Simulation and Implementation of Distributed Sensor Network for Radiation Detection

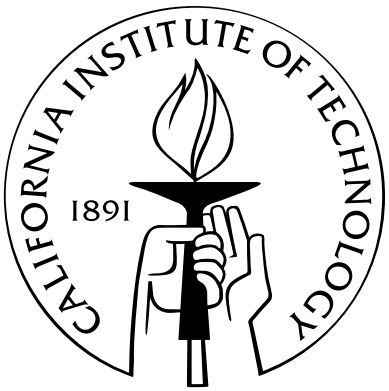
Thesis by

Annie Liu

In Partial Fulfillment of the Requirements

for the Degree of

Master of Science



California Institute of Technology

Pasadena, California

2010

(Submitted)

Acknowledgements

I would like to thank my advisor Prof. K. Mani Chandy for his invaluable supports and encouragement on this work. I would also like to express my special gratitude to Dr. Ryan McLean, Dr. Simon Labov, Dr. Karl Nelson, and my fellow graduate students Matt Wu and Concetta Pilotto for introducing me to such an interesting problem and sharing with me their insights and suggestions. Finally I would like to thank my parents and Benjamin Rossman for giving me the strength and courage to pursue my goals. This work is supported by the National Science Foundation Graduate Fellowship.

Abstract

The problem of monitoring and searching for threats that involve radiological weapons is extremely challenging because of he high variance in background radiation, the presence of benign sources and possible shielding on harmful sources. We present in this thesis a collection of algorithms and analysis that center around the problem of radiation detection with a distributed sensor network. We studied the basic characteristics of a radiation sensor network and focused on the tradeoffs between false positive rate, true positive rate, and time to detect one or more radiation sources in a large area. Three major results came out from this thesis work. First of all, we developed a simulation platform modified from multiplayer game engine that is capable of simulating realistic data in highly dynamic environments. Secondly, we provided mathematical and simulation analyses regarding critical system parameters such the number of sensors, sensor placement, and sensor placement. We also introduced a robust data fusion and parameter estimation method based on Bayesian framework. Lastly, we described an initial work to construct a ground mobile sensor for indoor search and surveillance purposes.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The Radiation Detection Problem

Defending urban regions and other key targets against attacks involving improvised nuclear explosive devices (INDs), stolen nuclear weapons, or radiological dispersal devices (RDDs) is a difficult yet critical challenge for the nation. Large monitoring systems at choke points (e.g. commercial airports and harbors) can prevent the entry or exit of nuclear sources, but it can not protect perimeter that spans a large area, e.g. land and sea borders.

Different scenarios require different counter measures. For example, static radiation sensors deployed on street lamp posts can serve as routine monitoring of traffic on major roads; human agent carrying sensors can be dispatched under the threats of a dirty bomb attack in a political rally. These counter measures all require a large integrated network of detection and interdiction assets that spans several areas, including hardware development, communication system designs, data fusion, detection, and detector allocation algorithms.

In addition to the complexity of such a large system, the ability of efficient detection, localization, and identification depends on a variety of uncontrollable factors such as the presence of benign sources that may cause false alarms, time and space varying background noise, and obstacles that may occlude signal from sources. Under the influence of these factors, a successful system should be able to handle detection of multiple anisotropic and moving sources within a limited amount of time with high true positive and low false confidence. It should then be able to localize the sources for further interdiction, and lastly correctly identify the source type to rule out benign sources. This process, as has been shown in the literature, is very difficult, and not a single algorithm that has been proposed so far produces good results in all scenarios.

Choosing parameters in the system is another challenge in the network design. How many detectors does it require to achieve a certain false positive and true positive confidence in a bounded area? Within a limited budget, should we deploy a large number of inexpensive count-base type of Geiger counters or just a few expensive ones that are capable of distinguishing the energy of each

photon (i.e. energy resolution)? Or maybe a mixture of both? How much does it help if these sensors are able to tell the direction of each incoming photon (i.e. directionality capability)? How many crystals should each detector contain? What is the minimal bandwidth requirement? What is the upper bound on error in sensor locations to achieve a certain system performance? Where should these sensors be placed? How should sensors move if allowed? These questions can be formulated as an optimization problem with the objective to minimize the time to detect, locate, and identify a radiation source subject to the constraints on false positive/true positive rate and cost. This problem, however, is not convex even for the simplest scenarios and may contain multiple minima.

Choice of algorithm also directly impacts key parameters in the system. For example, a simple threshold-based detection algorithm may consider data from each sensor and therefore requires no information transmission (i.e. no data fusion). This greatly simplifies requirements on network communication. In contrast, a sophisticated algorithm that involves data fusion may improve the overall detection performance but at the same time demand extra bandwidth and computation. It is therefore crucial to characterize each algorithm to understand the tradeoffs.

In some scenarios, a mobile sensor network may be the only plausible solution. Mobile sensors, while imposing more demands on wireless communication and bandwidth, can be redeployed to cover a large area with few sensors. Mobility of sensors can either come from human first responders that carry sensors or autonomous robotic devices. Depending on the carrier and the scenario, different redeployment algorithms can be developed to account for the difference in human and robot movement. How to efficiently redeploy the sensors taking different movement models and different terrains into consideration is an important problem when designing a mobile sensor network.

In this thesis work, we answer some of these questions with mathematical analysis, Monte Carlo methods, and data-driven experiments.

1.2 Radiation Physics

Radiation is energy in the form of waves or moving subatomic particles emitted by an atom or other body as it changes from a higher energy state to a lower energy state. There are two types of radiation: ionizing and non-ionizing radiation. Ionizing radiation has enough energy to ionize molecules it encounters whereas non-ionizing radiation does not. There are three major types of ionizing radiation: alpha, beta, and gamma, in increasing order of penetration level as shown in Figure 1.1. All three types of radiations can damage living tissue, however, most detectors are gamma ray detectors because it has the highest probability to be detected at a distance.

The strength of a radiation source is directly related to its half-life, i.e. the time duration when the material remains radioactive. The common units for measuring radioactivity are *curie (Ci)* and *becquerel (Bq)*. One Bq equates one decay per second and $1\ Ci = 3.7 \times 10^{10}\ Bq$, which is roughly

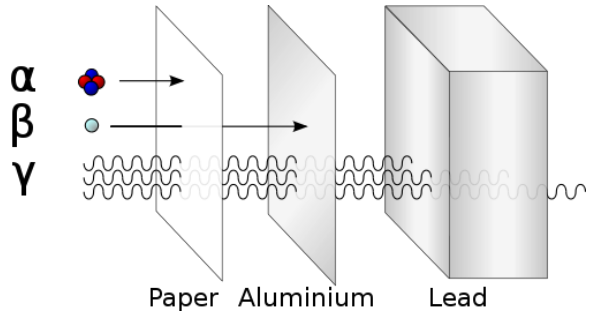


Figure 1.1: Penetration levels of the three types of ionizing radiation [26]

the activity of 1 gram of ^{226}Ra. A radiotherapy machine may have roughly 1000 Ci of a radioisotope, which can produce serious health effects with only a few minutes of exposure.

1.2.1 Health Effects

Radioactive material is composed of elements with unstable nucleus which emits gamma particles, or high energy photons. Due to its ionizing nature, these gamma particles can cause irreversible damage in molecular level in biological organisms if not death immediately.

1.2.2 Common Sources of Radiation

There are three types of benign radiation sources that present in everyday life.

1. Naturally Occurring Radioactive Materials (NORM): These gamma-ray emitting materials are abundant in dirt, air, tiles, walls, pavements, and pretty much everywhere. The most common isotopes are Potassium (K), Thorium (Th), and Radium (Ra).

2. Industrial Material: Radioactive materials have lots of industrial applications. Smoke detectors in every household contains Americium (Am). The attenuation property of radioactive materials also make them ideal for measuring the level of thickness underground.

3. Medical Sources: Radioactive materials are widely used in sterilization and treatment of cancerous cells. ^{137}Cs is one common isotope in this category.

1.3 Detector Physics

The detection of gamma rays can be explained as basic interactions which occur within the detector. There are three processes that a photon can interact with matters by which it can lose partial of all of its energy. These are: 1) the photoelectric effect, 2) Compton scattering by electrons in the atoms of the material, and 3) the production of a positron-electron pair in the electric field of an

atom [9]. It is important to know the basic principles of these three interactions to understand the limitations and complexity of gamma ray detectors.

Photoelectric Effect In the photoelectric process, all of the energy of the incident photon is transferred to the kinetic energy of a bound electron as it is ejected from the atom. A monoenergetic photon will produce a monoenergetic electron within the volume of the detector. If this is the only form of energy loss, the detector can easily decipher these individual photoelectrons.

Compton Scattering In the Compton process, incident photons are scattered by the electrons with partial energy loss to the electrons. This process is shown in Figure 1.2, where λ is the wavelength of the incident photon, and λ' is the wavelength of the scattered photon.

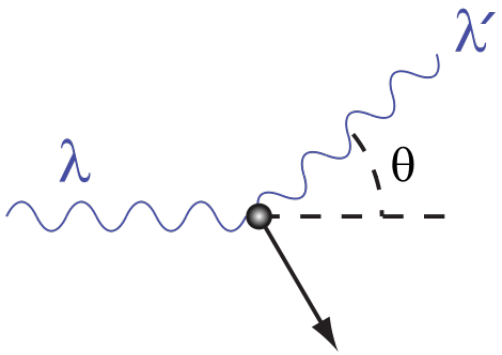


Figure 1.2: Pictorial illustration of Compton Scattering. The energy of the incident photon is partially transferred to the scattered photon [25].

The difference in energy between the incoming and the deflected photon is represented in the Compton scattering equation:

$$\lambda' - \lambda = \frac{h}{m_e c}(1 - \cos\theta) \tag{1.1}$$

where m_e is the mass of the electron. h is the Planck's constant. $\frac{h}{m_e c} = 2.43 \times 10^{-12} m$ is known as the *Compton wavelength*. A monoenergetic source of gamma radiation will produce a wide energy distribution within the detector in the interaction by Compton process.

Pair production Pair production occurs only when the incident photon has energy more than twice the rest-mass energy of positron-electron pair (1.02 MeV) and is not often observed. The process results in the disappearance of a photon and the creation of a electron-positron pair.

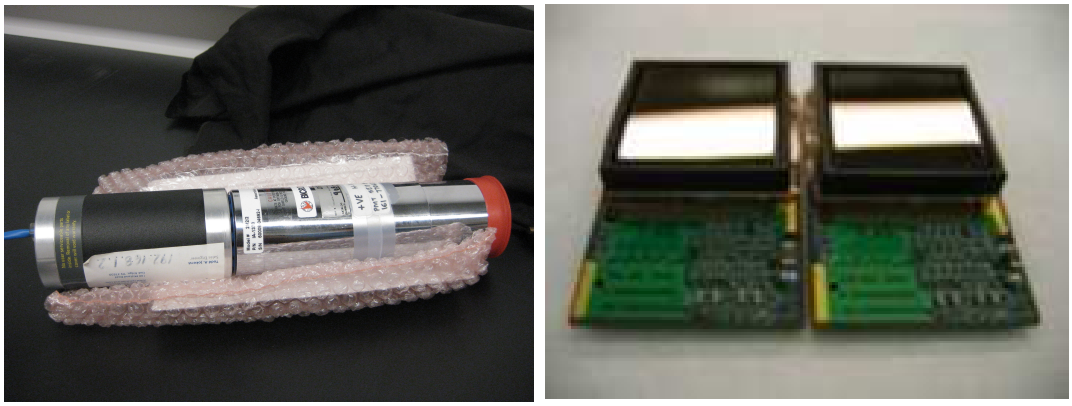
These three processes combined together create the complex electron energy pattern captured by a radiation sensor. Many sophisticated algorithms are developed with the goal to reconstruct the

incident photon streams from these patterns. Depending on the structure of the sensors, some can bin each photon to a certain energy range or even reconstruct the direction of the incident photons. The next section discusses these different types of sensors.

1.3.1 Types of Sensors

Good detector material is capable of stopping and differentiating energy of each photon at contact. The most common and the oldest type of radiation sensor is "scintillator" for its excellent light yield. Scintillator type of sensor interacts with photons through the "scintillation" process. Scintillation is a substance which emits light when struck by an ionizing particle. Modern scintillation detector are typically made of single crystals of sodium iodide (NaI) doped with thallium. Electrons from the ionizing event are trapped into an excited state of the thallium activation center and emit a photon when they decay to the ground state. This signal is then intensified by photomultiplier tubes for low-noise, high-sensitivity detection, as shown in Figure 2.7(a). However, its performance drops exponentially as the temperature goes up [12].

A few other less popular materials are also used in radiation detector, one of which is *Cademinum Zinc Telluride* ("CdZnTe" or "CZT" in short). CZT gamma-ray detector was introduced only in the last decade. It has shown great improvement in both size and energy resolution over other room temperature operated gamma-ray spectrometers. The technology with CZT has also made compact, light-weight detector system possible [**?**] [12]. Figure 2.7(b) shows a model of CZT detector.



