hussain
201488745

Prof. Kurt Langfeld

Submitted in accordance with the requirements for the module MATH5872M: Dissertation in Data Science and Analytics as part of the degree of

Master of Science in Data Science and Analytics

The University of Leeds, School of Mathematics

December 2021

Abstract

This study develops a continuum approach to generating an optimal path solution in a probability landscape by minimizing an associated cost function. The probability landscape pertains to the probability of detection in an area containing a target which is being guarded by the network of sensors. The study involves a deep dive into the formulation of the mathematical model depicting the scenario, using continuum approximation for which thorough mathematical derivations have been shown as well as the optimisation technique used (BFGS). Studies have also been conducted to analyse the impact of the constraints which have been accounted for in the cost on the optimal path solution. A demonstration of multiple intrusion scenario into the guarded landscape has also been shown. Keywords - Sensor, intruder, detection, registration, probability, landscape, path.

Contents

List of Figures

List of Tables

1 Introduction

In many real-world scenarios, sites of special importance are guarded by sensors of various kinds. Intrusion sensors are designed to determine whether an unauthorized individual has gained entry to, or attempted to gain access to, a protected area (Gracia 1999). Numerous sensor types can be installed around the perimeter of a building, within a smaller region within the facility, or on a specific location or object (Morris 2003).

Human intrusion can be detected by various different kinds of sensors. Some of the relevant ones are magnetic, seismic, acoustic optical, thermal and chemical. All of them are passive sensors, in contrast to radar or ultrasonic sensors, in that they do not produce a signal and monitor how it is modified by targets. In sensor networks with little energy, passive sensors are preferable. Magnetic sensors presumptively presume that the intruder, for example, an armed individual, possesses magnetically sensitive material. Ferromagnetic materials provide a distinctive magnetic signature that may be identified with a magnetometer. Electromagnetic methods can be used to identify any metallic material worn by the intruder. Seismic and acoustic sensors are triggered by the intruder's vibrations. Both are classified as vibration sensors (Kumar et al. 2010).

Vibration-based surveillance sensors fall into two broad categories: sound sensors and motion sensors. Acoustic sensors detect and monitor the sound generated by the thing being detected or monitored. In cars, the primary sources of noise are the engine and power train, the track/tyre, and the exhaust. In addition to the verbal sound of the creature, other sounds such as human and animal footsteps, birds fluttering their wings, and so on make sound. Acoustic sensors are mostly microphones and hydrophones. Vibrational motion sensors, on the other hand, detect displacement, velocity, and acceleration using seismometers/geophones, velometers, and accelerometers.

Additionally, there may be interaction between acoustic noise and ground vibrations in the case of large vehicles. Acoustic waves travel at varying speeds and their amplitudes diminish or are absorbed at varying rates with distance. This assists in determining the sort of vehicle or other source of noise. Hence both acoustic and vibration sensors are needed in a surveillance operation. A notable example of this is seen in an extensive research titled ”Bochum Verification project for Military Vehicle Detection” (Altman 2004), which was done to recognise vehicles in a variety of situations. Automobiles, small and big trucks, armoured personnel carriers, and combat tanks were all covered in this survey. This study included a variety of microphones, an accelerometer, and a geophone. While one accelerometer, six geophones, and two microphones were required for an experiment on a tarmac road with sandy soil, an experiment on a concrete road with weathered layer of old lava required 27 geophones and four microphones. The direction of arrival may be determined by using various sensors and signal processing.

Optical and thermal sensors both operate on the basis of disruption in the sensors' line of sight. Humans, animals, and vehicles all have 'hot spots' or unique thermal fingerprints that differentiate them from foliage and structures and allow for detection. Chemical sensors are based on the presence of certain chemical species linked with the invader. While humans do leave a chemical trail, recognizing it takes the skill of trained dogs and a sophisticated sensor array capable of detecting a wide variety of chemical species (Arora et al. 2004).

Wireless sensor networks have garnered considerable interest due to their critical function and numerous benefits in our daily lives. Indeed, recent and rapid advances in inexpensive sensor technology and wireless communications have made the design and development of large-scale wireless sensor networks affordable and appealing for a variety of mission-critical situations, including civilian, natural, industrial, and military applications, such as health and environmental monitoring, seism monitoring, industrial process automation, and battlefield surveillance (Ammari 2009). A wireless sensor network is made up of a large number of small, low-power devices called sensors that are distributed randomly or deterministically in an area of interest while cooperating and coordinating to complete their purpose successfully.

It is important to note that a single sensor might not be enough to completely meet the needs of a surveillance application. It requires installation of a network of well placed sensors, suitable for detecting human intrusion around the area of interest for its effective surveillance, while taking into consideration the range of detection of the sensors.

1.1 Background

With a suitable network of sensors, guarding an area of interest containing a target from un-authorized intrusion, we can imagine a scenario where one or more intruders might attempt to gain un-authorized access to the target. However, in order to do so successfully, one has to evade getting detected by the sensors. The scenario takes into consideration the probability of detection by the sensor network as well as the velocity with which the individual is moving in order to reach the target. These two factors generate a cost associated to the mission of the intruder to reach the target which has been discussed in detail in the subsequent sections.

Our primary goal as a part of this piece of research is to generate an optimal path for the intruder to take in order to reach the target with minimal chances of detection for the given set of mission constraints.

Through this piece of research, we aim to address the following questions for DSTL and PA consulting:

- What is the optimal path that intruder can take from a starting position to the target being guarded by the sensors ?
- How does a given time of travel of the journey to the target affect the optimal path ?
- What impact do the constraints on travel speed have on the optimal path ?

1.2 Mathematical setup

In this section, we shall discuss the mathematical setup that has been considered in order to simulate the scenario of a network of sensors guarding the target from unauthorized intrusion. We shall call the ability of detection of an intruder by a single sensor as registering with the sensor. It would require an event of registering with at least two sensors for the network of sensors to successfully detect the presence of an intruder. This measure has been taken in order to avoid generating a false alarm or a false positive. The requirement of two sensors to register the presence of an intruder in order to confirm an event of intrusion is a design principle given by DSTL.

The probability of registration by a single sensor is given by the formula –

$$P^{r}_{reg} = \frac{1}{1 + (\frac{r}{\sigma})^2} \tag{1}$$

Where **r** is the distance between the location of which we want the determine the probability of getting registered at, and any sensor in the network. The σ here in the equation, is a constant which denotes the range in units of distance of detection of a sensor.

The positioning of the sensors in the network has been carefully chosen so as to produce a conducive environment of guarding the target. In order to simulate the scenario, a two-dimensional plane on the X and Y axis of the size 10 x 10 has been considered. An array of 10 sensors has been placed at positions - (1,1), (4,3), (5,1), (2,7), (4,9), (7,3), (7,9), (9,1), (9,4), (9,7) with the target being guarded placed at position (5,6). The below figure depicts the arrangement of the sensors and the target where, the red dots signify the sensors and the green dot signifies the target.

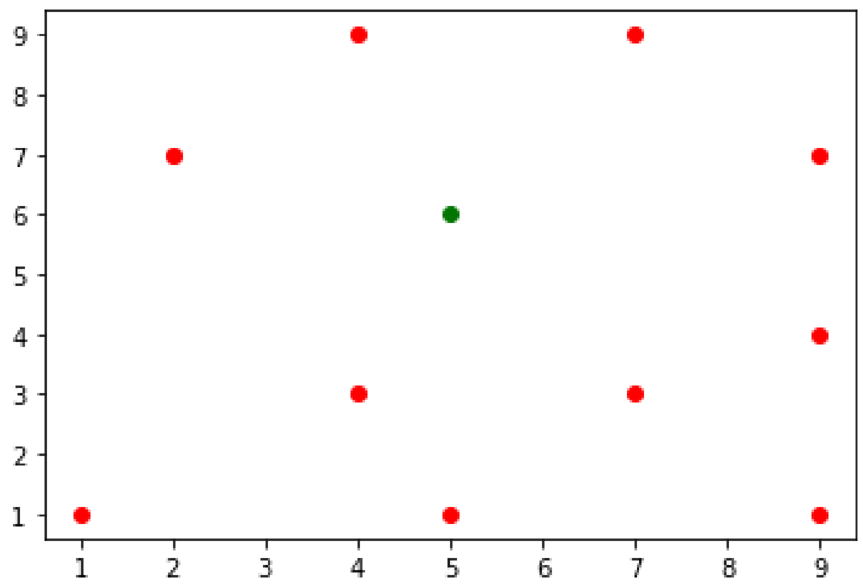


Figure 1: Sensor Network Arrangement

With a full knowledge of the sensor network mentioned above, an intruder has several options to reach the target. For example - the intruder could approach the target on foot or on a vehicle in order to reach the target swiftly. Theoretically the intruder could avoid being detected if they move with infinite velocity towards the target. However, this scenario is infeasible in practice.

With respect to the aforementioned discussion, at this stage it is important to mention that this optimization problem is modelled mathematically using a cost function. The cost function is primarily comprised of two components: a velocity component and a component that determines the probability of detection along the path of traversal. Combined value from these two components of the cost function gives us the overall cost of traversal on a given path. Our goal is to minimize this cost and thereby produce the optimal path feasible for the intruder to reach the target.

Since, there is a cost associated with the speed of traversal, meaning the higher the velocity, the more is the cost. The weightage to be given to the velocity component of the cost has been left at the discretion of the stake holders. This also implies that, if the velocity component is given a higher weight, the overall cost would be high if the intruder moves with high velocity. In that case the intruder would have little option to use a vehicle to move faster. However, if the velocity component on the other hand is given less weight, the intruder can take the advantage of speed by using a vehicle in order to reach the target. In both the cases the optimal path to be taken in order to reach the target changes which can be seen in the later sections of this report.

2 Mathematical modelling

Mathematical modelling and simulation are critical techniques in engineering and research. These days, everyone has access to powerful desktop PCs. These machines are suitable for conducting any type of professional data analysis. Even complicated structural, mechanical and fluid dynamical simulations that formerly needed supercomputers may now be conducted on desktop PCs (Velten 2008). Given modelling and simulation's enormous potential for solving complicated issues and saving money, one should anticipate widespread and professional adoption of this technology.

In this section of the document, we shall detail out the modelling approach taken in order to simulate this real-world scenario discussed in the previous section and solve this optimization problem using mathematical and computational techniques.

Given below is an illustrative example of this challenge. It is important to note that the sensor positions in this image are not the ones we are working with. This illustration has only been used in order to provide more clarity to the reader's understanding of the challenge. In reality we are still working with the sensor arrangement mentioned in the section 1.2 above.

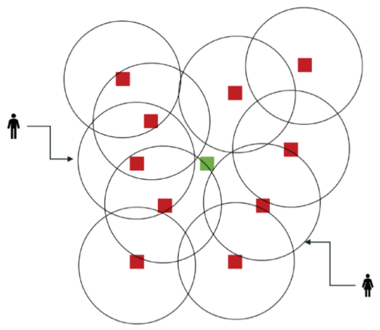


Figure 2: Illustration of the challenge of Escape the sensor

The red squares in the figure above denote the sensors while the green square denotes the target that is being guarded by the sensors. The images of humans in the figure depict the intruders attempting to breach into the target. The circles around the red squares denote the range of detection of each sensor. More specifically the probability of registering with a particular sensor is depicted by its corresponding circle. This implies that the farther the intruder is from a particular sensor the less is his probability of getting registered with that sensor. Mathematically this is in accordance to the equation (1) mentioned earlier in section 1.2.