(a) NaI scintillator in a photomultiplier tube. The 1 x 1 x 1 cm^3 NaI crystal is inserted at the very tip of the photomultiplier tube, in which the signal is enhanced.

(b) Prototype of a compact CdZnTe detector.

Figure 1.3: Two types of radiation sensors.

1.4 Single Detector Detection Algorithms Overview

Typical techniques in signal processing can not be easily applied to radiation data. For one thing, measurements from radiation detector can be surprisingly noisy. Depending on the time of the day, the surrounding environment, and weather conditions, measurements at the same location may differ up to several factors. For another, the source of interest can be occluded by a wide variation of shielding materials. Lots of efforts have been put into the development of radiation detection algorithm since early 1900. These algorithms mainly focus on detection using one single device and differ a great deal in complexity, which ranges from the simplest gross count rate comparison to application of machine learning techniques, each has its own strengths and weaknesses. This section gives an overview of popular detection algorithms. Knowledge of these algorithms, as well as how they compare, is especially important when dealing with real radiation data.

Gross Counts/kSigma This algorithm declares detection of a source if the received counts per second are above a preset threshold. The threshold is usually set as the k factors of the standard deviation of estimated background counts. This algorithm allows for easy calculation of detection and false positive rates.

Spectra Fitting This algorithm fits the measured spectra to spectral templates from a database of bare and shielded sources and an estimate of the background. A threat value from 0 to 7 is returned from the fitting.

Principle Component Analysis - Anomaly (PCA-A) This method calculates a set of principal components, defined as linear combinations of spectral channels that summarize the largest sources of observed variation in the spectra [27]. Spectra that are not well represented by the major principal components are flagged as anomalies.

Support Vector Machine (SVM) SVM finds a nonlinear hyperplane separator (i.e. the decision boundary) that allows most margins between threat and non-threat labeled data. [2]. It can also be used as radiation source classifier.

Random Forests A random forest classifier is a hierarchical sequence of decisions to classify input vectors into one of many classes. The decision tree is very flexible so that splits at a node may be univariate or multivariate. Random forests generate multiple independent decision trees each of which trains under random samples from the training set.

1.5 Related Works

Portal style radiation detector is limited by its size and imaging resolution. In comparison, distributed sensor network (DSN) is easily deployable, uses less energy per detector, and is more cost effective. Lots of research efforts have gone into evaluation of the potential of DSN in radiation detection. The idea is to replace expensive portal detector with many small, cheap ones, either as static arrangement of sensors that can be placed on street lamps, or as mobile sensors that can be carried around by security personnel or autonomous robotic devices. Early studies have shown that arrays of low resolution detectors sparsely placed on the street give higher detection versus false positive ratio [30]. A network of 11 detectors with wireless and computation capability not only outperforms large portal detector in a tunnel scenario, but also shows improvements in real-time detection [3]. Nemzek *et al* looked into the Signal-to-Noise ratio (SNR) for an array of detectors lining up along a road monitoring a possible source traveling pass them. The results show that the SNR first increases as the square root of the number of detectors then flattens out, and in fact, there's an optimal integration time for any source speed [17]. Using simulation, people have experimented with the optimal number of detectors required in a field to achieve a certain detection rate and confidence [5]. Los Alamos National Laboratory collaborated with University of Mexico to develop a simulation system for evaluating detection capability with off-shelf wireless component and suggested that decentralized computation is important for persistent remote sensing [4].

Source localization and parameter estimation are the two major areas of interests in radiation detection. Groups around the world have proposed different techniques for achieving these two very challenging goals. These groups, many of them in collaboration, include the government agencies such as the Lawrence Livermore National Laboratory (LLNL), Los Alamos National Laboratory (LANL), Oak Ridge National Laboratory (ORNL), Australian Department of defense; academic research facility, including Purdue University, University of Illinois Urbana-Champaign, University of Florida, and the California Institute of Technology; and many other industry partners.

In LLNL, groups have been working on the radiation detection problems for the past few decades. The achievements are mostly in radiation instrumentation, including introduction of new detector materials, enhancement of detector resolution [16], electronic designs, detector software development, and wireless communication improvements. Some resources also went to detection algorithm development, with a focus on improving the detection probability while minimizing the false positive rate, much of it is achieved by carefully separating out background from real sources. This is especially difficult under the influence of NORM and shielding effect [16]. However, these algorithms, as well as all those mentioned in Section 1.4 are all designed for single portal detector. Not until recently does the team start exploring the possibility of multiple mobile detectors. To cope with the complexity in experimenting with mobile devices, the team developed a measurement-based simula-

tion named *Masher*. *Masher* takes a detector model created from processed field measurements and uses Monte Carlo Sampling technique to generate simultaneous measurements for up to 10 virtual detectors at one time. In addition to the common background nuisances present in the model (e.g. ^{40}K, ^{232}Th, ^{32}Rh), target sources (e.g. ^{134}Cs) can also be injected into the simulated data using a simple $1/R^2$ attenuation. This technique, while preserving the real background information, allows for fast evaluation of data fusion and detection algorithm. The team used these simulated data to compare the performance gain of data fusion. Figure 1.4 shows an overlap of background data on top of Google Earth of the LLNL campus as well as its neighboring residential area. The red spots flag the sources present on campus. This data is recorded by a 3 x 3 NaI detector positioned in a car that toured around for a total of 20 hours, spread out in a few days.

Figure 1.4: Measurements are taken on LLNL campus and the residential area nearby. The color represents the count rate received; the count rate increases as the color shifts from green to red. The laboratory houses many radioactive materials, which naturally show up in the data as well. (*Image acquired and used with permission from Simon Labov at LLNL*)

To detect and identify one or a mixture of radioactive sources is not new in the area of radiation

physics, but to localize a shielded (therefore weak) source in an open ungated area is not trivial. The major challenge is to distinguish background from threats, where the background can be easily influenced by temperature, moisture in the air, objects in the environments. All these factors pairing up with mobile detector data makes the problem exponentially hard. Figure 1.5 shows how occlusion can cause benign sources to be seen as threats from a detector inside a moving car. Extra constraints have therefore been introduced to break down the problem to more manageable pieces, and different techniques have been presented to localize a source, from deterministic solutions such as inverse-law inference[6], Maximum Likelihood Estimator [7], or probabilistic solutions, including 2-dimensional least square fitting (LS) [10] [8], Bayesian posterior estimation[5] [28], Extend Kalman Filter and its variants[8]. All these algorithms are shown to perform well in some scenarios; however, many of them break down when background noise can longer be modeled as a simple homogeneous Poisson process. To address this issue, more sophisticated background model (usually from field testing) is introduced. The background at any given time can either be modeled from previous observation in time (temporal) or space (spatial). One approach borrowed from signal processing is Sequential Probability Ratio Test (SPRT). SPRT compares current measurement to previous baseline. If the ratio is below a certain value, it rejects the probability of a source present. If it's above another threshold, it declares a threat found. If it lies anywhere between the two values, the the integrating window is extended and baseline (background model) recalculated. This technique allows for real time/space detection[11] [19] and also allows for decision making on whether a source is present or not [19]. PCA can also be used in separating background from threat. However, with a weak source, there's an increasing chance of filtering out real threats.

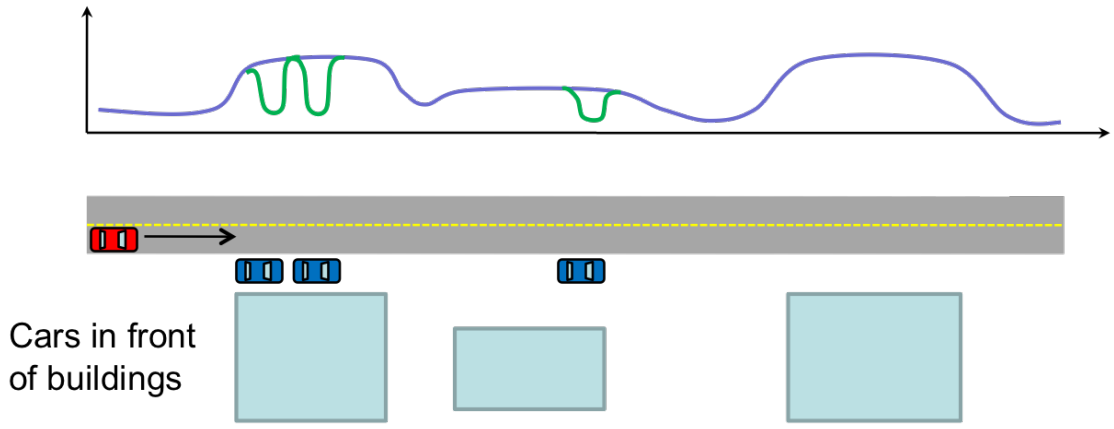


Figure 1.5: The graph indicates the count rate measured along the road. Due to occlusion by the cars parked on the side, benign sources from building materials can look like threat sources and trigger the alarms (*Image acquired and used with permission from Karl Nelson at LLNL*)

As for choice of data fusion techniques, algorithms such as MLE and Bayesian posterior estima-

tion do not require a separate fusion step. For MLE, source localization and parameter estimation is formulated as a multi-dimensional optimization problem that can be solved using state-of-the art search routine built in softwares like MATLAB. However, this method requires extensive computation power and therefore can not be executed in real-time easily unless tight constraints are given. It may also get stuck in local minimum and never reaches the real solution. Bayesian posterior estimation calculates the posterior probability distributions for each parameter (e.g. source intensity, background intensity, anisotropic sensor response, ...etc)using updates from all sensors at each time step. Although it generates real-time estimations it does not scale well as the number of detectors and the surveillance area increase, especially when the parameters space is large [14]. Smarter implementations of Bayesian method such as *Particle Filter* [21] improves the speed by probabilistic sampling the parameter space, but it doesn't solve the growth problem.

Other triangulation type of algorithms require a data fusion step. Rao *et al.* [20] proposed the mean-of-estimator (MoE) method to fuse the estimates from every group of 3 detectors. The fused estimate is the mean of all the estimates. This technique is significantly faster than MLE or Bayesian method, but because of symmetry of detector models, it cannot eliminate "phantom estimates" (i.e. estimates that are created from symmetry). To address this issue, Chin *et al.* [6] present a fusion method by clustering location estimates. The idea came from observation that when noise or statistical/measurement errors are introduced, source position estimates begin to scatter but still cluster around the true position, whereas the phantom candidates will not. They then proposed an iterative pruning clustering (ITP) algorithm that has a worst case time complexity of $O(|C|log_2 A)$. The algorithm first partitions the area to several overlapping regions and gradually eliminates less probable areas until a small region is remained. This technique is shown to be effective with large sensor density on detecting a weak source of 0.911 μCi. It does not outperforms MLE but it cuts down the execution time significantly.

Some of the techniques can also be extended to detection of multiple sources. MLE for multiple hypothesis testing shows to yield good results when two or fewer sources are present, but is not feasible for 3 or more [15]. Bayesian estimation is shown to be very flexible when an unknown number of sources are present [15] [14]. To explore and take advantage of DSN, real time sensor redeployment has also been studied. Existing redeployment algorithms are mostly variations of information gain driven for sequential search of a point radiation source. Ristic, *et al* present a single detector algorithm consists of two steps - estimation and relocation. Source detection and estimation is carried out using *Particle Filter*. The relocation of detector is done by maximizing Fisher Information gain [21]. Simulations using Monte Carlo sampling suggests promising performance from this approach though most constraints are optimistically simplified. This algorithm is further extended to include multiple detectors to search in a area of polygonal shape [22].

Algorithm for radiation detection in DSN can be extended to detection of other hazardous sources

such as chemical and biological hazards though the complexity of the latter problems is much greater. An integrated platform of different types of sensors is also an interest to the defense community [18].

1.6 Chapter Overview

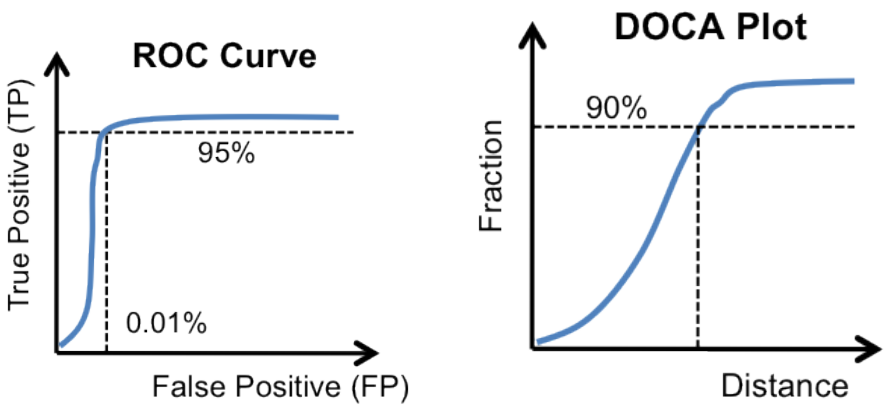
There are a total of six chapters in this thesis. Chapter 1 gives an overview of the state-of-the-art algorithms regarding radiation detection. Chapter 2 describes how we simulate basic interactions between photons and objects, and photons and sensor in a realistic virtual environment. Chapter 3 presents mathematical analysis on sensor data fusion and further quantifies the the gain in terms of time to detect. It also a describes a flexible data fusion and parameter estimation algorithm based on Bayesian framework for detection and localization. Chapter 4 illustrates tradeoffs in key system parameters, such as the density of the sensor network, and placement of sensors, and sensor redirection. Chapter 5 describes the initial work to develop a indoor ground mobile sensor for searching and monitoring radiation sources. A short summary and a list of possible future work is given in Chapter 6.

1.7 Evaluation Tools

In this work, we adopt several well-acknowledged tools to analyze and compare results acquired using different algorithms and different system parameters. These tools allows for objective comparison across several algorithms that operate on various mechanisms. Below are the metrics we use for evaluating performance in detection and localization.

Detection Receiver Operating Characteristic (ROC) curve is widely used to analyze decision making type of problem. ROC curve plots the detection probability (true positive rate) as a function as the false alarm rate (false positive rate). An ideal ROC curve should rise fast to one and stay steady afterwards. Figure is an example ROC curve.

Localization Localization performance is quantified using a Distance of Closest Approach (DOCA) plot, which plots the cumulative fraction of distance between the estimate and the truth. An ideal DOCA curve



(a) Sample ROC curve

(b) Sample DOCA plot

Figure 1.6: Tools for evaluating system performance in detection and localization using different algorithms.

Chapter 2

Measurement Based Simulation

Simulations are necessary to understand the behaviors of the system, the characteristics and limitations of the algorithms, and the impacts on the system from changes in parameters. It requires thorough knowledge in radiation and detector physics to correctly model the data recorded by the detectors. However, no model is perfect. It is very difficult to consider all the detailed interactions based purely on physics laws. Instead, we introduce a hybrid method that combines model based simulation with real measurements. This method, though still imperfect, can capture important sensor signatures that are hard to model otherwise.

2.1 Modeling Radiation Physics

Radiation source in real scenarios is usually anisotropic, which means the effective μ depends on the orientation of the source. A source should also takes volume of complex shape. However, without loss of generality, we only simulate using a simplified model of point, anisotropic source with time invariant intensity in this work.

2.1.1 Photon Detection Model

Radioactive materials emit photons in a Poisson manner. The Poisson distribution is a direct mathematical simplification of the binomial distribution under conditions that the success probability p is small and constant. The photon detection at a radiation detector can therefore be described by the Poisson process and characterized by 2.1, which relates the probability of the detector detecting n photons in time t to detector intensity λ, t, and n.

$$P(n \mid \lambda, \Delta t) = \frac{(\lambda \Delta t)^n \cdot e^{-\lambda \Delta t}}{n!} \tag{2.1}$$

λ is the rate at which the photons strike the detector and is determined by the source intensity (μ), the surface area of the detector (A), the distance from the source to the detector (r) as shown

in Equation 4.2.

$$\lambda = \frac{\mu \cdot A \cdot m}{r^2} \tag{2.2}$$

Where m is a detector specific constant in $[0, 1]$ that represents the effective fractional area. Derivation of k will be covered later and here we assume it's one. Given this property and assume a perfect detector model, it is easy to simulate photon hits at a photon by photon base. Consider a small time interval Δt, the probability of a detector with intensity λ receiving at least one photon is given by:

$$P(n > 0 \mid \lambda, \Delta t) = 1 - P(n = 0 \mid \lambda, \Delta t) = 1 - e^{-\lambda \Delta t} \tag{2.3}$$

We can then compare a random number drawn from an uniform distribution in [0, 1] to this probability to determine if at least photon is registered at the detector. This method allows no more than one hit in Δt, thus Δt needs to be sufficiently small. In fact Δt is the minimum time separation for a detector to distinguish between two consecutive photon strikes or the *dead time* of the detector. Two photons arrive in less than Δt will be registered as one hit. The parameter Δt varies from detector to detector.

Perfect detectors however don't exist. Real sensors, depending on where the crystals are placed and the packaging of the device, have response much less than ideal. To deal with this problem, we adopt real test-bed measurements from the device measuring a source of interest and use the average of the spectra over a long period of integration as input to the simulation. The number of photons generated in each Δt for a detector at a certain energy bracket is a Poisson random variable that can be sampled from a simple algorithm given by Knuth [**?**]. The algorithm is based on the observation that the Poisson process is a sequence of exponential inter-arrival times. One can then count the number of independent draw from a uniform draws before the multiplied value becomes less than $e^{-\lambda_n}$. λ_n is the mean photon counts in the energy bracket n. This algorithm is linear in λ_n.

Algorithm 1 Knuth's cumulative exponential generator method

```
k_n ← 0, L ← e^{-λ_n}, p ← 1
repeat
    k_n ← k_n + 1
    u ← an uniform random number in [0, 1]
    p ← p × u
until p > L
return k_n − 1
```

We can then replace λ with λ_n in Equation 2.1 to generate a new spectrum in each Δt by combining all the k_n for $n = 1, 2, \cdots, N$, where N is the number of energy brackets. Figure 2.1 plots a set of λ_n as a function of energy recorded from a CZT detector. The Poisson process is

memoryless, which means we can conveniently stitch together spectra from different time sequences if needed given that λ remains constant.

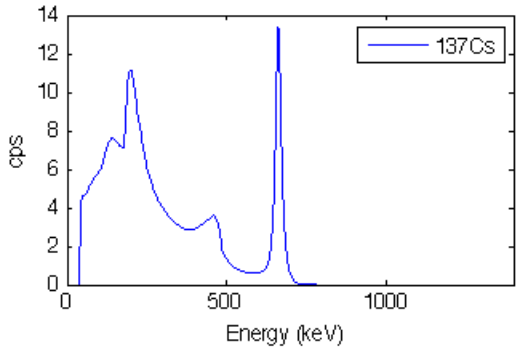


Figure 2.1: ^{137}Cs spectrum recorded using a CZT sensor. The histogram is collected and averaged over a 24-hour period with a sensor placed at 1 meter away from the 1mCi source.

Note that this method suffers when the distance between the source and the detector becomes infinitely small ($d \to 0$) and λ goes to infinity. This problem can be fixed by adding a cap on λ or put a lower bound on d.

2.1.2 Gamma-Ray Attenuation

Photons can be absorbed or scattered upon contact with objects, during which the intensity of the radiation is reduced. This attenuation in energy can be characterized by a fixed probability of occurrence per unit path length in the absorber, also known as *linear attenuation coefficient* (μ).

$$\mu = \tau(photoelectric) + \sigma(Compton) + \kappa(pair) \tag{2.4}$$

A more commonly used term is the *mass attenuation coefficient*, which is defined as $\frac{\mu}{\rho}$ where ρ is the density of the medium. The product ρt is known as the *mass thickness* of the medium, where t is the distance the photons travel inside the material. When the medium is a compound or mixture of elements, the mass attenuation coefficient is calculated as

$$(\frac{\mu}{\rho})_c = \sum_i w_i (\frac{\mu}{\rho})_i \tag{2.5}$$

w_i is the weight fraction of element i in the compound or mixture. The intensity before contact I_0 and the intensity after contact I can be expressed in terms of mass attenuation coefficient and mass thickness.

$$\frac{I}{I_0} = e^{-\mu t} = e^{-(\frac{\mu}{\rho})\rho t} \tag{2.6}$$

Figure 2.2 gives an example of how the *mass attenuation coefficient* affect the amount of photons transmitted through two different materials. Table 2.1 lists a few common absorbers and their density and mass attenuation coefficient for gamma-ray at three different energies.

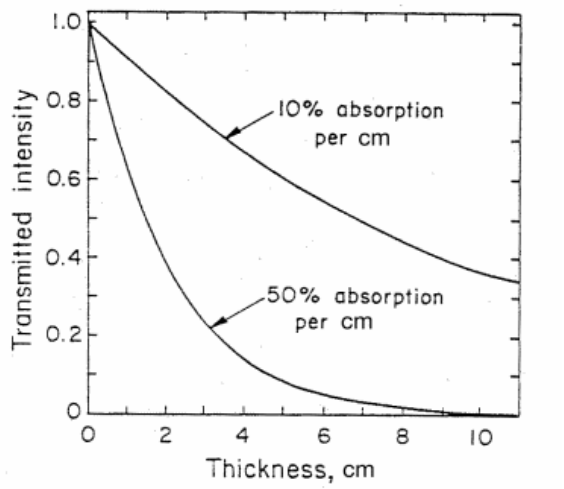


Figure 2.2: The intensity of light transmitted through two absorbing materials with different absorption coefficient [**?**]

Absorber	Density (gm/cm^3)	Mass Attenuation Coeff. (cm^2/gm)		
		100 keV	200 keV	500 keV
Water	1.00	0.167	0.136	0.097
Aluminum	2.70	0.161	0.120	0.084
Iron	7.87	0.033	0.081	0.135
Copper	8.93	0.020	0.059	0.106
Lead	11.34	0.001	0.006	0.088

Table 2.1: Mass attenuation coefficient for a range of materials at gamma-ray energies of 100, 200, and 500 keV.

We simulate the photon attenuation behavior by casting and tracing rays from the center of the detector to the point radiation source. When the rays intersect an object, the information about its material and density is retrieved. By doing this iteratively, multiple objects on the path can be located as well as the length of the path of intersection. The effective detector intensity λ can then be adjusted according to Equation 2.6. This method allows us to simulate the effect of shielding similar to Figure 2.3 where it shows the anisotropic response of a source placed at the back of a SUV.

2.1.3 Modeling Background Radiation

Background radiation is the ionizing radiation emitted from a variety of natural and artificial sources. Primary contributions come from:

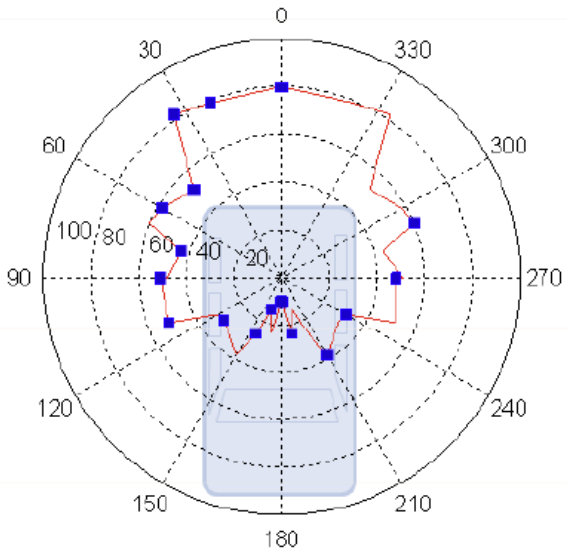


Figure 2.3: Plot of photon count data of a source placed at the back of a SUV.

1. Radiation from the activity of the earth's surface (*terrestrial radiation*), construction materials, food and water incorporated in the body.

2. Cosmic radiations from the space

3. Radiation in the atmosphere.

The contributions from cosmic and atmospheric radiations do not vary much in a small area (e.g. a football field) and can be modeled with a fixed addition to detector intensity. Here we focus on modeling the influence from man-made sources. Ordinary construction materials often contain low concentrations of naturally radioactive elements. The most important components are potassium (^{40}K), thorium (^{232}Th), and uranium (U). Table 2.2 lists the measured activities for some common materials in the environment.

Material	pCi/g		
	Uranium	Thorium	Potassium
Granite	1.7	0.22	32
Sandstone	0.2	0.19	11.2
Cement	5.1	0.57	6.4
Limestone concrete	0.8	0.23	2.4
Sandstone concrete	0.3	0.23	10.4
Brick	3.0	1.2	18.0
Wood	-	-	90
Grass	-	-	-

Table 2.2: Mass attenuation coefficient for a range of materials at gamma-ray energies of 100, 200, and 500 keV.