It is worth reemphasizing that an event of detection of an intruder can occur only if the intruder is registered with at least two sensors. This means that the overlapping regions of the circles are the ones stepping into which, by the intruder, will cause a detection event to occur. The intruder must carefully avoid these regions on his way to the target in order to avoid being detected by the sensor network.

2.1 The Continuum Approach

Continuum modelling approaches have extensively been used in cost optimisation problems. There has been a number of applications of this approach in works related to equilibrium prediction in traffic related problems to determine traffic flows while taking into consideration the various aspects of a traffic network such as travel cost, flow intensity, demand. Similar work has been done by a researcher on Dynamic Traffic Assignment for autonomous vehicles while taking into consideration the impact on environment, by traffic emissions (Rong et al. 2018).

A discrete modelling approach to solve the optimal path problem in our project, would require us to model each probable path within the network separately. This is a conventional methodology used in detailed planning and analysis of travel patterns and road network systems (Ho, Wong, and Loo 2006). However the problem that we are trying to solve does not need detailed planning and analysis of the sensor network and its landscape at this stage but rather focuses on the pattern of distribution of the cost and travel choices for the intruder. This is where continuum approximation approach is efficient in which these aspects are worked upon at a macroscopic level rather than at a detailed level. In continuum approach, a dense network is approximated as a continuum in which users are free to choose their routes in a two-dimensional space. The fundamental assumption is that the differences in the modeling characteristics of adjacent areas within a network, such as travel cost are relatively small in comparison with the variations over the entire network (Jiang et al. 2011). Hence the characteristics of the network, such as probability of detection, cost of travel etc can be represented by smooth mathematical equations. Moreover the continuum approach can reduce the problem size. In continuum approach, the problem size depends on the approximation technique used to approximate the modelling region rather than on the the number of paths modeled. Any effective approximation technique such as finite element method, simulated annealing etc., can significantly reduce the problem size. The reduction in problem size saves computational cost. Moreover, less data is required for the model setup process in a continuum model. As a continuum model can be characterized by a small number of spatial variables, it can be set up with a much smaller amount of data than that needed under the discrete modeling approach, which requires data for all of the probable paths.

In this section of the report, we shall discuss the formulation of the continuum mathematical model developed in order to solve the optimal path problem discussed earlier. It closely follows (Atayev and Delaosa 2021).

Throughout this section, we shall assume a two-dimensional working domain on the $\mathbf{xy}$ plane, with all relevant objects lying on the plane.

Let a point in the $\mathbf{xy}$ plane denoted by $\mathbf{X}$ be the target which is being guarded by the sensors. Therefore the target position is mathematically denoted by $\mathbf{X} \in \mathbf{R^2}$. For the current scenario of our project, this is placed at $(\mathbf{5}, \mathbf{6})$ on the $\mathbf{xy}$ plane. This target is guarded by 10 sensors placed at different locations on the $\mathbf{xy}$ plane as seen in Figure 1. However, for generality let us consider a network of $\mathbf{N}$ sensors guarding the target at $\mathbf{X}$ and for $\mathbf{i} = \mathbf{1}$ to $\mathbf{N}$, let $\mathbf{x_i}$ be the position of each sensor. As discussed in the previous section, the probability of registering with a particular sensor is given by the equation (1). Given by the same equation (1), let $\mathbf{p_i}$ be the probability of registering with the $\mathbf{i_{th}}$ sensor. The values of probability of registering with a sensor $\mathbf{i}$, range between $\mathbf{0}$ and $\mathbf{1}$ with values closer to $\mathbf{1}$ meaning high probability of registering while values closer to $\mathbf{0}$ meaning low probability of registering by the sensor. We have seen in the equation (1), how the distance of a location from a sensor influences this value of registering the location with that sensor. Let $\mathbf{P}$, denote the probability of detection of an intruder at location $\mathbf{z}$ i.e. $\mathbf{P}(\mathbf{z})$ denotes an intruder's detection probability at position $\mathbf{z} \in \mathbf{R^2}$ ($\mathbf{z}$ is any location in the $\mathbf{xy}$ plane). $\mathbf{P}$ will be defined in terms of the registration probabilities of the sensor i.e. $(\mathbf{p_i})_{\mathbf{i=1}}^{\mathbf{N}}$ and the location of the sensors in the network i.e. $(\mathbf{x_i})_{\mathbf{i=1}}^{\mathbf{N}}$. For the current scenario of our project, for a detection event to occur requires two sensors to register the presence of an intruder. However, the stakeholders may choose to change this criteria and the subsequent implementation is anticipated to hold true for a vast class of alternatives.

Notation	Description
$\mathbf{X} \in \mathbf{R^2}$	Position of the Target being guarded
$\mathbf{x_i} \in \mathbf{R^2}$	Position of a sensor $\mathbf{x_i}$ for $\mathbf{i} = \mathbf{1}$ to $\mathbf{N}$
$\mathbf{p_i} : \mathbf{R^2} \to [\mathbf{0}, \mathbf{1}]$	Probability of registration of an intruder by sensor $\mathbf{i}$.
$\mathbf{P} : \mathbf{R^2} \to [\mathbf{0}, \mathbf{1}]$	Detection probability of the intruder at any position $\mathbf{z}$ in the $\mathbf{xy}$ plane.

Table 1: Mathematical notations used

2.2 The Probability of Detection

In this section of the report, we discuss the rationale and mathematically derive the probability of detection. But before we dive into the mathematical derivations, it is important to discuss certain scenarios, the occurrence of which could lead to a perceived detection event that is actually misleading. These are called false alarms.

2.2.1 False Alarms

A perimeter intrusion detection system, which in our instance is a network of sensors, is often deployed in outdoor settings and is susceptible to environmental and other local circumstances changes.

Therefore, it is critical to consider and describe the environmental conditions under which the sensor network is intended to operate.

Detailed site drawings illustrating entrances, exits, roads, walkways, fences, electricity, and ducting, among other things, should be supplied with the specification to assist potential providers in identifying and assessing the situation in its entirety.

A network of sensors defending a target is likely to encounter changes in its working environment during the day an in night (for example, differences in animal and on-site staff activities) and also throughout the year (for example seasonal weather variations). These variables can have a variety of implications on the functioning of a sensor network. Certain conditions may result in an increase in false alarms, while others may result in a decrease in detection performance.

The below table provides information on the various weather conditions and their impact on different kinds of sensor networks like – **barrier mounted, ground based, free standing** (CPNI 2017).

Condition	**Eg-Value/range**	**Barrier-Mounted**	**Ground-Based**	**Free-Standing**
Sunlight Exposure		UV radiation can brittle and finally fail plastic components, notably cable ties.	There is no discernible impact.	Solar radiation can have an effect on the functioning of some technologies, and quick variations produced (for example, by clouds passing across the sun) create false alarms in sensor network systems. Consideration should be given to the location of sensor networks, particularly PIR and video-based detection systems, in relation to the rising and setting sun.

Wind Speed	up to 65 KM/hour	Vibrations induced by strong wind on the fence and topping are a significant source of false alerts for barrier-mounted sensor networks.	Ground heave produced by wind-driven tree roots might raise concerns. Alarms may be triggered by debris pushed through the field of an operational system.	Alignment of free-standing systems may be impacted. Blowing grass has the potential to drastically alter the temperature patterns seen by PIR sensors. High winds may potentially blow debris through the detecting zone.
Rainfall	up to 25mm/hour	Vibrations in fences and toppings can be caused by heavy rainfall.	Sodden ground is inefficient at transferring pressure, and pooled water is extremely radio frequency absorbent, posing special challenges for active systems.	Extreme rainfall may result in false alarms, a decrease in detection effectiveness, or a reduction in the effective range of detecting zones.

Fog		No effect.	No effect.	Other than active infrared and laser scanners, where fog disperses the beam, fog has no impact. For certain system types, dense fog might result in constant false alarms. CCTV views may be occluded, reducing the efficiency of video-based detection systems and alarm verification.
Snowfall	up to 30 cm/hr	Snow settling against host fences may significantly increase the strain on the fence and alter its dynamics.	Deep snow can help scatter an attacker's pressure. Additionally, it can function as settled water, decreasing the capacity of electro-magnetic radiating devices to detect them.	Snowfall can cause false alerts in beam break systems. On the other hand, the constant ground temperature provided by snow cover minimises the possibility of PIR systems emitting false alarms.

Freezing conditions (ground frost, ice)		Significant ice development can significantly increase the strain on a fence, resulting in false alerts. At extremely low temperatures, relay switches can freeze in place, resulting in continuous alerts. Not likely to be a serious issue in the United Kingdom.	At extremely low temperatures, relay switches can freeze in place, resulting in continuous alerts. Not likely to be a serious issue in the United Kingdom.	Freezing temperatures might result in the formation of ice on the sensors' surfaces, lowering their detection capability or raising their false alarm rate. At extremely low temperatures, relay switches can freeze in place, resulting in continuous alerts. Not likely to be a serious issue in the United Kingdom.
Lightning strikes	Within a radius of 1 KM	Lightning strikes can damage system electronics.	Lightning strikes can damage system electronics.	Lightning strikes can damage system electronics.

Table 2: Impact of weather conditions on sensor networks

The below table provides information on the non weather environmental conditions that needs to be considered on their impact on these sensor networks.

Condition	**Impact**
Pedestrians in close proximity to the perimeter	Microwave or other radiating field devices may identify persons who have access to the perimeter (e.g., a public sidewalk alongside the perimeter fence). Pedestrians may potentially wander into detecting zones that lack a physical barrier. Additionally, pedestrian movement adjacent to a barrier may trigger warnings from the barrier's sensor networks.
Pedestrian or vehicular access that is legitimate	Some zones may need to be turned off at specific times of the day.
Routes for vehicular traffic next to the perimeter	When heavy automobile traffic passes close to a ground-based or barrier-mounted sensor network installation, vibrations might trigger false alerts. Vibrations of extreme magnitude may cause free-standing sensors in the network to drift out of alignment, resulting in false alerts. Passive infrared systems are sensitive to hot things in the distance, such as cars. If they are not properly oriented, they may be activated by the hot exhaust of passing automobiles.
Machinery	As mentioned previously, heavy machinery in the neighbourhood may induce vibrations and trigger false alerts in the sensor network.
Rivers and streams	Microwave devices are very susceptible to moving bodies of water. Consider whether flooding is a possibility, as very few sensor networks are built to be submerged in water.
Coastal	Electrified fence systems are particularly susceptible to short circuits in high saline situations due to the formation of salt deposits. Other considerations to consider for all sensor network types are strong gusts, sea fog, and birds.