We can model the influence of background radiation by sampling from real data collected in the environment of interest. This process is no different from simulating contribution from source as described in 2.1.1. It is also possible to dynamically simulate background spectra purely artificially from an environment model. The idea is to assign all objects in the environment a set of parameters that dictates the shape, positions, and material composition. For each detector at any given time, we sample the environment surrounding the detector to build a background profile. This is done by first subdividing the surface area of a unit circle centering at the detector into a fixed number (N) of evenly sized grids. Within each subdivision, we randomly choose a point to cast a ray through. We then trace the ray to locate the first environmental object (e.g. wall, ground, ceiling, ... etc..). After the property of this object is retrieved from the world, we can map its material in Table 2.2 to get the unit area radiation intensity (p_m). This process is illustrated in Figure 2.4. The contribution from this ray in total background intensity is calculated from:

$$\int_x \int_y \frac{p_m \cdot A \cdot \cos\theta}{d_m^2} dxdy = \frac{p_m A \cos\theta}{d_m^2} \times A_m = \frac{p_m A \cos\theta}{d_m^2} \times \frac{4\pi d_m^2}{N} = \frac{4\pi p_m A}{N} \tag{2.7}$$

Where A is the surface area of the detector. A_m is the surface area of grid m on the unit globe. The sum of intensities from each subdivision is the total background radiation intensity observed at a detector at a given location μ_B.

$$\mu_B = \sum_{m=1}^{N} \frac{4\pi p_m A}{N} = \frac{4\pi A}{N} \sum_{m=1}^{N} p_m \tag{2.8}$$

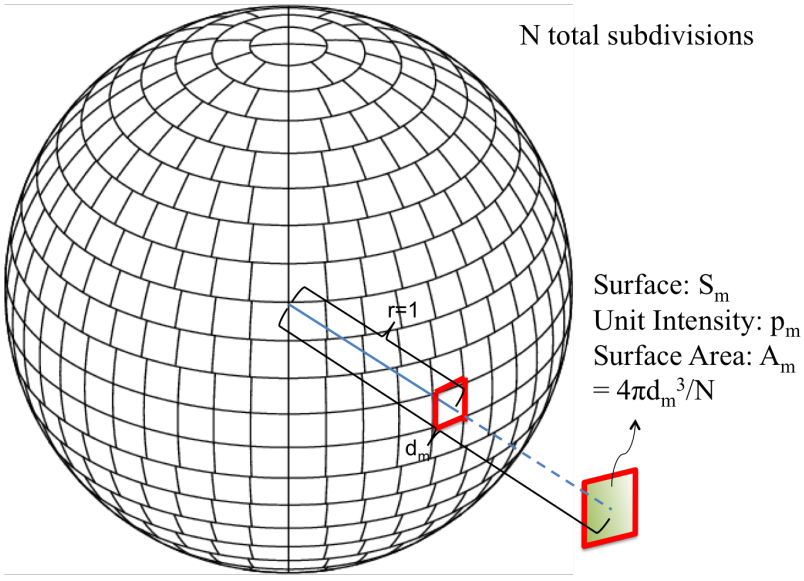


Figure 2.4: N rays are cast and traced to determine the amount of background radiation received at a detector at any given time

By varying the number of subdivision (N), we can control how finely we want to model the influence of background radiation. The indeterministic method for randomizing the direction of the ray within each grid allows high performance with smaller N.

2.2 Modeling Detector Physics

Radiation sensors vary in size, sensitivity, efficiency, resolution, and many other factors. These characteristics are captured by the *response function* of the sensor. A *response function* is a collection of spectra that a sensor should behave in response to monoenergetic sources. The construction of response function requires extensive experimentations on the device. With response functions, we can simulate realistic data for any sensor and source combination. In the absence of it, however, a measurement base simulation as described earlier is as good as we could do.

Depending on the number and the placement of crystals, the same type of sensors (e.g. scintillators) can have very different behaviors. The major differences come from the following two factors: 1) anisotropic response and 2) directionality.

2.2.1 Anisotropic response

Radiation sensor stops and detects photons inside a slab of crystal. In a simplified scenario where we model the crystal as a sphere, the sensor should receive photons with equal probability in all directions. However in reality, depending on the source location relative to the sensor orientation, the percentage of photons that can be detected varies. This results in the "anisotropic sensor response" to photons. This idea best illustrated in Figure 2.6 which shows that different detector models give different response patterns that is $cos(\theta)$ dependent, where θ is the angle between the the normal to the (flat) crystal and the point source. Recall that m is the sensor *effective fractional area* constant in $[0, 1]$. The anisotropic response dictates the value of m. If the crystal were perfectly flat (no depth), we can directly substitute m for $cos(\theta)$ in Equation 4.2 to get the sensor intensity λ. However, the actual response is more spread out because of the crystal depth and various other physical effects. This phenomenon can be approximated using a weighted combination approach of the perfectly flat model ($m = \cos(\theta)$) and spherical model ($m = 1$).

$$\text{sensor effective fractional area} = m = w \cdot \cos(\theta) + (1 - w) \tag{2.9}$$

We can tune the model by adjusting the weight w as shown in Figure 2.5. Note that this is still a less than realistic model. In reality, the sensor response pattern is much noisier.

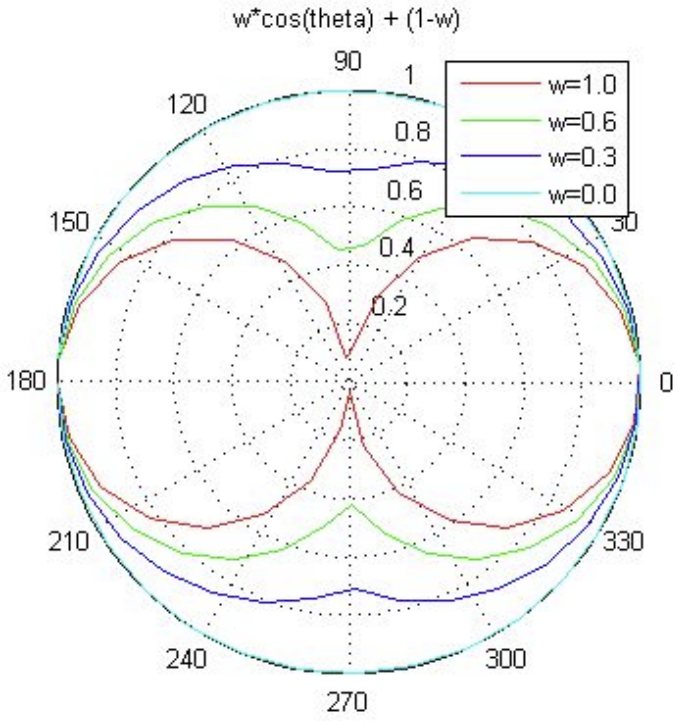


Figure 2.5: Sensor anisotropic response patterns can be approximated using a weighted combination of the two models: *spherical* and *perfect flat panel*. By tuning the weight constant w, we can roughly model the behavior of the sensor.

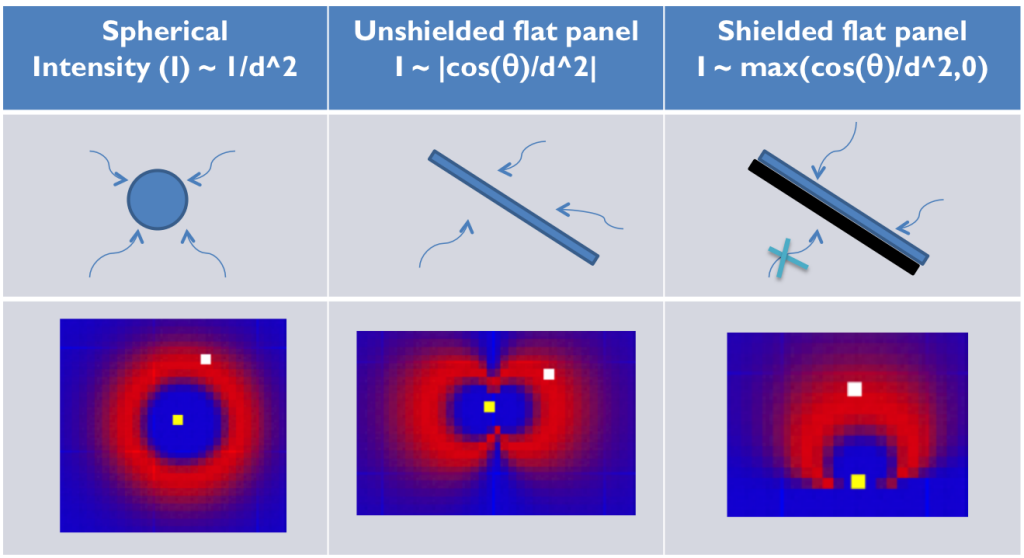


Figure 2.6: Using different assumption of sensor model we observe different response pattern.

2.2.2 Directionality

In addition to anisotropic response, it may be advantageous to implement sensor directionality, which refers to the behavior that the sensor is only sensitive to photons received at a certain range of angles θ. This can be achieved by attaching a lead shield on the back of the flat crystal ($\theta = [0, \pi]$ as shown in the last column in Figure 2.6) or putting the sensor in a well shielded tube. This type of directionality can improve SNR by cutting down noise when the sensor is aimed towards the source. For example, a directional sensor of range $\theta = 2\pi/36$ can detect a source at 30 meters away just as effectively as a source at 5 meters away using a nondirectional sensor ($\theta = 2\pi$) given that SNR increases as $r^2 e^{-\alpha r}/\theta\Gamma$. Directionality also allows for localization with as few as two sensors instead of three as current methods require. We can model directionality in our simulation by specifying a range of effective angles.

2.3 Modeling Realistic Environment In Multiplayer Games

The quality of the models used in the simulation directly affect the results. The better we can model the physical interactions in the world, the more confidence we can have in the simulation results. While simple MATLAB simulation is suffice for the purpose of trade studies, it is difficult to evaluate the system behavior in more complex scenarios that consist different terrains (e.g. grass, water, concrete), miscellaneous obstacles (e.g. building, walls, cars, cargo container), multiple sensors, and intelligent source movement. This section looks into combining the detection problem and virtual environment to create a closer-to-reality simulation. Such a setup gives us insights into the highly dynamical problem and can be potentially adopted as a training tool for government agencies.

2.3.1 Second Life

Second Life is a 3D virtual world developed by the Linden Lab in 2003 and is accessible via the Internet. To join the world, one needs to sign up for a free account and download a client portal program. Users interact with the world's objects as well as other users as personalizable "avatars". Most activities that are common in the real world can be done in the virtual world. For example, users can produce and sell goods, construct buildings, purchase land ownership, chat with other users, participate in meetings, watch television shows, ...etc.. Some of those activities require specific virtual currency - Linden dollar (L$), which can be traded in with real world currency. Most item created int the world can be made interactive by attaching a script that describes the behaviors in Linden Scripting Language (LSL). LSL is a C-like, object-oriented scripting language that has over 300 library functions available. It allows a decent amount of complex interaction patterns. For example, an elevator object inside a building can move vertically in response to buttons pushed.

Because of its ease of access, Second Life is readily adopted as an educational platform by many institutions, such as universities, libraries, museums, and government agencies, hosting events and online resources [**?**]. Organization such as NASA, NIH, JPL also have virtual bases in Second Life that serves as means for making information more transparent to the public in an interactive way. Two examples are given in Figure 2.7.

(a) Ken Davidian from NASA Headquarters speaking at the mixed-reality Next Generation Exploration Conference on NASA CoLab Island in Second Life, and at NASA Ames in Silicon Valley.

(b) The Second Life Obama for President Headquarters.

Figure 2.7: Second Life as means of information circulation.

The virtual world in Second Life is gridded into regions, each of which is either public of personally owned. Diverse terrains and architectures mimicking cities and sites in real world have been built into these regions. The environment together with the avatars wandering in it create a visually rich testbed for simulations that involve dynamic interactions such as detecting a dirty bomb carrying by a suicide bomber in a city. We modeled the physics of a detector detecting photons inside the virtual world. A detector and a radiation object is first created using the 3D graphics tool in the client program, then two scripts encapsulating the algorithm described in 2.1.1 are attached to the objects. Since there is no direct way of getting the source location in the detector object using the API provided in Second Life, the radiation source broadcasts its location periodically. The detector listens on the channels and calculates the probability of getting at least one hit since last update. Figure 2.8 is a screen shot of such a process.

While it requires very little efforts to have a highly complex and dynamic environment in Second Life, it is however hard to carry out data fusion and any extensive calculations. This is because Second Life limits the amount of computation that can be run on its distributed physics engine to avoid crowding. It also lacks sufficient API for modeling photon attenuation and background interference. It is an environment more suitable for experimenting the social aspect of radiation detection.

Figure 2.8: Photon detection simulation in Second Life. The source and detector are labeled. The grey box on the left shows the history of the probability of receiving at least one photon in the time segment of 20 ms when the detector is moving away from the source.

2.3.2 Half-Life 2

Half-Lift 2 is a multi-player first-person shooter game developed by *Valve Corporation*. A full code base of the source engine can be downloaded with each purchase of the game license (less than $30 USD). Developers can freely modify the source to create their own versions of the game that are usually called "mods". There is detailed SDK supports for creating mods as well as extensive community forums. Like *Second Life*, the 3D game environment allows for creation of visually rich complex environment. In addition to the maps that come with the source codes, more maps created by other developers can be found online. Unlike *Second Life* though, *Half-Life 2* has no bandwidth constraints, so we can take full advantages of the physics engine. Also, because that the developers have total control of the source codes, it is much easier to model and simulate any desired behaviors of the sensors. The hardware accelerated ray tracing function also allows for fast re-computation of background influence on the detector side.

We modified the source engine into a multiplayer detection game called *Radination*. Upon entering the game, player can choose to either join the "civilian" group, which plays no roles in the game but to spectate, the "terrorist" group, in which each person carries a radiation source (dirty bomb) trying to evade detection, or the "police" group, in which each person is assigned a radiation detector and is able to deploy mobile detectors mounted on small unmanned aerial vehicle (UAVs). Data from each detector is sent back to a centralized server for processing, and the fused information is displayed as an overview mini map for each "police" player as a "heat map" that marks regions where the source is most likely to be as red and blue otherwise. The 2D probability distribution is computed using Bayesian inference method that will be discussed in a later chapter.

The display gives the user visual input as to which direction to explore next. Figure 2.9 shows a screenshot of what a "police" player will see during the game.

Figure 2.9: Screenshot of Half-Life 2 displaying what a "police" player would see during the game play while holding a detector searching for a radiation source in a flat field. The white and yellow dot son the minimap on the top left corner mark the source and detector position world with respectively.

This setup not only is an excellent testbed for algorithm evaluation it is also a great tool for observing the adversarial response to detection through game plays, from which we may be able to develop better strategies for collaborative searching, detection, and interdiction.

Chapter 3

Sensor Data Fusion

3.1 Why Fuse Data?

Data collected at more than one sensor can either be combined together or considered separately; the former is often referred to as the "data fusion" step. Algorithms without data fusion make decisions based on readings from a single detector only and thus require no sensor-to-sensor or sensor-to-server communication. Algorithms that fuse sensor data on the other hand require the data be wired to one or a few centralized servers for processing, which creates more complexity in system architecture and introduces higher costs. It is therefore very important to understand and quantify the benefits from sensor data fusion when designing a radiation sensor network. For example, if independent analysis is almost as good as algorithms with data fusion, communication between sensors can be simplified.

Data fusion is clearly beneficial for locating and identifying a source, but it is not obvious how much it helps for detection, if at all. Naively combining data (e.g. summing all data together without any filtering) improves signal strength but also adds in noise, which may decrease SNR. The combined SNR may be lower than considering SNR from each sensor individually. To give a dramatic example, consider a grid of widely-spaced sensors and a weak static source placed near one detector. The detector closest to the source will have a high SNR whereas the other sensors receive mostly noise. Combining the data blindly from all four sensors will always decrease the overall SNR. To quantify the benefits from data fusion, we will first introduce some analysis based on Poisson statistics.

3.1.1 Classical Statistics Analysis

We consider the case where the background has been characterized and remains unchanged for the duration of the experiment. The experiment consists of a number of tests for detecting a radiation source in a given region which we assume is homogeneous without structures that absorb photons.

On a given test either no radiation source is introduced into the region, or a shielded radiation source within the region is unshielded when the test starts. The problem is to detect the radiation source, if one is present, with probability at least $bPTP$ (lower bound on the probability of a true positive), and to claim erroneously that a radiation source is present, when one is not present, with probability at most $bPFP$ (upper bound on the probability of a false positive).

We first consider systems that use simple detectors that count the total number of photons but do not measure their energies. The null hypothesis is that no source is present. The alternate hypothesis is that there is at least one source present. The null hypothesis is rejected if and only if some function of the detectors' photon counts exceeds threshold(s); different functions and thresholds are used according to whether decisions are being made with or without data fusion.

Assume that the estimation of whether a source is present or not is made at a given fixed time T. Let $c[j]$ be the random variable which is the number of photons from the background detected at sensor j in time T. $c[j]$ is identically distributed for all sensors j. Let Γ and σ be the mean and standard deviation of this random variable.

We begin with an analysis of a situation where the field is a square with a sensor at each corner. We compare the case where each of the four sensors makes independent decisions with the case where the data from the four sensors is fused.

No fusion: The null hypothesis is rejected if the photon count for any sensor exceeds $\Gamma + K\sigma$ where K is a constant determined by the bounds, $bPTP$ and $bPFP$, on the probabilities of true positives and false positives.

With data fusion: The photon count data from each of the sensors can be fused in several ways. We explore a simple way which builds upon the idea used for the no-fusion case. The null hypothesis is rejected if and only if any of the following conditions described hold:

1. The photon count for any sensor exceeds $\Gamma + K_1\sigma$, where K_1 is a constant. This corresponds to the no-fusion case except that the threshold K_1 is different from (and is larger than) K. If the source is near a corner of the square then the photon count for the sensor at that corner is likely to exceed this threshold. If, however, the source is far away from every corner then this threshold is unlikely to be exceeded.

2. We compute the average of the counts of each pair of neighboring sensors - there are four such pairs. If the average of any of the pairs exceeds $\Gamma + K_2\sigma$ then reject the null hypothesis. The reason for fusing data from adjacent pairs of sensors is to deal with the possibility of a source being along an edge of the square, near the middle of the edge. For example, if the length of the square is 20 meters, and the source is midway along an edge of the square then r^2 is 100 for the two nearest sensors and is 500 for the two farthest sensors, where r is the distance between the source and sensor.

3. Compute the average of the counts from all four sensors and reject the null hypothesis if the count exceeds $\Gamma + K_3\sigma$ where K_3 is a constant. This threshold deals with the possibility of a source near the center of the square.

The values of K_1, K_2 and K_3 are determined to satisfy the given bounds on the true positive and false positive probabilities. Note that by setting K_1 to K and K_2 and K_3 to infinity, the data fusion algorithm becomes the same as the independent algorithm. Therefore, we can ensure that the fusion algorithm is at least as good as the independent algorithm; we will, however, set the values of K_i so that the fusion algorithm performs better than the independent algorithm.

3.1.2 Analytical Results

We derive formulae for the case of two detectors and use Monte Carlo methods for the case of four detectors. First consider two detectors a distance R apart. Let G_μ and F_μ be the cumulative distribution and probability mass functions for a Poisson random variable Y with parameter μ; thus $G_\mu(n)$ is the probability of $Y \leq n$, and $F_\mu(n)$ is the probability of $Y = n$. Let Γ be the expected count of photons from the background at a sensor in time T. The decision strategy is to declare that a source is present if and only if:

1. When data is not fused: the count of any sensor exceeds a threshold Q. The threshold Q is $\Gamma + K\sigma$ (rounding off to integer values).

2. When data is fused: the count of any sensor exceeds a threshold Q_1 or the sum of the counts of the two sensors exceeds a threshold Q_2 (or equivalently, the average of the two values exceeds $Q_2/2$).

Probability of declaring that a source is present: The number of photons received by a sensor in time T is a Poisson random variable with parameter μ where $\mu = \Gamma T$ when there is no source present because the sensor only records background photons which arrive at rate Γ, and $\mu = (\Gamma + \lambda)T$ when photons from radiation sources arrive at rate λ in addition to the background rate of Γ. The value of λ is determined by the location of the source and its strength.

Let $ps[\mu]$ be the probability that an algorithm declares that a source is present when the count at each sensor is a Poisson random variable with parameter μ. Since a false positive is a declaration that a source is present when there is only background radiation, and a true positive is a declaration that a source is present when there is both background and source radiation, we have the following relationship:

$$(PFP = ps[\Gamma]) \ \wedge \ (PTP = ps[\Gamma + \lambda])$$

No fusion case: In this case a source is not declared when $c[1]$ and $c[2]$ are both less than Q. Therefore:

$$ps[\mu] = 1 - G_\mu(Q)^2$$

A ROC (Receiver Operating Characteristic) curve plots $ps[\Gamma]$ on the x-axis and $ps[\Gamma + \lambda]$ on the y-axis for different values of Q and a given fixed λ.

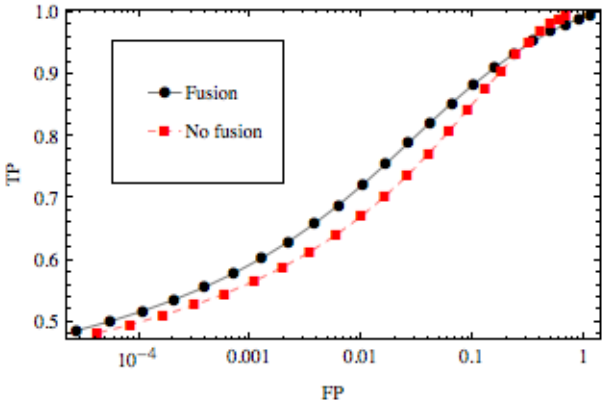
Fusion case: In this case the algorithm claims a source is present when $c[1]$ or $c[2]$ exceeds Q_1 or their sum exceeds $Q2$. Therefore, the algorithm declares the presence of a source when (a) $c[1]$ is greater than or equal to Q_1 , or (b) $c[1]$ is less than Q_1 and $c[2]$ is greater than either $Q[2] - c[1]$ or Q_1. Since the two disjuncts are mutually exclusive their probabilities are summed:

$$ps[\mu] = (1 - G_\mu(Q_1)) + \int_0^{Q_1} F_\mu(c[1]) \cdot (1 - G_\mu(Min\{Q_2 - c[1], Q_1\}))\, \mathrm{d}c[1] \tag{3.1}$$

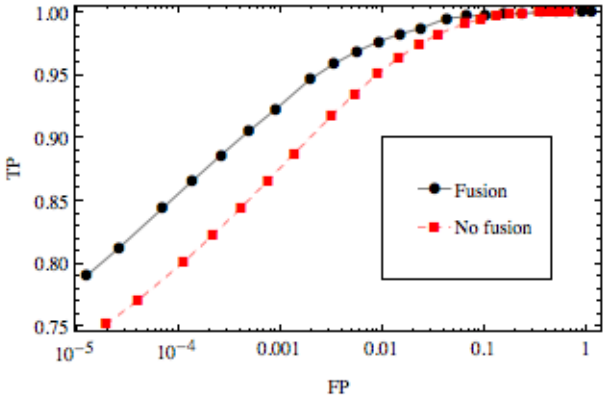
Figure 3.5 shows the ROC curves for the independent and fusion algorithms for T = 3, 15 and 50 seconds. The values of the thresholds, Q_1 and Q_2 could be optimized for each value of PFP and PTP; however, in these plots, the same values of Q_1 and Q_2 were used throughout a plot.The value of λ used in these plots corresponds to a 1 millicurie point source of Cesium placed along the line joining two sensors 20 meters apart, where the location of the source is equally likely to be any points on the line between 1 and 19 meters. Absorption in the air is ignored in this calculation. The graphs show that the fusion algorithm provides better results. For $T = 3$, the fusion algorithm behaves slightly worse than the independent algorithm for extremely high false probabilities; this is because the values of Q_i were not optimized for that case.

The analytic results suggest that fusion does help, but not by much. Why? Fusion helps because the values of Q_1 and Q_2 can be adjusted so that the fusion algorithm is the same as the independent algorithm for some values of these parameters and better for other values — fusion provides more degrees of freedom. The following example illustrates why fusion doesn't help much. When a source is within 6 meters of a sensor (and therefore more than 14 meters from the other sensor), the ratio of the counts of the two sensors is greater than $(14/6)^2 = 5.44$; so the distant sensor adds relatively little value. Thus for at least 12 of the 20 meters, data fusion helps but not by much. Moreover, the single sensor threshold, Q_1 has to be larger in the fusion algorithm than in the independent algorithm to ensure that the false alarm rates don't increase; thus, for some identical counts of the nearer sensor, the independent algorithm will detect the source whereas the fusion algorithm will not.

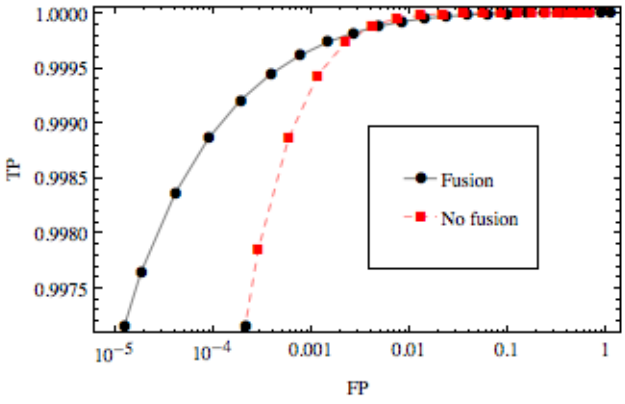
We can measure the performance gain in terms of the amount of time. How much time do we save to reach a desired PTP and PFP by using fusion? Figure 3.2 shows that it takes $\approx 19, 23$ seconds to reach $PTP = 0.99$ for fusion and no fusion, respectively. In other words, the gain from fusion is 4 seconds.



(a) $T = 3$



(b) $T = 15$



(c) $T = 50$

Figure 3.1: ROC curves of K-Sigma method with and without data fusion. $\Gamma = 8T$, $\Lambda = 200T$, $R = 20$. The y-axis is on log scale.