Plants and trees	When blown by the wind, trees and plants encroaching on the detection field of free-standing sensors or extremely near to / touching fences (barrier-mounted sensors) might generate false alerts. Additionally, trees may produce fruit and other organic waste that interact with the detecting zone. Wind-driven tree roots can provide unique challenges for ground-based sensor networks, as the movement is perceived as a change in ground pressure. If grass is allowed to grow unchecked, it might trigger false alerts in free-standing sensors throughout the network when blown by the wind.
Subterranean and aerial electricity cables and supplies	Electrical interference caused by power lines, transformers, and other components can impair some sensor networks. Any power lines or supplies located inside or next to the detecting zone should be mentioned in the specification. Electrical shielding may be necessary to avoid false alarms.
Wild animals	The presence of common animals such as rabbits, foxes, dogs, or birds frequently results in false alarms. Systems that are impervious to false alarms caused by a few animals may continue to do so in the presence of vast numbers of creatures.
Drainage issues	Flooding or saturation with water in any portion of the detecting zone can have a substantial influence on the functioning of some devices. For instance, moving bodies of water might trigger false alarms in microwave systems. Moving or stationary bodies of water can result in false alarms or decreased performance in certain ground-based sensor networks, most notably radio frequency radiating field systems. Drainage should always be built and maintained properly.

Table 3: Impact of non weather and environmental conditions on sensor networks

Typically, false alarms are triggered by the weather or wildlife. The rationale for some of these warnings is because the occurrence seems to the detection system to be a genuine attack. For instance, an animal racing over a detection zone might result in a reduction in the received signal of a bistatic microwave system, comparable to the reduction in signal generated by a human breaching the boundary. However, there may be times when there is no evident cause.

Naturally, it is critical for any sensor network defending a target to generate as few false alarms as possible. If there are a significant number of false alarms, additional labour will be produced in analysing and responding to the alerts. This can quickly erode operator faith in the sensor network, resulting in a legitimate alert being missed or disregarded. A system's false alarm rate can be reduced by modifying the sensitivity or other system settings. Typically, the sensitivity would be adjusted to limit the amount of false alarms; nevertheless, this frequently results in a reduction in detection performance. A balance must be struck between the allowed amount of false alarms and the detection performance necessary (Vuong and Chanson 2013).

One may assume that if the sensor is a camera, visual registration on a single device would be enough to detect an intruder. However, if you have a motion sensor, in order to reduce the instances of false positives, sensors may detect an agent's presence only if registrations are completed on at least two sensors.

2.2.2 Detection probability derivation

As discussed earlier, for this project, the criteria given by DSTL is that for a detection event to occur, it requires at least two sensors to register an intruder. Following that, we'll introduce a binary detection variable specified as $\mathbf{k_i}$ significant for all the sensor in the network where a value of **1** signifies that any sensor **i** in the network has registered the intruder while **0** signifies that the intruder is not registered by that sensor.

$$\mathbf{k_i} = \begin{cases} 1, & \textit{if sensor i registers the intruder.} \\ 0, & \textit{otherwise.} \end{cases} \tag{2}$$

Therefore the probability of registering an intruder at the location $\mathbf{z} \in \mathbf{R^2}$ by a sensor $\mathbf{i}$, in the network of sensors, can be written as –

$$(p_i(z))^{k_i}(1 - p_i(z))^{1-k_i}$$

In that case, we may assume that the total probability of detection for an intruder at the location $\mathbf{z} \in \mathbf{R^2}$ by the network of sensors is given by –

$$\mathbf{P(z)} = \sum_{\mathbf{k=1}}^{\mathbf{N}} \prod_{\mathbf{i=1}}^{\mathbf{N}} (\mathbf{p_i(z)})^{\mathbf{k_i}}(\mathbf{1} - \mathbf{p_i(z)})^{\mathbf{1-k_i}} \tag{3}$$

In this equation the summation is a summation over the entire set $\mathbf{k_1}$ to $\mathbf{k_N}$ containing at least two **1s** (i.e. two sensors detecting the intruder's location). If we suppose that only sensors $\mathbf{i_1}, \mathbf{i_2}$ have been able to register an intruder at point $\mathbf{z}$, then the above equation gives us –

$$\mathbf{P(z)} = \mathbf{p_{i_1}(z)p_{i_2}(z)} \prod\nolimits_{\mathbf{j=1}}^{\mathbf{N}} (\mathbf{1} - \mathbf{p_j(z)}), \tag{4}$$

where $\mathbf{j} \neq \mathbf{i_1}, \mathbf{i_2}$. Now if we further assume that the detection probabilities decay fast with respect to their position, we discover that the dominating contributions to $\mathbf{P(z)}$ for suitably well spread sensor network is as below since $\mathbf{min}\{|(\mathbf{z} - \mathbf{x_{i_1}})|, |\mathbf{z} - \mathbf{x_{i_2}}|\} \rightarrow 0$.

$$\mathbf{P(z)} = \mathbf{p_{i_1}(z)p_{i_2}(z)} + \mathbf{0}\bigg(\mathbf{max(p_j(z))}\bigg), \tag{5}$$

where $\mathbf{j} \neq \mathbf{i_1}, \mathbf{i_2}$. As a result, for the sake of simplicity, one might suppose that the contribution to the detection probability is dominated by the two sensors $\mathbf{i_1}$,

$\mathbf{i_2}$ located closest to an agent's position $\mathbf{z}$. Therefore the approximate detection probability at position $\mathbf{z} \in \mathbf{R^2}$ is defined as –

$$\mathbf{P_{approx}(z) = p_{i_1}(z).p_{i_2}(z)}, \tag{6}$$

where, $\mathbf{i_1}$ and $\mathbf{i_2}$ are the two closest sensors to the intruder's position $\mathbf{z}$, i.e. $\mathbf{|x_{i_1} - z|, |x_{i_2} - z| < |x_j - z|}$ for $\mathbf{j = 1}$ to $\mathbf{N}$ such that $\mathbf{j \neq i_1, i_2}$.

If the intruder at location $\mathbf{z}$ is equidistant from more than two sensors, pick at random any two that are closest to $\mathbf{z}$.

The approximation in Equation (6) overestimates the likelihood of detection, because every additional contribution to the product from additional sensors either maintains or decreases the probability. As a result, the approximation does not detract from the challenge's objective. Since for $\mathbf{j = 1}$ to $\mathbf{N}$, $\mathbf{p_j(z) \leq 1}$ and therefore $\mathbf{0 \leq 1 - p_j(z) \leq 1}$, for all $\mathbf{z} \in \mathbf{R^2}$ it follows –

$$\mathbf{0 \leq \prod_{j=1}^{N} (1 - p_j(z)) \leq 1}, \tag{7}$$

where $\mathbf{j \neq i_1, i_2}$. There it is clear that for all locations $\mathbf{z}$ in the $\mathbf{xy}$ plane, it follows -

$$\mathbf{0 < P(z) = p_{i_1}(z).p_{i_2}(z)} \underbrace{\prod_{j=1}^{N} (1 - p_j(z))}_{\mathbf{\leq 1}} \leq \mathbf{p_{i_1}(z).p_{i_2}(z) = P_{approx}(z)}, \tag{8}$$

where $\mathbf{j \neq i_1, i_2}$.

2.2.3 The probability landscape

In this section of the report, we discuss about the generation of the probability landscape for our experiment. In a **2 dimentional 10 X 10** grid, the position of the sensors where fixed in terms of their **x** and **y** coordinate pairs. The position of sensors can be changed at the discretion of the stakeholders. However for the brevity of the situation, a suitable set of **10**, **x** and **y** co-ordinate pairs depicting a sequence of 10 sensors were established at positions – **{(1,1), (4,3), (5,1), (2,7), (4,9), (7,3), (7,9), (9,1), (9,4), (9,7)}**

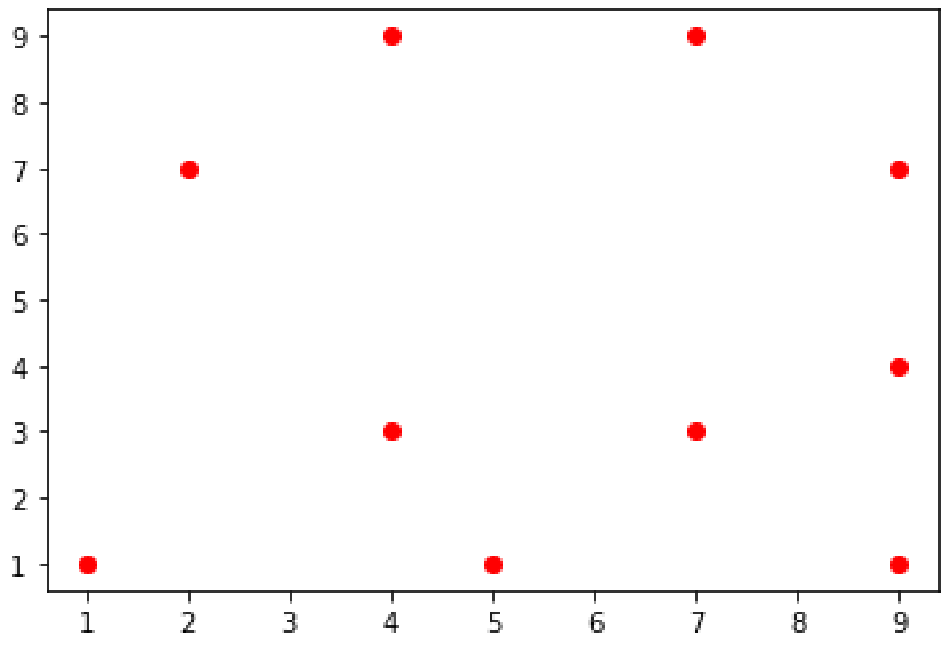


Figure 3: Red dots depicting the senors in a 2D 10x10 grid

We have seen before in equation$(\mathbf{1})$, the probability of getting registered by a sensor. We shall denote this probability of registering by a sensor as $\mathbf{p_i}(\mathbf{z})$. To be precise replacing $\mathbf{P^r_{reg}}$ with $\mathbf{p_i}(\mathbf{z})$ in equation$(\mathbf{1})$. Which gives us -

$$\mathbf{p_i}(\mathbf{z}) = \frac{\mathbf{1}}{\mathbf{1} + (\frac{\mathbf{r_i}}{\sigma})^{\mathbf{2}}}, \mathbf{r_i} = |\mathbf{z} - \mathbf{x_i}| \tag{9}$$

As discussed before, where $\mathbf{r_i}$ is the distance between the location of which we want the determine the probability of getting registered at and a sensor and σ is a constant which denotes the range in units of distance of detection of a sensor.

Now, we also know from the section **2.2.2** above that the approximate probability of detection at a location **z** can be approximately determined by the equation **(6)** which is -

$$\mathbf{P}(\mathbf{z}) = \mathbf{p_{i_1}}(\mathbf{z}).\mathbf{p_{i_2}}(\mathbf{z}), \tag{10}$$

For brevity, we shall drop the subscript **approx** from the equation and shall refer to the probability of detection at any location **z** given by its **x** & **y** co-ordinate pair in the 2 dimensional **10 x 10** grid, simply as **P(z)**.

The distance $\mathbf{r_i}$ between any location **z** given by its **x** & **y** co-ordinate pair and a sensor **i** is determined by the Euclidean method of calculating distance between two points and is calculated by the formula -

$$\mathbf{r_i} = \sqrt{(\mathbf{x} - \mathbf{x_i})^2 + (\mathbf{y} - \mathbf{y_i})^2}, \tag{11}$$

where $\mathbf{x_i}$ & $\mathbf{y_i}$ are the co-ordinate pairs signifying the location of a sensor (**i**) in the grid.