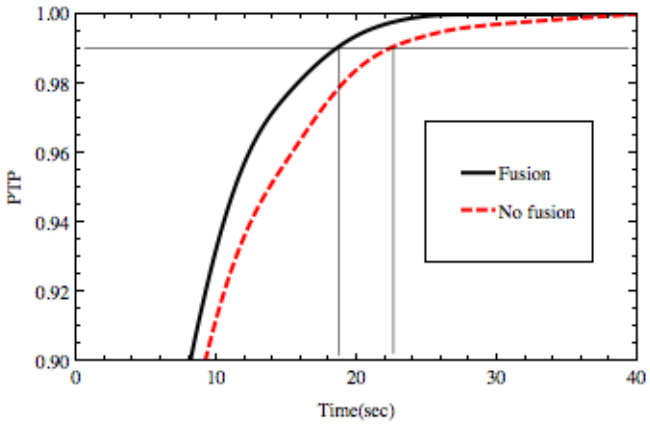


Figure 3.2: PTP as a function of time while fixing $PFP = 0.01$ using the best Q_1 and Q_2

3.1.3 Simulation Results

Next we present the case where four sensors are arranged in a square and a radiation source is placed within that square. Figure 3.3 shows the detection capability of the four sensor ensemble, with and without fusion, as a function of the source position. The vertical axis is the logarithm of the detection variable K; the higher the value the better. The left plot is for the independent algorithm, the right plot is for the fusion algorithm. The points at which it is most difficult to detect a source are those along the edges and the middle of the square, and the plots show that fusion does help. The same arguments as for the 2-sensor case show that though fusion does help it doesn't help a great deal. In simulations, the time to detection using fusion improved by approximately 10%.

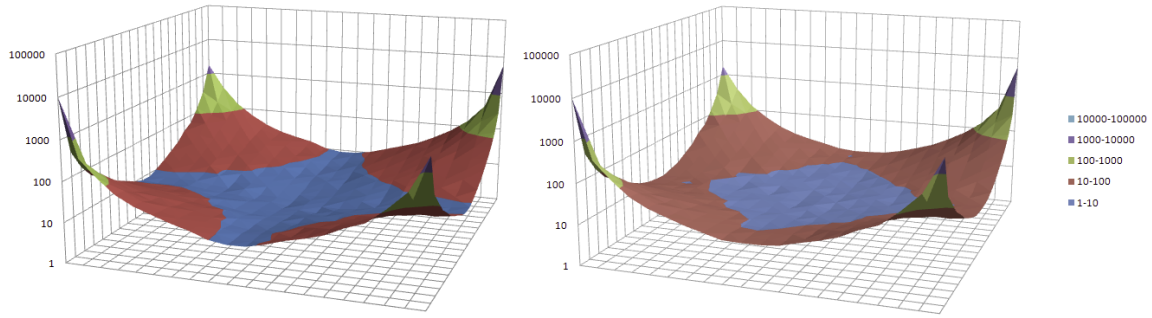


Figure 3.3: Source detection capability for four sensors, without (left) and with fusion.

3.2 Bayesian Method For Parameter Estimation

The detection and localization of radiation source(s) can also be formulated as a parameter estimation problem. Let θ be the vector that contains all parameters we want to estimate, e.g. source presence, source intensity, background intensity, source position, source type, ...etc.. By observing

the posterior PDF $\pi(\theta)$, we can infer the probability that a source is present, the probability distribution of source location as well as source and background intensities. Assuming a prior PDF π_0 for θ, the posterior PDF of π is:

$$\pi(\theta) = L(\theta; y)\pi_0(\theta) \tag{3.2}$$

where L is the likelihood of observing $y_1, y_2, ..., y_m$ photons at detector $j = 1, 2, ..., m$ given the vector of parameters θ. Since each sensor makes independent measurement, L is found by the detector measurement equation as:

$$L(\theta; y) = \prod_{j=1}^{m} f(y_j; \Lambda_j(\theta)) \tag{3.3}$$

$f(y; \Lambda(\theta)) = \Lambda^y e^{-\Lambda}/y!$ is the Poisson statistics function. Note that Λ is the expected sensor intensity that is θ dependent. L can be further modified to account for spectrum data. Assuming independence between energy channels, f can be rewritten as:

$$f(y; \Lambda(\theta)) = \prod_{E=E_{min}}^{E_{max}} f(y_j(E); \lambda_j(\theta)) \tag{3.4}$$

where $y(E)$ is the number of photons received in a particular energy bin. Note that the assumption of independence between energy channels is not completely correct due to Compton Scattering and various other interactions between photons and sensor crystals. However since the spectrum data is simulated from real measurements, the discrepancy is minimized.

Since the posterior PDF of the Bayes method has no closed-form solution, iterative approximation procedure is necessary. In each iteration, $\pi(\theta)$ is updated using (3.2). The iteration time step size t is determined by the sensor live time and bounded by the network capacity. In general, the longer the system observe, the more accurate the posterior PDF resemble the truth. Below we give the details of how detection and localization is achieved using the Bayesian method. This algorithm assumes only two hypothesis - there is no source present or there is exactly one source present. The hypothesis space can be expanded to accommodate detection of more sources.

3.2.1 Localization

We partition the search area into finite grid. Let K denote the total number of grids and p_k^t the probability that a source is present in grid k. At $T = 0$, p_k^0 is determined by the prior distribution we specify. At each time step, p_k^t is updated according to Equation 3.2. Note that both expected sensor intensity Λ combines contributions from both source and background. Therefore $\Lambda = \lambda + \Gamma$. Both λ and Γ are location dependent. λ_{jk} is calculated using Equation 4.2 by replacing r with the

distance between the sensor j and grid k. Rewriting Equation 3.2, we have

$$p_k^t = \prod_{j=1}^{J} f(y_j^t | (\Lambda_{jk} = \lambda_{jk} + \Gamma_k), t) \cdot p_k^{t-1} \tag{3.5}$$

p_k is then normalized. Grid k that has the highest p_k^t is the location estimate of the source at time t. We display p_k as a "heatmap". Areas that are blue have low p_k whereas areas that are red have high p_k (and therefore "hot"). Figure 3.4 shows a succession of heatmaps.

3.2.2 Detection

In addition to the procedures in source localization, we introduce the alternative hypothesis that there exists no source (*null hypothesis*). We define a second set of q_k that represents the probability that a source is not in grid k. At time t, q_k^t is updated similar to p_k^t, but with sensor intensity Λ replaced with just Γ since now $\lambda = 0$ for all k. A common normalizing factor A is used for both p_k and q_k.

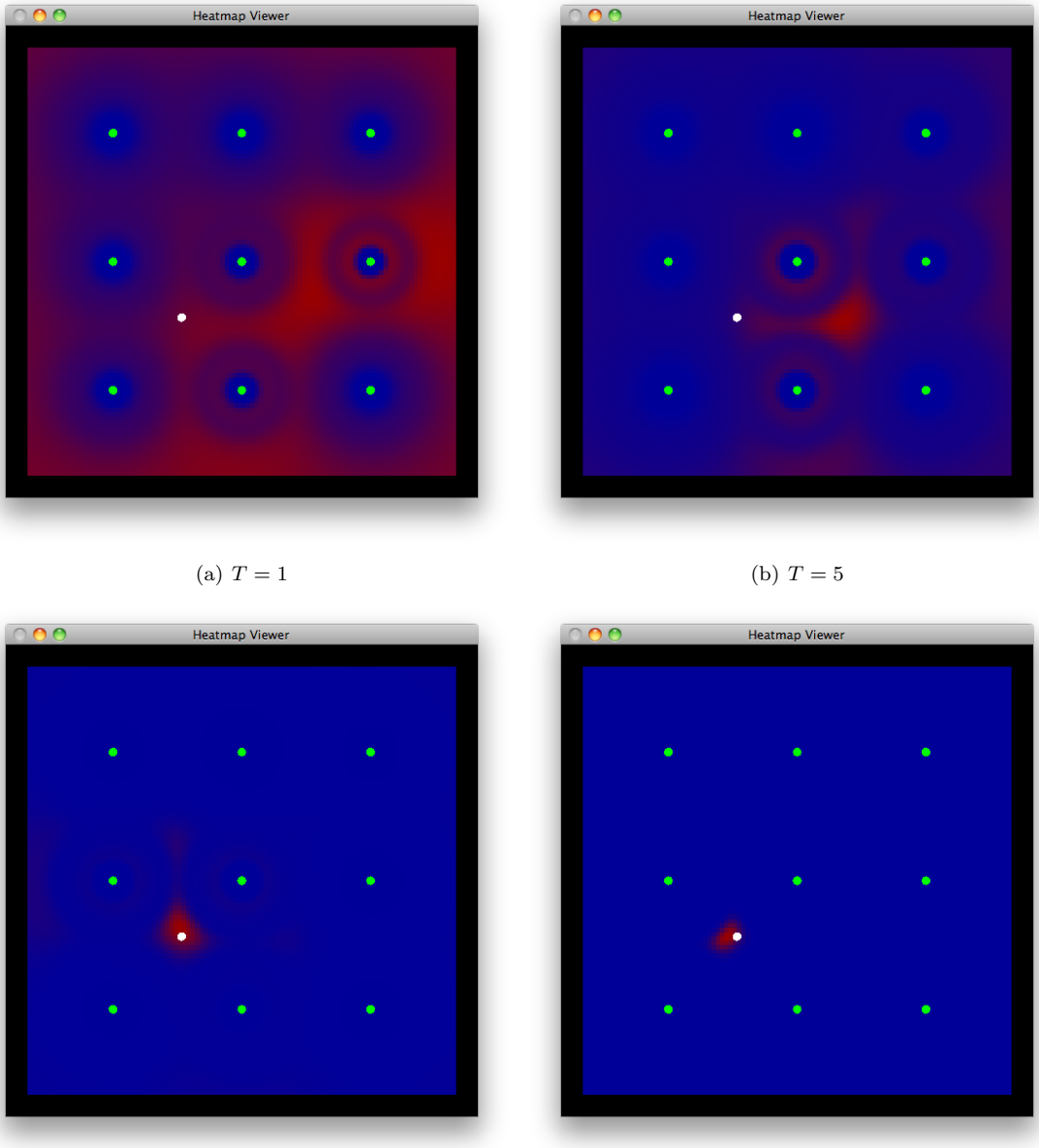
$$A = \sum_{k=1}^{K} (p_k + q_k)$$

Let p'_k and q'_k denote the normalized probabilities. We declare detection at $t = T$ if $\sum p'_k$ exists a certain threshold R and no detection otherwise. By adjusting the prior on source presence and R, we can change the "sensitivity" of the algorithm.

The amount of time allowed before making the detection decision directly affects the performance of the algorithm. Figure 3.5(a) shows the improvement of detection performance going from $T = 3$ to $T = 60$. Spectrum data also improves the accuracy of detection. Typical background noise occupies the lowered end of the spectrum and lacks significant peaks. Therefore for sources like ^{137}Cs that peaks at high energy (662 keV), it is comparatively easier to improve SNR by windowing (using only spectrum within a certain range of energies) or applying algorithm in Equation 3.3. At $T = 9$, there is significant performance improvement by considering spectrum data, as shown in Figure 3.5(b)

3.3 Maximum Likelihood Estimator Method

One other way to deal with the localization problem of unknown source strength is to construct a maximum likelihood estimator and solve it with optimization techniques. Assume a uniform background whose strength has been measured a priorly as Γ. The remaining unknown variables we want to estimate are the source strength μ and source position $X = (x, y)$. Recall from Equation **??** that the frequency of photons received by a radiation detector can be characterized by the discrete Poisson mass function



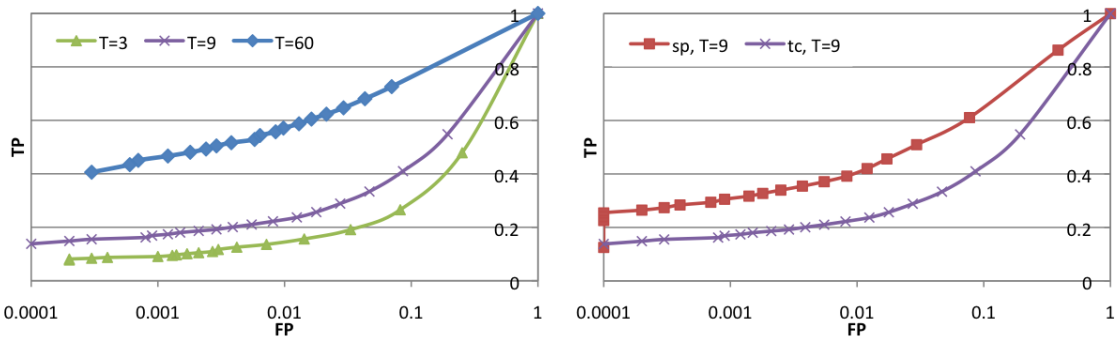
(a) $T = 1$

(b) $T = 5$

(c) $T = 15$

(d) $T = 80$

Figure 3.4: A sequence of heatmaps that map the 2D posterior probability distribution of source location. The field is 100 x 100 m with 9 static sensor evenly placed at (20, 50, 80). The source is a 1mCi ^{37}cS at (36, 37). The uniform background strength is 48 cps. After $T = 15$ seconds, the Bayes algorithm narrows down the probability to a sufficiently small region.



(a) Increasing T improves detection performance.

(b) Given the same amount of time, Bayes algorithm that uses spectrum data performs better.

Figure 3.5: ROC curves of detection using the Bayesian estimation algorithm.

$$P(y|\lambda) = \frac{e^{-\hat{\lambda}}\hat{\lambda}^{y}}{y!} \tag{3.6}$$

To simplify the derivation, we denote $\hat{\lambda}$ as the expected counts a sensor at position D should receive in time t from a source at X.

$$\hat{\lambda}(\mu, X, t) = \frac{\mu A t}{4\pi d^2} + \Gamma t \tag{3.7}$$

where the distance between the source and the sensor is $d = |D - X|$. When there are multiple sensors taking measurements, the combined likelihood becomes

$$\prod_{i=1}^{N} P(y_i|\hat{\lambda}_i) = \prod_{i=1}^{N} \frac{e^{-\hat{\lambda}_i}\hat{\lambda}_i^{\,y_i}}{y_i!} \tag{3.8}$$

The best estimates of μ and X will optimize this function, as well as the log of the function. We take the natural log of the function:

$$\begin{aligned}
\ln\left(\prod_{i=1}^{N} P(k_i|\hat{\lambda})\right) &= \ln\left(\prod_{i=1}^{N} \frac{e^{-\hat{\lambda}_i}\hat{\lambda}_i^{\,y_i}}{y_i!}\right) \\
&= \sum_{i=1}^{N} \ln\left(\frac{e^{-\hat{\lambda}_i}\hat{\lambda}_i^{\,y_i}}{y_i!}\right) \\
&= \sum_{i=1}^{N} -\hat{\lambda}_i + \sum_{i=1}^{N} y_i \ln(\hat{\lambda}_i) - \sum_{i=1}^{N} \ln(y_i!)
\end{aligned}$$

Substituting $\hat{\lambda}$ in, we have

$$-\sum_{i=1}^{N}\frac{\mu At}{4\pi d_i^2}+\sum_{i=1}^{N}y_i\ln(\frac{\mu At}{4\pi d_i^2})-\sum_{i=1}^{N}\ln(y_i!) \tag{3.9}$$

By optimizing this function, we can find the best estimates of source intensity μ_{MLE} and its location X_{MLE}. However, observe that μ_{MLE} can be expressed in terms of X_{MLE}. We take the derivate of function 3.9 with respect to μ.

$$\frac{d}{d\mu}\ln\left(\prod_{i=1}^{N}P(y_i|\lambda)\right)=\sum_{i=1}^{N}-\frac{At}{4\pi d_i^2}+\sum_{i=1}^{N}y_i\frac{1}{\mu}$$

Let it equate to 0 and solve for μ_{MLE}, we get,

$$\frac{1}{\mu}\sum_{i=1}^{N}y_i=\sum_{i=1}^{N}\frac{At}{4\pi d_i^2}$$

$$\Rightarrow \mu_{MLE}=\frac{\sum_{i=1}^{N}y_i}{\sum_{i=1}^{N}\frac{At}{4\pi d_i^2}}=\frac{4\pi}{A}\frac{\sum_{i=1}^{N}y_i}{\sum_{i=1}^{N}\frac{t}{d_i^2}}$$

This technique reduces the variables by one. We then maximize Equation 3.9 in the space of X to get X_{MLE} and μ_{MLE}. This method though simple, may be difficult to implement in real time depending how the maximization routine is implemented and how large the valid set X is.

Chapter 4

System Characteristics

4.1 How Many Sensors Are Required?

The results for data fusion algorithms are better than, but similar to, results for independent sensor algorithms. Therefore, we carry out an asymptotic analysis of sensor density requirements based on the independent sensor algorithm [**?**]. Let Γ be the rate at which photons from the background are measured by the sensor, and let λ be the rate from a source if one is present. The value of λ depends on the location and strength of the source, but for the time being let's assume that λ is given. Let μ_{null} and σ_{null} be the mean and standard deviation of the number of photons measured by the sensor in time T when there is no source present; and let μ_{source} and σ_{source} be the corresponding values when the source is present. Using Gaussian approximation of Poisson distribution with high mean, we have:

$$(\mu_{null} = \Gamma T) \ \wedge \ (\mu_{source} = (\Gamma + \lambda)T)$$

$$(\sigma_{null} = \sqrt{\Gamma T}) \ \wedge \ (\sigma_{source} = \sqrt{(\Gamma + \lambda)T})$$

The no-fusion algorithm rejects the null hypothesis (and claims that a source is present) when the count in time T exceeds a threshold Q. Recall that $bPFP$ is the given upper bound on the probability of false positives and $bPTP$ is the given lower bound on the probability of true positives. For this analysis assume that false positives and false negatives have equal weight, i.e., $bPTP = 1 - bPFP$. Let K be a positive real value such that the probability of values in a standard normal distribution exceeding z is $bPFP$. Then

$$Q - \Gamma T = K\sigma_{null}[T]$$

Since false positives and negatives have equal weight:

$$(\lambda + \Gamma)T - Q = K\sigma_{source}[T]$$

Summing these two equations and substituting for σ:

$$K = \frac{\lambda\sqrt{T}}{\sqrt{\Gamma} + \sqrt{\lambda + \Gamma}}$$

When the source intensity λ is much lower than the background intensity Γ:

$$\lambda \ll \Gamma \quad \Rightarrow \quad K \approx \frac{\lambda\sqrt{T}}{2\sqrt{\Gamma}} \tag{4.1}$$

When a source is r meters away from the sensor, for large r, the rate of photons from the source detected by the sensor is λ, where

$$\lambda = \frac{\Lambda e^{-\alpha r}}{r^2} \tag{4.2}$$

Λ is a constant determined by the strength of the source, and α is the photon absorption rate, per meter, in air. To a first approximation, Λ is the rate of photons per second measured by a sensor when the source is one meter away. From the previous equations

$$\frac{\Lambda}{\Gamma} \ll e^{\alpha r} r^2 : \quad \Rightarrow \quad T \approx \frac{4K^2 \Gamma r^4 e^{2\alpha r}}{\Lambda^2} \tag{4.3}$$

We use this equation to analyze the following situations.

1. **Increasing distance between sensors**: When r is large, *the time T to make a decision increases more rapidly than the fourth power of r*. For example, with $\Lambda = 200$, $K = 3.2$, and $r = 20$ meters, then a 25% increase in r requires a 150% increase in T. Thus timely detection forces dense deployments of sensors.

2. **Shielding radiation sources**: If the source is shielded so that the rate at which photons are detected is halved (and therefore Λ is halved), then the time T to detection is quadrupled.

3. **Increasing sensor density**: Let D be the sensor density: it is the surveillance area in meters-squared divided by the number of sensors. To a first approximation sensor density is proportional to $1/r^2$ where r is the average spacing between sensors laid in a grid. For large r we see that:
   T increases more rapidly than $\Gamma/D^2\Lambda^2$.

Figure 4.1 shows the benefits of dense networks. A sparse network (9 sensors with a single crystal per sensor) won't detect threats rapidly, whereas dense networks (16 sensors with 6 crystals per sensor, increasing photon measurement rates by a factor of 6) can. Dense networks are expensive if each sensor is carried by a security officer; however, if sensors can be placed on street lights or in traffic cones, and the communication demands are low then dense networks are feasible.

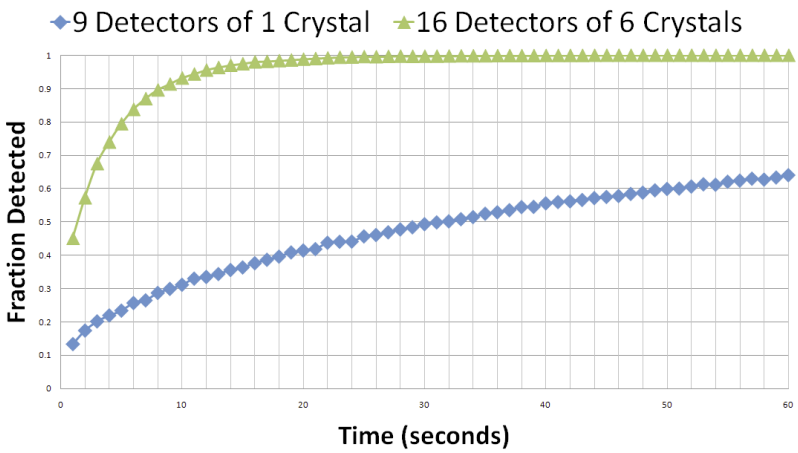


Figure 4.1: Increasing the density of detectors improves detection. The graph compares detection efficiency using 9 single-crystal detectors with that obtained using 16 six-crystal detectors, under otherwise identical conditions.

Dense coverage and limited budgets suggest the use of inexpensive sensors; however, isotope identification requires more expensive sensors that can monitor energy spectra accurately. We discuss source localization and identification, and the impact of cost constraints on network design, later in the paper.

4.2 Where Should Sensors Be Placed?

Sensor placement is a difficult optimization problem because the objective functions, such as minimizing time to detection, are not convex and have multiple minima. Next, we compare placement of identical sensors on a uniform grid with a greedy algorithm and a genetic algorithm. We study a greedy algorithm for two reasons: (1) If sensors are added to the network incrementally — as for example, if security personnel equipped with sensors join teams that have already been deployed at a site — then placing additional sensors where they can do the most good may be more feasible than rearranging the locations of all the existing sensors. (2) The greedy algorithm is simple and we can prove a bound on the goodness of the algorithm compared with the optimum solution.

4.2.0.1 Greedy Algorithm

Assume that the surveillance area is gridded, and associate a random variable with each cell in the grid. The random variable takes on the value 1 if there is a source in that cell and 0 otherwise. The *a priori* probability of a source in each cell is given. In this calculation we assume that each cell has the same probability p of containing a source. Consider a point on the grid and assume that there is a sensor at distance d from it. Measurements by the sensor over some time interval T will allow an algorithm to compute the *posteriori* probability that the cell contains a source. The

smaller the distance d the more certain we are about the presence or absence of a source at that point; equivalently the posteriori probability density will be heavier near 0 or 1, and thus the entropy associated with that random variable is reduced.

Assume that M sensors are deployed over the surveillance region. Let D be the vector of sensor locations with $D[j]$ being the cell of the j-th sensor. Let $f_i(D, p)$ be the *posteriori* probability of there being a source at cell i given sensor placements D and *a priori* probability p. We seek to compute D that minimizes the total entropy

$$F(D) = \sum_i f_i(D,p) \log f_i(D,p) \; + \; (1 - f_i(D,p)) \log(1 - f_i(D,p)) \tag{4.4}$$

The greedy algorithm adds one sensor at a time, increasing the length of D by one. We determine the location of the first sensor to minimize $f(D[1])$, i.e., assuming that there is only one sensor in the field at location $D[1]$. The first security officer coming to a field, assuming that he or she is the only one, will move to location $D[1]$. Given the locations $D[j]$ of M sensors we determine the location of the $(M+1)$-th sensor as follows: We assume that $D[1], \ldots, D[M]$ are fixed and we determine the location l of the $(M+1)$-th sensor that minimizes $F(D+l)$. This corresponds to the $(M+1)$-th security officer moving to the location that best helps the existing deployment of the previous M officers.

Function F has two important properties that ensure the greedy algorithm is within $(1 - 1/e) \approx 63\%$ of the optimum [**?**]: (1) F is monotone, i.e., adding sensors doesn't make the objective function worse and (2) F is *submodular* [13], i.e., there is a decreasing marginal benefit from adding more sensors.

In the calculation, we use the following heuristic.

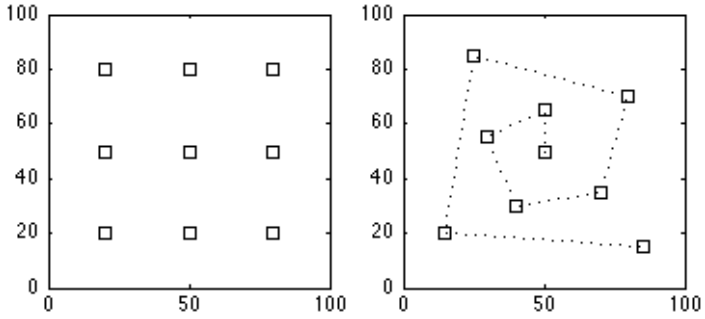
$$f_i(D,p) = Min \left\{ (p + \sum_{m=1}^{M} K(d_{im})), 1 \right\} \tag{4.5}$$

where K decreases as distance d.

The performance of this layout with 9 sensors in a 100 x 100 m field (Figure 4.2(b)) is compared against a grid layout (Figure 4.2(a)) in simulation. Each curve in Figure 4.3 is compiled from 10,000 runs of T seconds with a randomly placed source and 10,000 runs with no source. At $T = 60$ seconds, the new layout outperforms the grid layout by $\approx 5\%$.