A computer program using the the below logic has been developed in Python programming language, which can generate the probability of detection at all the locations in the entire grid.

Programming logic for probability landscape -

- For a given location, measure the distance between the location in the grid and each sensor placed in the grid. This has been done using the equation **(11)** and a looping mechanism in the programming language which has been used to determine this distance for all possible locations in the grid from each sensor in the network.

- After an iteration of calculating the distance of a particular location from each sensor, store it in an array or list.
- By sorting the array or the list, find the two smallest distances which gives us the two closest sensors to a given location.
- Determine the probability of registering at the location by each of these two closest sensors using the equation **(9)**.
- Determine the probability of detection at the given location by computing the product of these two values, returned from the previous step, as shown in equation **(10)** above.
- Repeat the aforementioned steps in iteration for all possible locations in the grid to generate the probability of detection at each of those locations.

Using the values of probability of detection from the python program mentioned above for all possible locations in the grid, we generate a heat map showing the landscape of probability of detection across the entire grid. Which is shown in the figure below -

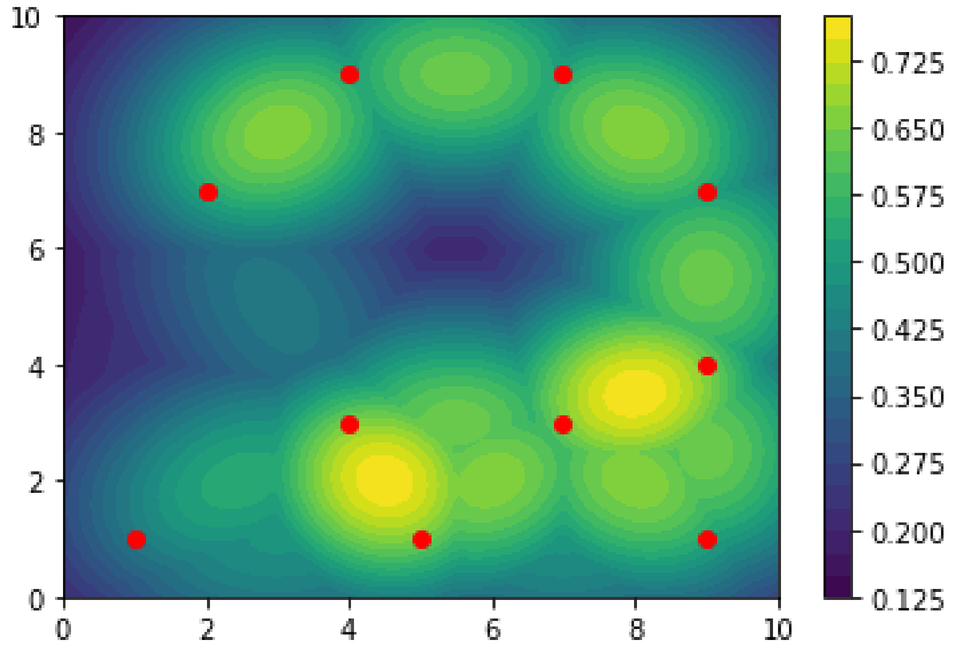


Figure 4: Heat map showing the probability landscape across the grid.

The figure above depicts the probability of detection at different regions in the entire grid with the sensors being positioned. As seen in Figure **(4)**, the contouring goes from dark blue depicting regions of low probability of detection to bright yellow depicting regions of high probability of detection (regions between two sensors close to each other).

2.3 The path

In the early part of the 1600s, Fermat and Descartes created coordinate geometry. They defined coordinates $(\mathbf{x}, \mathbf{y})$ for locations in the plane and established the relationship between a plane curve and an equation in $\mathbf{x}$ and $\mathbf{y}$. This established a link between geometry and algebra (Joyce 2014).

A path can be defined as a function $\mathbf{x} : \mathbf{R} \to \mathbf{R^m}$. It can be thought of as a point moving in $\mathbf{R^m}$. We shall apply the same principle in this project in order to generate a path for the intruder to reach the target.

Here the path is a two dimensional vector, x y as a function of a one parameter. The path here can be defined as $\mathbf{r} : \mathbf{R} \to \mathbf{R}^{\mathbf{2}}$.

Let the total time of travel to a target given by their $\mathbf{x}$ & $\mathbf{y}$ coordinates as $(\mathbf{a}, \mathbf{b})$ be $\mathbf{T}$. Therefore the path can be given by – $\mathbf{x} = \mathbf{a}(\mathbf{T})$, $\mathbf{y} = \mathbf{b}(\mathbf{T})$ which is a continuous path however, in order to discretize it, we can break the total time of arrival to $(\mathbf{a}, \mathbf{b})$, into $\mathbf{N}$ number of small time intervals delta $\mathbf{T}$, given as –

$$\delta \mathbf{t} = \frac{\mathbf{T}}{\mathbf{N}} \tag{12}$$

Therefore a discrete point in the path to the target at $(\mathbf{a}, \mathbf{b})$ can be given by – $\mathbf{x} = \mathbf{a}(\delta \mathbf{t}), \mathbf{y} = \mathbf{b}(\delta \mathbf{t})$. Hence for the $\mathbf{N}$ small time intervals, we can generate a set of $\mathbf{N}$ discrete points with each point away from its previous one by the small time interval $\delta \mathbf{t}$. More elaborately with $\mathbf{t} = \mathbf{0}$ at the starting point and gradually increasing the value of $\mathbf{t}$, in $\mathbf{x} = \mathbf{a}(\mathbf{t})$ & $\mathbf{y} = \mathbf{b}(\mathbf{t})$ by $\delta \mathbf{t}$ for each iteration and for $\mathbf{N}$ iterations until $\mathbf{t} = \mathbf{T}$, what we generate is an approximate path of $\mathbf{N}$ discrete points as $\mathbf{N}$ co-ordinate pairs in the $\mathbf{xy}$ plane. Using this technique we generate an approximate path between our starting point to the target location at $(\mathbf{5}, \mathbf{6})$ comprising of a 100 points with each point varying from its previous point by $\delta \mathbf{t}$.

It is important to note that this path is not the optimal path taking into consideration the cost components i.e. the velocity and the probability of detection. This path is an initial path and has only been generated to make the process compatible with the functioning of the optimisation program used later in the project. The optimisation program used, needs an initial input path as an argument which it then tries to optimise with respect to the cost function. An optimal path would be the one that minimizes the cost. Our goal is to generate a path which minimizes this cost.

2.4 The cost function

A cost function or loss function is a function in mathematical optimization and decision theory that maps an event or the values of one or more variables to a real number intuitively expressing some "cost" associated with the event (Sebastian 2019). The Cost Function quantifies the difference between predicted and expected values and displays it as a single real number. Cost Functions can be constructed in a variety of ways, depending on the nature of the problem. The goal of the Cost Function is to be one of the following (Krzyk 2018):

- **Minimized -** In this case, the returning value is typically referred to as the cost, loss, or mistake. The objective is to determine the model parameter settings for which the Cost Function returns the smallest possible value.
- **Maximized -** In this case, the value it produces is then referred to as a reward. The objective is to identify model parameter settings that generates the largest possible returning number.

In section **1.2** above, we had briefly introduced our cost function. From the discussions in the sections mentioned above it must be evident by now that the primary goal of this project, is to generate an optimal path for the intruder, to reach a given target guarded by a network of sensors, with minimum total probability of detection along the path.

In this section of the thesis, we shall detail out the factors which have been considered to build the cost function. Mathematical implementation of the same, requires assuming a target $\mathbf{X}$ such that $\mathbf{X} \in \mathbf{R^2}$, a starting location for an intruder from where they begin the journey to the target from, denoted by $\mathbf{A} \in \mathbf{R^2}$ and the total time of journey denoted by $\mathbf{T}$. We shall also introduce another parameter $\mathbf{t}, \mathbf{t} \in [\mathbf{0}, \mathbf{T})$ which is the time at any point in the journey.

With this in mind, the goal is to determine an optimal path γ, $\gamma : [\mathbf{0}, \mathbf{T}] \to \mathbf{R^2}$, such that $\gamma(\mathbf{t}) = \mathbf{A}$, at time $\mathbf{t} = \mathbf{0}$ and $\gamma(\mathbf{t}) = \mathbf{X}$, at time $\mathbf{t} = \mathbf{T}$. It is evident that depending upon the stakeholder's choice of the value of $\mathbf{T}$ and the constraint on velocity of the intruder, the resultant path could be unreasonably long for a very large value of $\mathbf{T}$ or too short with high velocity if $\mathbf{T}$ is too small.

Given the objective of this project, our next step is to built the cost function $\mathbf{F}$ taking into consideration, the factors affecting cost of the journey i.e. the velocity of the intruder and the total probability of detection along the path he is taking, in order to be able to generate an optimal path from $\mathbf{A}$ to $\mathbf{X}$.

Cost function build -

Notation	**Description**
$\mathbf{X} \in \mathbf{R^2}$	Position of the Target.
$\mathbf{A} \in \mathbf{R^2}$	Starting position of the intruder.
$\mathbf{T}$	Total time of travel of the intruder from A to X.
$\mathbf{t} \in (\mathbf{0}, \mathbf{T}]$	point in time of the journey from A to X.
$\delta\mathbf{t}$	Is the small time interval $\frac{\mathbf{T}}{\mathbf{N}}$.
$\mathbf{N}$	Number of uniformly long time intervals.
$\mathcal{A}$	Space of all possible paths from A to X.
$\gamma : [\mathbf{0}, \mathbf{T}] \to \mathbf{R^2}$	Is an admissible path from A to X
λ	Coefficient of velocity term

Table 4: Mathematical notations used for cost function

Let us consider a space for all possible paths that the intruder can take from the starting location $\mathbf{A}$ to the target at $\mathbf{X}$ denoted by $\mathcal{A}$.Therefore $\mathcal{A}$ can be defined as -

$$\mathcal{A} := \{\gamma : [\mathbf{0}, \mathbf{T}] \to \mathbf{R^2} | \gamma(\mathbf{0}) = \mathbf{A}, \gamma(\mathbf{T}) = \mathbf{X}, \mathbf{T} > \mathbf{0}\} \tag{13}$$

Therefore, $\gamma \in \mathcal{A}$ denotes an allowed path traversing on which takes a total time for the journey $\mathbf{T}$ where $\mathbf{T} > 0$

The most obvious choice for a cost function that comes to mind, is the cumulative probability of detection along the path γ, with the assumption that detection is also proportional to the amount of time spent at a given position, a minimization of which would yield a path from an initial position to the target that minimises the total probability of detection.

With $\mathbf{t} \in [\mathbf{0}, \mathbf{T})$ already defined as a point in time of the journey along the path, let us consider a small time parameter $\delta\mathbf{t} > \mathbf{0}$. Therefore between time $\mathbf{t}$ and $\mathbf{t} + \delta\mathbf{t}$ the intruder changes position from $\gamma(\mathbf{t})$ to $\gamma(\mathbf{t} + \delta\mathbf{t})$. With the assumption that $\gamma(\mathbf{t})$ is small enough, the probability of detection in moving from position $\gamma(\mathbf{t})$ to $\gamma(\mathbf{t} + \delta\mathbf{t})$ can be approximated to be $\mathbf{P}(\gamma(\mathbf{t}))\delta\mathbf{t}$.

Now by splitting the total time of journey $\mathbf{T}$ i.e. from time $\mathbf{t} = \mathbf{0}$ to $\mathbf{t} = \mathbf{T}$ into $\mathbf{N}$ uniformly long intervals of time, with $\mathbf{t_i} = \frac{\mathbf{iT}}{\mathbf{N}}$ for i = *0,...,N* and choosing the small time interval $\delta\mathbf{t}$ to be $\delta\mathbf{t} := \mathbf{t_{i+1}} - \mathbf{t_i} = \frac{\mathbf{T}}{\mathbf{N}}$, the cumulative probability of detection of the whole journey along the path γ ,can be approximated by summing up all the individual probability of detection along these tiny time steps that makes up the whole journey. Written as below -

$$\sum_{i=1}^{N} P(\gamma(t_i))\delta t \tag{14}$$

To be able to apply the principles of calculus into it, we transform this into its continuous counterpart by making the time steps extremely small by introducing the limit $\mathbf{N} \to \infty$ i.e. as $\mathbf{N}$ tends to infinity, $\mathbf{t_i}$ becomes smaller and smaller. Therefore the cumulative probability of detection of the whole journey converges from the discrete version above in equation **(14)** to its continuous version in the limit as $\mathbf{N} \to \infty$ as below -

$$\sum_{i=1}^{N} P(\gamma(t_i))\delta t \xrightarrow{N\to\infty} \int_0^T P(\gamma(t))dt. \tag{15}$$

Introducing velocity in cost -

Although the continuous counterpart of the cumulative probability of detection in equation (**15**) i.e. $\int_0^T \mathbf{P}(\gamma(\mathbf{t}))\mathbf{dt}$ can be thought of as the cost function, however it is important to note that this definition is not complete. This expression can be minimised to a value of 0 with respect to $\gamma \in \mathcal{A}$ by selecting a path which connects **A** to **X** in a straight line on which the intruder moves with infinite velocity. As mentioned earlier in the introduction above, this is not a practical scenario. Hence we should only take into consideration those solutions that do not break the laws of nature. Therefore we should take into account, in the cost function, the velocity and add a penalty to it if it is too high. Thus, we introduce a velocity term into the cost function which grows as the velocity of the intruder increases there by increasing the cost of high velocity traversals.

We do this by adding a term for the average velocity of travel during the journey made by the intruder i.e. $\int_0^T |\gamma'(\mathbf{t})|^2\mathbf{dt}$ where $\gamma'(\mathbf{t}) = \frac{\mathbf{d}\gamma(\mathbf{t})}{\mathbf{dt}}$. With this the cost function can be defined as -

$$F[\gamma; T] := \int_0^T P(\gamma(t))dt + \lambda \int_0^T |\gamma'(t)|^2 dt \tag{16}$$

The prefactor λ to the velocity term, in the equation has been added in order to facilitate the stakeholders in giving appropriate weight to velocity, as deemed important by them.

Our goal now is to minimise this cost function **F** with respect to the paths $\gamma \in \mathcal{A}$ and time of the journey $T > 0$ for a given value of $\lambda > 0$.

3 Gradient based optimisation

Several numerical approaches have been explored extensively for tackling optimization problems. The gradient descent technique is one of the most straightforward and often used approach in the field of the optimization (Mishra and Ram 2019). This method is globally convergent, but suffers from the slower convergence rate as the iterative point approaches the minimizer. In order to enhance the convergence rate, optimizers apply the Newton approach. This approach is one of the most used methods due to its quadratic convergence. A significant downside of the Newton approach is its slowness or non-convergence for the starting point not being taken close to an optima, and it also needs one to compute the inverse of the Hessian at every iteration, which is fairly costly. The components of the Hessian matrix are created using the classical derivative, which is positive definite at every iteration. In quasi-Newton techniques, instead of computing the real Hessian, an estimate of the Hessian is considered (Mishra, Panda, et al. 2020).

3.1 Quasi-Newton approach - BFGS

The principle of Quasi-Newton approach is that to estimate the inverse of Hessian by some other matrix which should be positive definite so that we may achieve a decent approximation of the Hessian inverse at a particular iteration. This reduces the effort of computation of second derivatives and also avoids the complications related with the loss of positive definiteness. This approximation matrix is updated on every iteration such that as the search direction advances the second-order derivative information increases (Lam 2020).

The **Broyden–Fletcher–Goldfarb–Shanno (BFGS)** technique is the kind of quasi-Newton methods for solving unconstrained nonlinear optimization problems which came into existence from the independent work of Broyden (Broyden 1970), Fletcher (Fletcher 1970), Goldfarb (Goldfarb 1970), and Shanno (Shanno 1970). Since the 1970s the BFGS approach gained enormous popularity and today it is considered as one of the best quasi-Newton methods. The BFGS method is one specific approach for updating the calculation of the inverse Hessian, rather than recalculating it after every iteration. It, or its variations, may be one of the most common Quasi-Newton techniques used for numerical optimization (Nocedal and Wright 2006).

In order to minimise our cost function **F**, we have used the Python optimisation routine from the **'scipy'** package with the **'BFGS'** method and there by derived the optimal path for with the the cost function yields the minimum value. In order to converge more quickly to the solution, this method leverages the gradient of the objective function. If the gradient is not specified by the user, then it is approximated using first-differences. The **'BFGS'** approach often needs fewer function calls than the simplex algorithm even when the gradient must be approximated.

4 Research Outcome & Results

We have discussed in this paper before that the primary goal of our project is that with full knowledge of the sensor network and the knowledge of what the probability of registering with each sensor is, find out the most optimal path for an intruder to take from a starting point to the target.

In order to solve this problem, we first surveyed the landscape with respect to the sensor positions in order to calculate the probability of detection across all the locations in the vicinity of the sensor network which is illustrated by the heat map in figure 4 in section 2.2.3 above.

We then built a cost function as discussed in section 2.4 which generates a cost associated with a path that the intruder could take from a start point to the target. The strategy then implemented in order to derive the most optimal path for the intruder to take is to obtain those values of x & y carving out the path from a fixed starting point to the fixed target for which the cost function yields the minimum value. This for a given time of travel (T) and a given weight to velocity (λ).

4.1 Initial Analysis (optimum path generation)

For-

- Time of travel, $\mathbf{T = 6}$
- Pre-factor of velocity, $\lambda = \mathbf{0.1}$

- Start position in the grid, $\mathbf{A} = (\mathbf{4}, \mathbf{1})$
- Target position in the grid, $\mathbf{X} = (\mathbf{5}, \mathbf{6})$

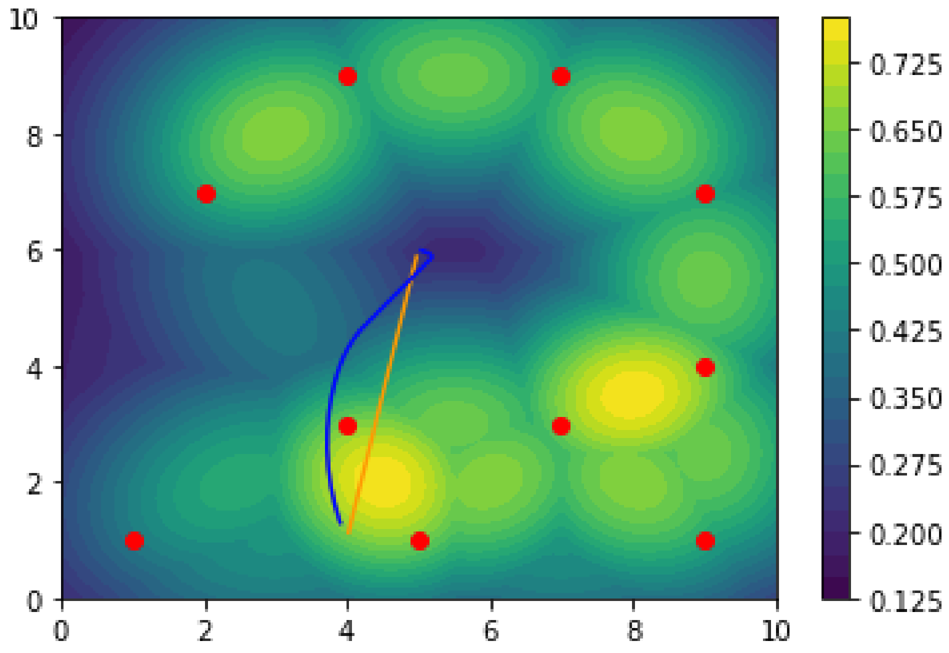


Figure 5: The optimal path solution

In this figure above, the blue line represents the optimal path solution to the problem for the given value of $\mathbf{T}$, λ, the start position $\mathbf{A}$ and the target at $\mathbf{X}$ mentioned in bullets above. The orange straight line on the other hand is just an initial path generated for the optimisation program to use. However, it can also be thought of as a hypothetical path that the intruder can take from the start position to the target.

It should be noted in the observation above that the weight assigned to the velocity of the intruder by its coefficient or pre-factor $\lambda = \mathbf{0}.\mathbf{1}$ is very low. This means that the intruder can take advantage of velocity by moving with high speed and yet incurring a low cost. In this case the the probability of detection term in the cost function $\mathbf{F}[\gamma : \mathbf{T}]$ becomes the dominating contributor to the overall cost. Hence the optimal path solution deviates away from the high probability of detection while approaching the target.

It should also be noted that with a low penalty on speed and the given fixed time of travel, the intruder has reached near the target very quickly. However, due to the constraint on travel time, the intruder travelling at this speed needs to linger in areas of low probability in order to adhere to this constraint. This is evident from the little "hook" like looking curve towards the end of the optimum path of the intruder while getting close to the target.

4.1.1 Stability analysis

It is important to keep in mind that the cost function approximates the cost outcome of the path by first calculating it for its small time steps $\delta\mathbf{t} > \mathbf{0}$ and eventually adding them up for the entire path. By making $\delta\mathbf{t}$ smaller and smaller we can better approximate the cost. This implies that by gradually decreasing $\delta\mathbf{t}$ and generating the optimal path in each iteration, if the optimal path does not change, we know we have reached a point of stability in terms of the small time steps $\delta\mathbf{t}$ and that our optimal path is robust.

In order to implement the stability analysis we gradually increase the value of N in the equation (12), $\delta\mathbf{t} = \frac{\mathbf{T}}{\mathbf{N}}$ for a fixed value of T and thereby making $\delta\mathbf{t}$ smaller and smaller. The results of which are metioned below.

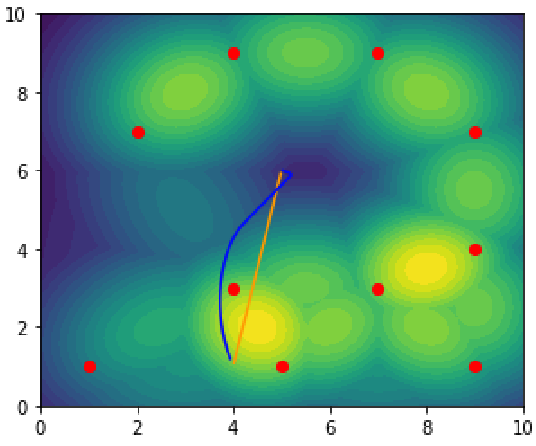


Figure 6: Optimal path, N=60

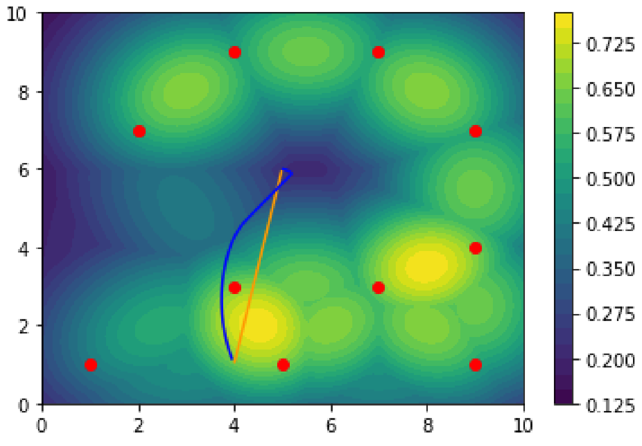


Figure 7: Optimal path, N=80

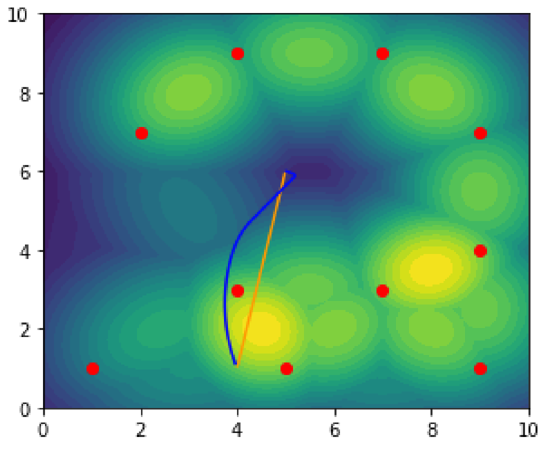


Figure 8: Optimal path, N=100

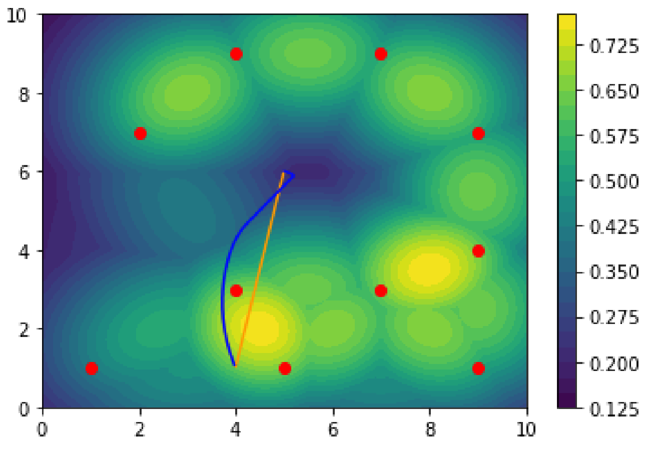


Figure 9: Optimal path, N=150

We can see from the results in the figures above that the optimal path being generated for each increased value of N is similar. The result in figure (5), pertains to N=40. In fact from N=60 as seen in (figure(6)), the curve becomes smooth.

This is an indication that our procedure is stable. We shall continue to use N=100 for subsequent analysis, results of which are mentioned in the below sections.

4.2 Study of time of Travel (T)

Our next objective was to study the impact of - time of travel T on overall cost yielded by the cost function and thereby, the optimal path of the intruder.

With the weight assigned to the velocity fixed to 0.1 as before and with the same starting position and target position, we conducted this experiment on different time of travel (T) values.

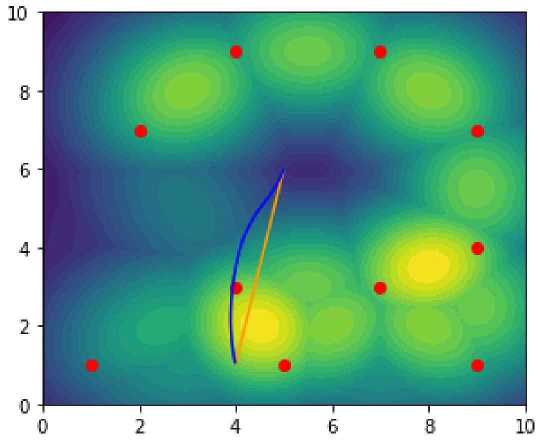


Figure 10: Optimal path, T=3

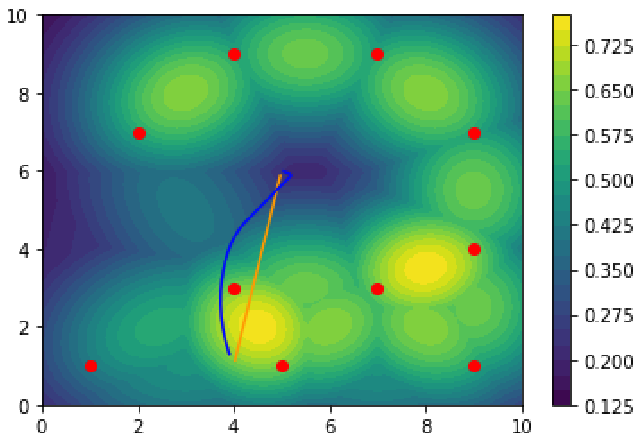


Figure 11: Optimal path, T=6

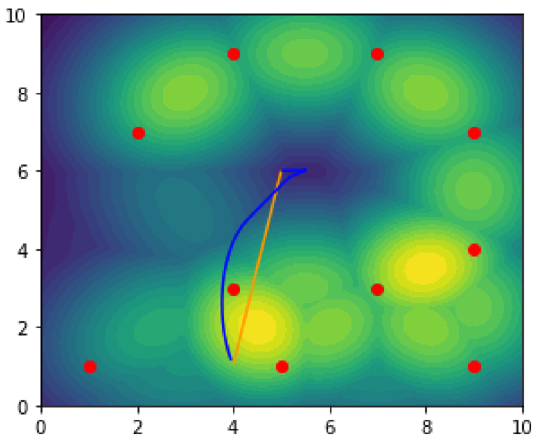


Figure 12: Optimal path, T=9

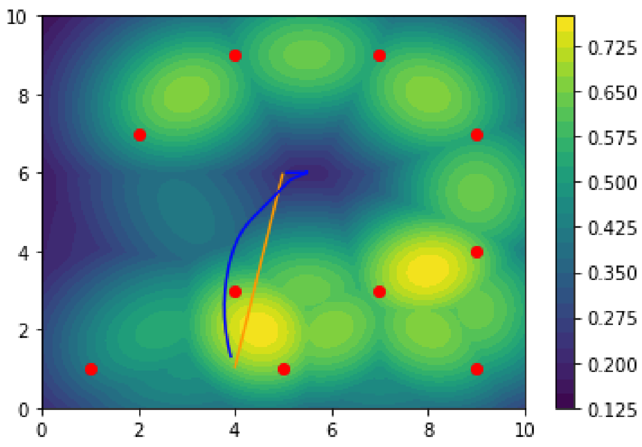


Figure 13: Optimal path, T=15

The figures above show the optimal path solutions at different values of T. From these results, the most contrasting observation can be seen in **Figure 10**, where the time of travel is reduced to half as compared to the previous runs i.e. T=6 **(Figure 11)**. It can be seen in this figure that with a very low time of travel such as T=3 and with the speed that the intruder is travelling, the cost function yields a slightly different optimal path. If carefully noticed, in this path the 'hook' like portion at the end disappears. This happens owing to the fact that there is no more time for the intruder to loose.

Here, the cost function tries to reach a balance between the velocity of the intruder and probability of detection in order to yield an optimal path that grazes past one of the sensors in the network to reach the target within the stipulated amount of time.

It is worth re-emphasizing at this point, that it takes registration by two sensors for a detection event to occur. Owing to that fact, our cost function in this scenario has correctly yielded a path which passes by the lowest probability of detection that it can afford within the boundaries of the constraints pertaining to this scenario.

The results in **Figure 12** and **Figure 13**, where the time of travel is set at higher values, show the cost function yielding longer path. Slightly closer observation shows that the 'hook' region reappears in these results and is in fact larger than what can be seen in **Figure 10**. This reaffirms the fact that the intruder in these two cases is bound to consume the remaining time of the journey and therefore lingers in regions of very low probability of detection towards the end of the path.

4.3 Study of the velocity co-efficient (λ)

Following the study on the time of travel **T** and its impact on the optimum path solution, a sensible idea was to study the pre-factor of the velocity term λ in the cost function. This study, could help us understand the effect on the optimal path solution, brought about by the importance given to the velocity of the intruder by the stake holders.

We already know, from **section 2.4** above that the co-efficient λ of the velocity term in the cost function allows flexibility to the stake holders in assigning a weight to the velocity of the intruder. Meaning higher the value of λ the higher is the penalty on velocity of the intruder and vice-versa.

This implies that with a low value of λ, meaning low penalty on velocity, the intruder gains the liberty to use a vehicle to reach the target thereby taking the advantage of speed. However, for a high value of λ, the intruder would have little choice other than to be on foot in order to reach the target.

With a fixed time of travel, $\mathbf{T} = \mathbf{6}$, the starting position of the intruder at $\mathbf{A} = (\mathbf{4}, \mathbf{1})$, the target fixed at $\mathbf{X} = (\mathbf{5}, \mathbf{6})$, we conducted this study for different values of λ.

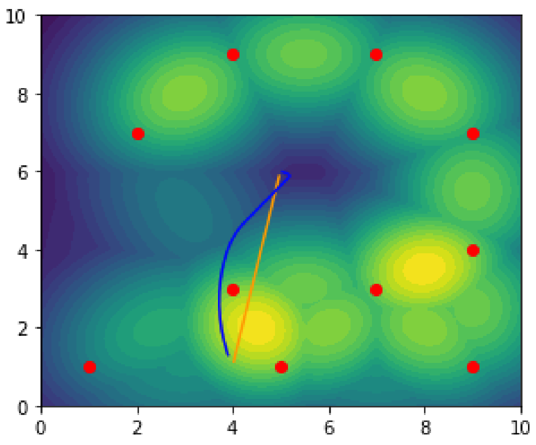


Figure 14: Optimal path, $\lambda = \mathbf{0.1}$

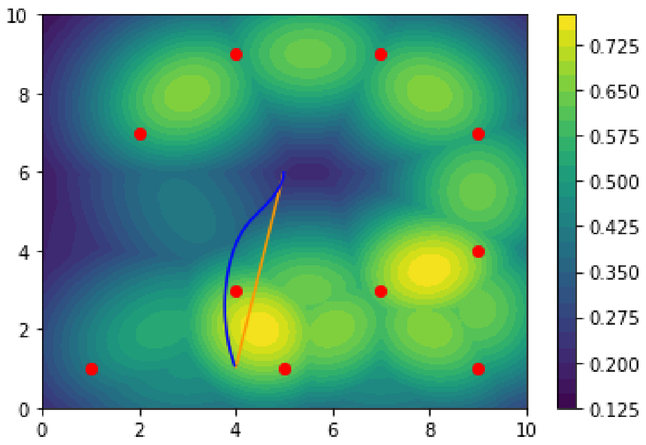


Figure 15: Optimal path, $\lambda = \mathbf{0.25}$

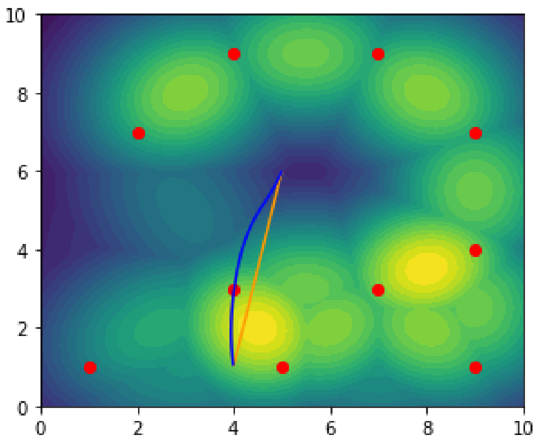


Figure 16: Optimal path, $\lambda = \mathbf{0.5}$

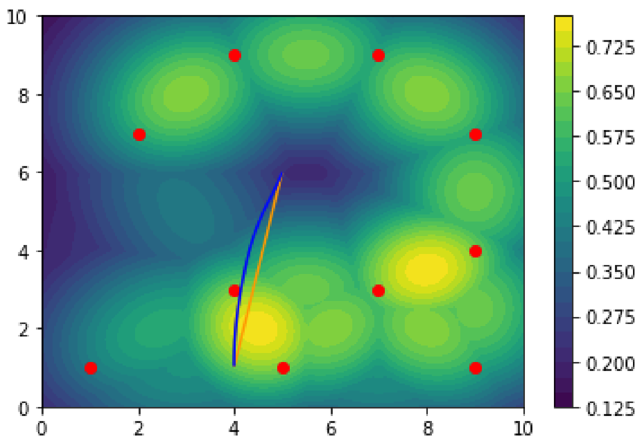


Figure 17: Optimal path, $\lambda = \mathbf{0.75}$

It can be observed from the figures above that the optimal path changes based on the co-efficient of velocity, λ for a fixed time of travel and a fixed start & target position.

For $\lambda = \mathbf{0.1}$ as seen in **Figure 14**, the cost function takes advantage of the velocity term and for the fixed time of arrival T yields an optimal path for the intruder that passes through regions of low probability of detection.

However, as the value of λ increases, the velocity cost goes up. The intruder then cannot take advantage of speed but still has to reach the target within the stipulated time. Therefore the optimal path yielded by the cost function after optimisation, pass through regions with relatively higher probability of detection as seen in **Figures - 15, 16 & 17** but shorter in length.

We also studied the probability of detection of the intruder along these optimal paths for their corresponding values of λ. In order to do so, we feed each of the optimal path generated, back into the cost function again but this time shunning out the velocity component of the cost function. The resultant cost for each path is nothing but the cumulative probability of detection along those paths. Dividing this cost with the total time of travel **T**, gives us what we can call the average probability of detection of the intruder along the optimal path. Which is nothing but a measure of the likely hood of getting detected along the optimal path. Mentioned in the below table are those values.

Velocity coefficient (λ)	**Optimal path - Average probability of detection**
0.1	0.3578
0.25	0.4306
0.5	0.4753
0.75	0.4967

Table 5: Average probability of detection along optimal path

4.4 Multiple intruder scenario

In this part of the research we think of a real life scenario, where a mission requires several intruders to reach the target by a given time T from the start of the mission. The intruders in this case have different start positions where they start their mission to reach the target X from. The starting positions of the intruders are given as p(4,1), q(7,1), r(9,9), s(3,9).

The co-efficient of the velocity term λ is set to 0.1.

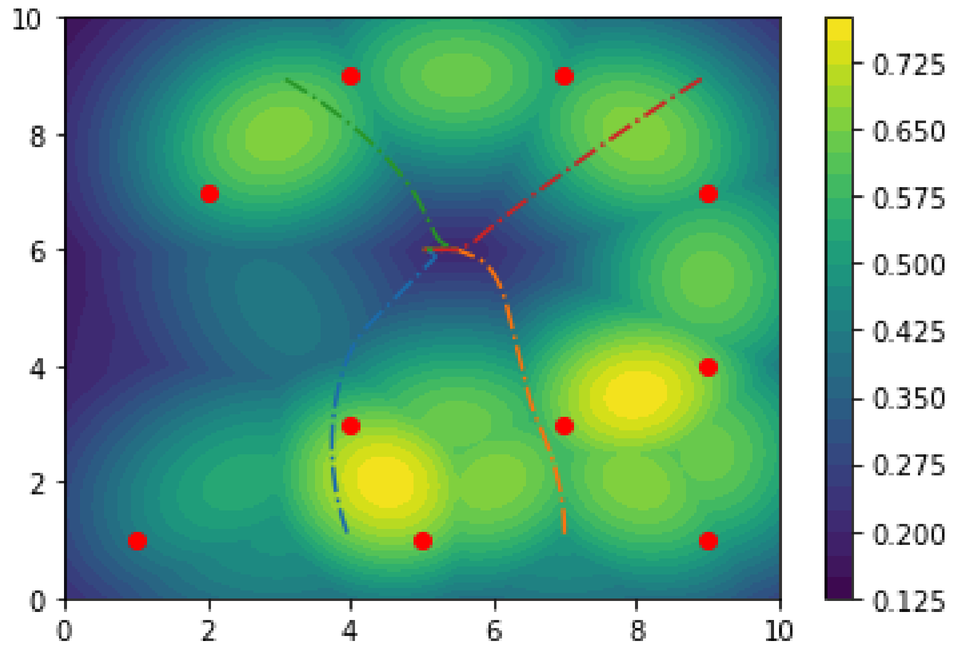


Figure 18: Optimal path solution for multiple intruder scenario

It can be observed from this figure that the trajectory of the optimum path varies depending upon the starting positions of the intruders.

The intruders starting from positions p(4,1) & s(3,9) which are relatively at a closer proximity to the target pass through the regions of low probability of detection and tend to reach the target before time. Which is why they are bound to linger around the target to consume the extra time before they actually reach the target.

On the other hand, the intruders starting from positions q(7,1) & r(9,9) relatively farther from the target, traverse on their optimal path which consumes the mission time **T** more efficiently.

5 Conclusion and future work

In this paper, we have proposed a continuum approach to determine an optimal path for an intruder to reach a target which is guarded by a network of sensors. We have generated a landscape of the density of probability of detection covering area being guarded and produced a visual representation of the same. We have implemented a frequently used Quasi-Newton technique called the 'BFGS' in order to approximate the optimal path.

The continuum modeling approach is by no means a substitute for a discrete modeling approach, but rather can be complementary, especially for the macroscopic modeling of very complex landscapes with barriers and constraints on routes etc.

We have also performed a stability analysis to ensure that the optimal path output is not misleading and hence are able to conclude that the results obtained are reliable.

We have been able to study the velocity coefficient $(\lambda)'s$ impact on the optimal path solution.The measure of probability of detection along the optimal path for the different weights assigned to velocity of the intruder can aide stakeholders in deciding on how much can they leverage the advantage of speed and reach a suitable trade off point.

We have studied time of travel (T) and its impact on the optimal path for different values of it. Although for the scope of this piece of research, we have considered (T) to be constant but it is can be considered as free parameter and a candidate for optimisation in future work. We have also been able to demonstrate a multiple intruder scenario in the probability landscape for a given mission time (T).

6 Acknowledgement

I offer my sincerest gratitude to **James Matthews** from **PA consulting** and **Olly Gage** from **DSTL** for supporting this work and to **Prof. Kurt Langfeld** from **University of Leeds** for his guidance and supervision.

References

Altman, J. (2004). "Acoustic and seismic signals of heavy military vehicles for co-operative verification". In: *Journal of Sound and Vibration* 273, pp. 713–740.

Ammari, Habib M (2009). "Challenges and Opportunities of Connected K-Covered Wireless Sensor Networks". In: *Sensor Deployment to Data Gathering* Germany: Springer Berlin Heidelberg.

Arora, A. et al. (2004). "A line in the sand: a wireless sensor network for target detection, classification and tracking". In: *Journal of Computer Networks.*

Atayev, A. and C. Delaosa (2021). "Escape the Sensor - Mathematical Challenges in the Electromagnetic Environment". In: *University of Cambridge, Newton Gateway to Mathematics.*

Broyden, C. G. (Mar. 1970). "The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations". In: *IMA Journal of Applied Mathematics* 6.1, pp. 76–90. ISSN: 0272-4960. DOI: 10.1093/imamat/6.1.76. eprint: https://academic.oup.com/imamat/article-pdf/6/1/76/2233756/6-1-76.pdf. URL: https://doi.org/10.1093/imamat/6.1.76.

CPNI (2017). *Guide to Perimeter Intrusion Detection Systems (PIDS).* URL: https://www.cpni.gov.uk/resources/perimeter-intrusion-detection-systems-guidance-document (visited on 11/15/2021).

Fletcher, R. (Jan. 1970). "A new approach to variable metric algorithms". In: *The Computer Journal* 13.3, pp. 317–322. ISSN: 0010-4620. DOI: 10.1093/comjnl/13.3.317. eprint: https://academic.oup.com/comjnl/article-pdf/13/3/317/988678/130317.pdf. URL: https://doi.org/10.1093/comjnl/13.3.317.

Goldfarb, D. (1970). "A family of variable-metric methods derived by variational means". English. In: *Math. Comput.* 24, pp. 23–26. ISSN: 0025-5718. DOI: 10.2307/2004873.

Gracia, M.L. (1999). "Security systems design and evaluation. In S.J. Davies RR Minion(Eds.)" In: *Security supervision: Theory and practice of asset protection* 2nd ed.Burlington: Elsevier Science. Pp. 236–254.

Ho, H.W., S.C. Wong, and Becky P.Y. Loo (2006). "Combined distribution and assignment model for a continuum traffic equilibrium problem with multiple user classes". In: *Transportation Research Part B: Methodological* 40.8, pp. 633–650. ISSN: 0191-2615. DOI: https://doi.org/10.1016/j.trb.2005.09.003. URL: https://www.sciencedirect.com/science/article/pii/S0191261505001049.

Jiang, Yanqun et al. (2011). "A dynamic traffic assignment model for a continuum transportation system". In: *Transportation Research Part B: Methodological* 45.2, pp. 343–363. ISSN: 0191-2615. DOI: https://doi.org/10.1016/j.trb.2010.07.003. URL: https://www.sciencedirect.com/science/article/pii/S0191261510000986.

Joyce, D. (2014). *Curves and Paths Math 131 Multivariate Calculus.* URL: http://aleph0.clarku.edu/~djoyce/ma131/curves.pdf (visited on 02/10/2021).

Krzyk, K. (2018). *Linear Regression (Part 2): Cost Function.* URL: https://towardsdatascience.com/coding-deep-learning-for-beginners-linear-regression-part-2-cost-function-49545303d29f (visited on 11/10/2021).

Kumar, Anurag et al. (July 2010). "Wireless Sensor Networks for Human Intruder Detection". In: *Journal of the Indian Institute of Science* 90, pp. 347–380.

Lam, A. (2020). *BFGS in a Nutshell: An Introduction to Quasi-Newton Methods.* URL: https://towardsdatascience.com/bfgs-in-a-nutshell-an-introduction-to-quasi-newton-methods-21b0e13ee504 (visited on 11/10/2021).

Mishra, S.K., G. Panda, et al. (2020). "On q-BFGS algorithm for unconstrained optimization problems". In: *Advances in Difference Equations volume* SpringerOpen.638. DOI: https://doi.org/10.1186/s13662-020-03100-2.

Mishra, S.K. and B. Ram (2019). *Introduction to Unconstrained Optimization with R*. Springer Singapore. ISBN: 978-981-15-0894-3. URL: `https://doi.org/10.1007/978-981-15-0894-3`.

Morris, C. (2003). "Alarm systems fundamentals." In: *In Protection officer training manual* 7th ed.Burlington: Elsevier Science. Pp. 87–92.

Nocedal, J. and S.J. Wright (2006). *Numercial Optimization*. Springer. ISBN: 978-0387303031.

Rong, B. et al. (2018). "Continuum Dynamic Traffic Assignment Model for Autonomous Vehicles in a Polycentric Urban City with Environmental Consideration". In: *Mathematical Problems in Engineering* 8345979. DOI: `https://doi.org/10.1155/2018/8345979`.

Sebastian, R. (2019). "Machine learning and deep learning with python, scikit-learn, and tensorflow 2." In: *Python machine learning* Birmingham: Packt Publishing, Limited, pp. 37–38.

Shanno, David F (1970). "Conditioning of quasi-Newton methods for function minimization". In: *Mathematics of computation* 24.111, pp. 647–656.

Velten, K. (2008). "Introduction for Scientists and Engineers". In: *Mathematical Modelling and Simulation* WILEY-VCH Verlag GmbH Co. KGaA.

Vuong, S.T. and S.T. Chanson (2013). *Protocol Specification, Testing and Verification XIV*. IFIP Advances in Information and Communication Technology. Springer US. ISBN: 9780387348674. URL: `https://books.google.co.uk/books?id=mMnTBwAAQBAJ`.

7 Appendix

Mentioned below are the program codes used for this work.

```
In [1]: import matplotlib.pyplot as plt
        import numpy as np
        import pandas as pd
        import math as mt
```

```
In [2]: def prob_dt(a, b):

            sigma = 3 # assigning the delta value (constant)
            dist = [] # Intitializing the list for distance
            x = float(a) #a[0] # Assigning variable for the x coordinate of agent position
            y = float(b) #a[1] # Assigning variable for the y coordinate of agent position

        # Following loop iterates through the list of sensor positions and returns the list distances of the agent
        # from the each sensor
            for i in s_loc:
                x1 = i[0]
                y1 = i[1]
                k = (x - x1)**2 + (y - y1)**2
                d = mt.sqrt(k)
                dist.append(d)

            dist.sort() # sorting the list
            d1 = dist[0] # assigning the smallest distance
            d2 = dist[1] # assigning the second smallest distance

            p1 = 1/(1+((d1/sigma)**2)) # Calculating probability of registering with the closest sensor
            p2 = 1/(1+((d2/sigma)**2)) # Calculating probability of registering with the second closest sensor
            px = p1*p2 # total probablity of detection by the two closest sensors

            return(px)
```

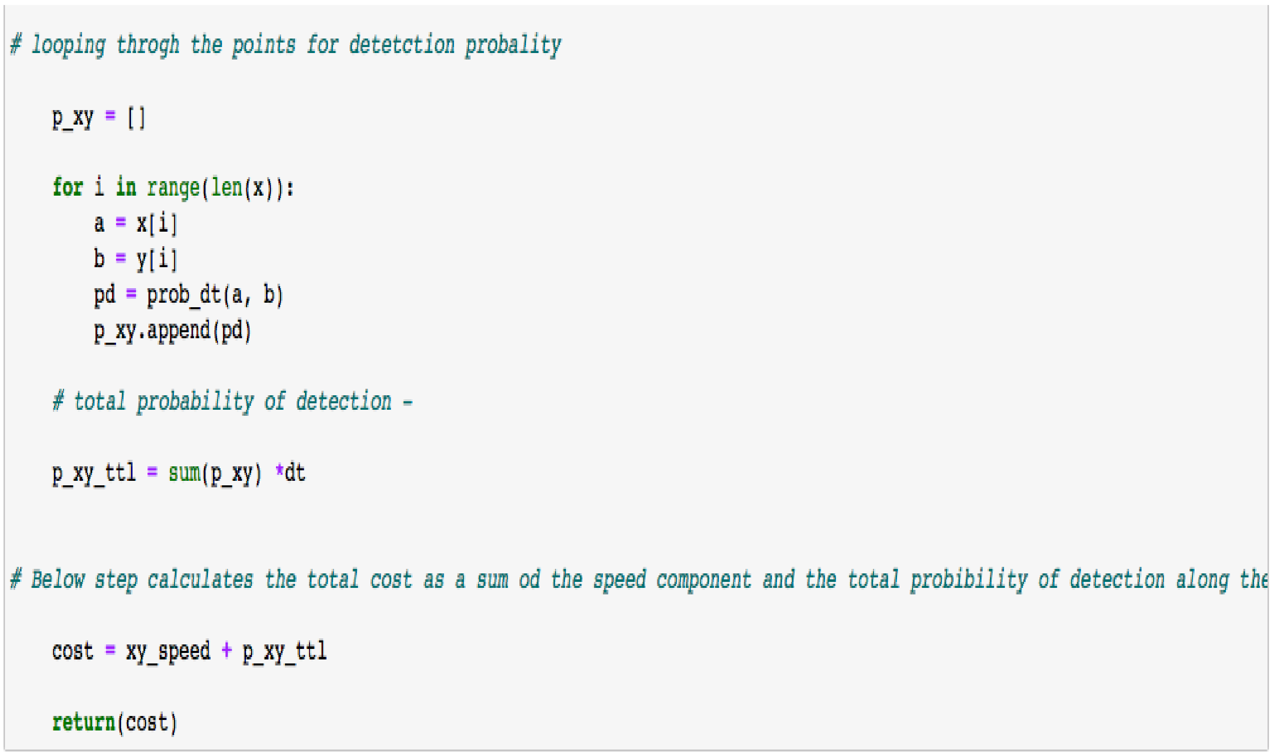
```
In [3]:     # Settinhg the parameters

            s_loc = [[1,1], [4,3], [5,1], [2,7], [4,9], [7,3], [7,9], [9,1], [9,4], [9,7]]
                  # Hardcoding the sensor positions
```

In [4]:

```
# plotting the probality landscape

x = y = np.linspace(0, 10, 100) # Generating the coordinates for a 10 X 10 grid

z = []

for j in y:
    for i in x:
        a = i
        b = j
        pr = prob_dt(a, b) # Generating the probability of detection for all the coordinates in the grid
        z.append(pr)
        k = [a, b, pr]

z = np.array(z)
Z = z.reshape(100, 100) # Reshaping z to fit to the grid
X, Y = np.meshgrid(x, y)

j = []
k = []

for i in s_loc:
    a = i[0]
    b = i[1]
    j.append(a) # assigning the coordinates of the sensors for plotting
    k.append(b)


plt.contourf(X, Y, Z, 30)
plt.colorbar()
plt.plot(j, k, 'ro')
```

In [5]:

```
def cost(z,x_start,y_start,x_end,y_end,dt):

# retrieving number of points on paths
    nn = int(len(z)/2)

    x = []
    y = []

    x.append(x_start)
    y.append(y_start)

    for i in range (nn):
        x.append(z[i])
        y.append(z[i+nn])


    x.append(x_end)
    y.append(y_end)

# looping through the inside points of the loop for tyhe speed

    x_s1 = []
    y_s1 = []

    for i in range(len(x) - 1): # looping through n time intervals from the start point (0,0) to n leaving behind n+1
        kx = ((x[i+1] - x[i])**2)/(dt**2) # calculating the speed over the x coordinates
        x_s1.append(kx)

    x_s2 = sum(x_s1)*dt

    for i in range(len(y) - 1): # looping through n time intervals from the start point (0,0) to n leaving behind n+1
        ky = ((y[i+1] - y[i])**2)/(dt**2) # calculating the speed over the y coordinates
        y_s1.append(ky)

    y_s2 = sum(y_s1)*dt

    xy_speed = 0.1*(x_s2 + y_s2) # Change the prefactor of velocity here
```



```
# looping throgh the points for detetction probality

    p_xy = []

    for i in range(len(x)):
        a = x[i]
        b = y[i]
        pd = prob_dt(a, b)
        p_xy.append(pd)

    # total probability of detection -

    p_xy_ttl = sum(p_xy) *dt


# Below step calculates the total cost as a sum od the speed component and the total probibility of detection along the

    cost = xy_speed + p_xy_ttl

    return(cost)
```

In [6]:
```
## Parameters


n = 80                  # number of time interval, variavles in path 2*(n-1)
t_ariv = 6.           # arrival time
dt = t_ariv /n        # time step

x_start = 4.          # start postion
y_start = 1.0
x_end = 5.            # end positon
y_end = 6.
```

In [7]:
```
# initialise path


z=[]
x0=[]
y0=[]

for i in range(n-1):
    z.append(x_start + (x_end-x_start)/n*(i+1))
    x0.append(x_start + (x_end-x_start)/n*(i+1))

for i in range(n-1):
    z.append(y_start + (y_end-y_start)/n*(i+1))
    y0.append(y_start + (y_end-y_start)/n*(i+1))


# minimise and visualise

from scipy.optimize import minimize

#res = minimize(cost, z, (x_start,y_start,x_end,y_end,dt), method='Nelder-Mead', tol=1e-6)

res = minimize(cost, z, (x_start,y_start,x_end,y_end,dt), method='BFGS', tol=1e-6)

p = (res.x) # assigning the output of minimization to a variable

nn = int(len(p)/2)

x1 = []
y1 = []
for i in range (nn):
    x1.append(p[i])
    y1.append(p[nn+i])
```

```
In [8]: def cost_prob(z,x_start,y_start,x_end,y_end,dt):

        # retrieving number of points on paths
            nn = int(len(z)/2)

            x = []
            y = []

            x.append(x_start)
            y.append(y_start)

            for i in range (nn):
                x.append(z[i])
                y.append(z[i+nn])


            x.append(x_end)
            y.append(y_end)

        # looping throgh the points for detetction probality

            p_xy = []

            for i in range(len(x)):
                a = x[i]
                b = y[i]
                pd = prob_dt(a, b)
                p_xy.append(pd)

            # total probability of detection -

            p_xy_ttl = sum(p_xy) *dt

            return(p_xy_ttl)
```

```
In [86]: n = 40                # number of time interval, variavles in path 2*(n-1)
         t_ariv = 6.         # arrival time
         dt = t_ariv /n       # time step

         x_start = 4.         # start postion
         y_start = 1.0
         x_end = 5.           # end positon
         y_end = 6.



         plt.contourf(X, Y, Z, 30)
         plt.colorbar()
         plt.plot(j, k, 'ro')
         plt.plot(x0, y0, 'orange')
         plt.plot(x1, y1, 'blue')  # With velocity prefactor as 0.75
```