4.2.0.2 Genetic Algorithm

Heuristics, such as genetic algorithms, are employed to get good solutions to difficult optimization problems. We compare the results of genetic algorithms with uniform grid layouts. In the experiment, the genetic algorithm (GA) uses a population of 400 chromosomes. Each chromosome encodes



(a) Grid layout

(b) Greedy entropy minimization layout. $K(d) = 2/d$

Figure 4.2: Sensor placement layouts. The dotted line connects positions in the order they are chosen by the greedy algorithm at each step. The order starts from the center.

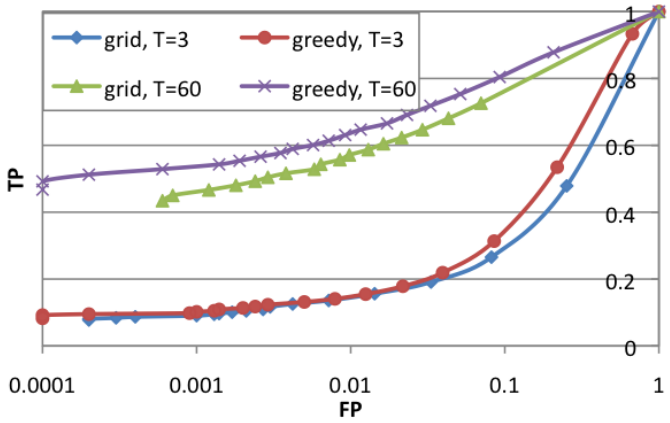


Figure 4.3: Comparison of greedy and grid sensor layout using ROC curves. The source strength is 200 cps at 1m. The background strength is 8 cps.

the coordinates of M sensors. Initially, the coordinates of the sensors are chosen at random. At each epoch, each chromosome determines the value of the objective function for its layout of sensors. The best chromosomes (those with largest values of the objective function) are mated, mutated and proceed to the next epoch. In these experiments, a heuristic was used to compute the objective function.

Figure 4.4 illustrates a typical result from the GA for a set of 9 sensors in a field of 100 x 100m. The layout is equivalent to tesselation of the field by discs of radius proportional to the sensor sensitivity.

Summary Uniform grid layouts of sensors are not optimal. Simple greedy algorithms that minimize entropy do reasonably well, and these algorithms can be adopted when the number of security personnel available is not known for certain, and where security officers arrive at a scene incrementally, in small groups. Genetic algorithms and other heuristics do better than uniform grids, and we

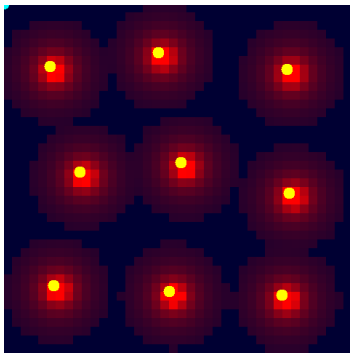
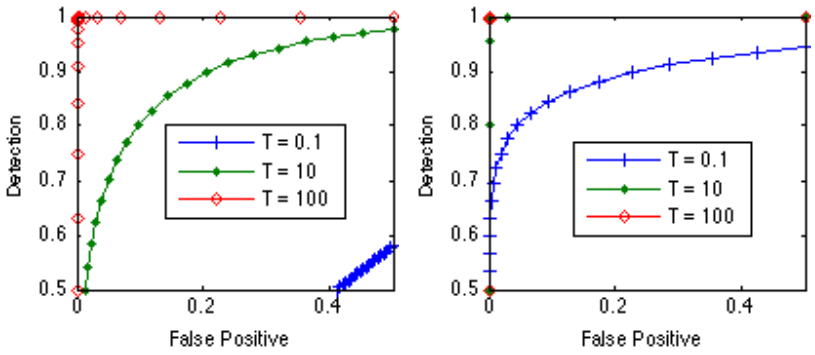


Figure 4.4: A candidate optimal layout for 9 sensors (shown as yellow discs), as suggested by the GA. The red shading indicates the sensitivity.detection sensitivity.

are exploring these heuristics further.

4.3 Does Mobility Help?

Given a fixed amount, sensitivity of stationary detectors are limited by its distance to the source. Figure 4.5 plots the detection performance of two sensors at different distances from a potential source. Sensor B is 4 times closer to the source than Sensor A and therefore has sensor intensity λ 16 times higher. While all other conditions hold the same, at $T = 10$ seconds, we get the desired ROC curve from Sensor B, but not Sensor A.



(a) Sensor A. $d = 10$, $\lambda = 2$, $\Gamma = 8$ (b) Sensor B. $d = 2.5$, $\lambda = 32$, $\Gamma = 8$

Figure 4.5: ROC curves of the probability of detection.

To further quantify how distance affect detection accuracy for single detector, Figure 4.6 plots the time it requires to achieve a certain true positive rate (0.99) at different distance. The growth in time to detect is more than the fourth power of distance when $d > 10$. This is especially obvious when the tolerance on false positive is small (FP=0.01).

One way to deal with this problem is to deploy a dense sensor network. But since placing arbitrarily many detectors in a large field is not feasible given a budget constraint, the detectors

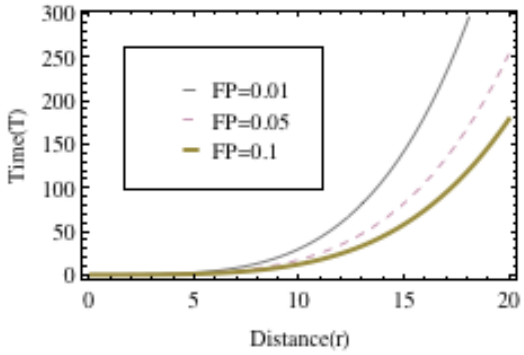
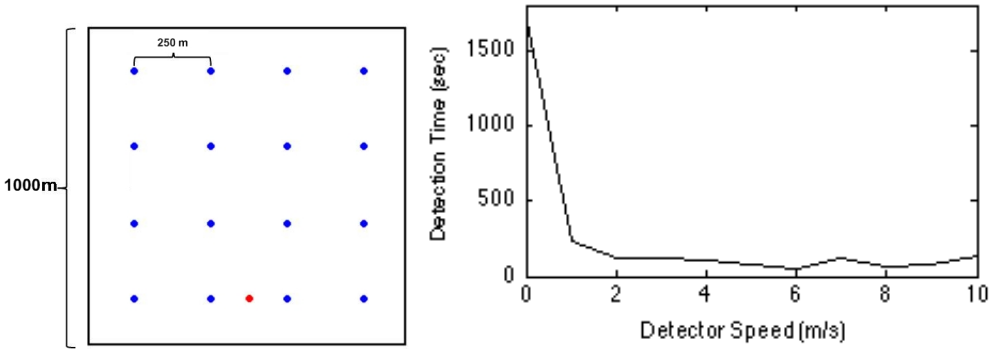


Figure 4.6: Time to detect as a function of distance for single sensor. The desired true positive rate is 0.99.

simply have to move in order to achieve the desired speed and accuracy to detect, localize, and identify. We studied the performance of a simple sensor redeployment algorithm that works as follows: At each Bayesian update cycle, move each sensor towards the "hottest" spot in the field at a constant speed. Figure 4.7 shows the time to detect as a function of detector speed in a 1000 x 1000 m field with 16 detectors in 4 x 4 grid formation (Figure 4.7(a)). When the detectors are stationary, it takes on average of 28 minutes to detect a source in the center of 4 detectors. But when the detectors are allowed to move, even at only 1 m/s, the time to detect dramatically decreases to 4 minutes.



(a) 16 sensor placed in a 1000 x 1000m field (blue dots) with a source (red dot).

(b) Time to detect as a function of detector speed.

Figure 4.7: Simple sensor redeployment algorithm.

We made two important observations from this result:

1. The gain from sensor speed becomes negligible after a certain speed. In this experiment, it is 2 m/s. What this could means is that depending on the field size and number of sensors, there is likely an optimal sensor speed.

2. This simple heuristic is not optimal since it takes no advantage of sensor coordination. It is easy to see why this is the case - directing all sensors towards the hottest spot results in

redundant information. While assigning a few sensors to check out the hottest spot, it may be more beneficial to allocate the rest to second or third hottest points. This observation leads to the work on information-driven sensor redeployment method described by other people in our group - Matt Wu and Daniel Obenshain [29], and independently from several other groups [21] [**?**] [22]. The idea is to maximize the information gain (or minimizing the entropy) at each time step by considering sequence of positions the sensors may take in the future. The problem is similar to optimal sensor placement in Section 4.2.0.1, taking into account of prior measurements.

Chapter 5

Implementation of An Autonomous Agent

In Section 4.3, we showed that sensor mobility significantly improves the system performance in terms of detection, especially when the area to cover is too large for deployment of a dense network. There are two ways through which we can implement sensor mobility: 1) human agents or 2) autonomous robotics agents. Autonomous agents have the advantage over human agents in that they can be deployed in potentially dangerous areas, such as mine field. They are also suitable for routine labor-intensive tasks such as surveillance of land and sea borders. In this chapter, we describe the initial work of implementing an autonomous agent for radiation detection.

5.1 Hardware

To focus on the integration of radiation sensor and autonomous agent, we constructed the system from a modified version of the commercially available *Create* robot from *iRobot corporation*. Create is the developer's variant from iRobot's popular Roomba intelligent vacuum machine. This off-the-shelf robotic platform requires no custom hardware or construction to have it up and running, and has become very popular among educators and hobbyists [**?**]. Create robot platform houses a variety of basic peripheral sensors and a spacious cargo bay for mounting other sensors. These sensors can be connected through Create's 25-pin expansion port. Create robot has two degrees of freedom, which is sufficient for maneuvering and surveying a level plan. Though low-cost and simple, studies have shown that these robots, with proper range sensors and algorithms for adjusting its limited odometry, are capable of localizing itself and even performing simultaneous localization and mapping (SLAM) [23] [24]. Create robot, like Roomba, supports a *serial command interface* API published by iRobot. This interface allows commands and sensor data to be sent to and from the robot as if it is a serial device [**?**].

A Verdex Pro XM4 from *Gumstix* is added to the Create robot as its processing unit. Verdex

Table 5.1: iRobot Create Robot with Gumstix Microprocessor

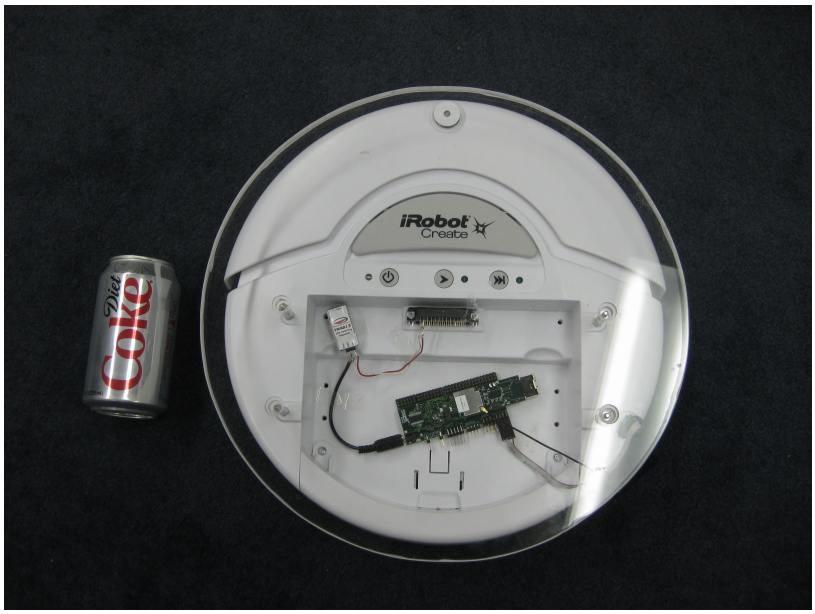
Sensors (built-in)	Actuators (built-in)	On-board processing unit
Wheel encoders x 2	Wheel motors (L & R)	Verdex Pro XM4 (Gumstix)
Bump sensors x 2	LEDs x5-7 (color)	NetPro-VX (Gumstix)
IR Wall sensors x 1	Speaker x 1	Robostix (Gumstix)
Cliff sensors x 7		5v 1A voltage regulator (Dimension Engineering)
		USB serial assembly (Acroname)
		Miscellaneous parts for customized cables (Digikeys)

Pro is a 32-bit microprocessor with 64MB RAM and a special version of Unix operating system preloaded. It is chosen over the *command module* from *iRobot* for its better processing power and its light-weight. An NetPro VX board is attached to the Verdex Pro XM4 to allow wireless and wired communication from other computers to the robot. Connection between the processor and the robot is done through a custom-made serial cable. The Verdex Pro board is powered by the robot's rechargeable battery. A voltage regulator is used between the connection to down regulate the voltage output on the 25-pin connector from 12V to 5V [1]. A list of parts used in the construction is provided in Table 5.1. These customizations are derived from suggestions given in The Robotics Primer Workbook [**?**].

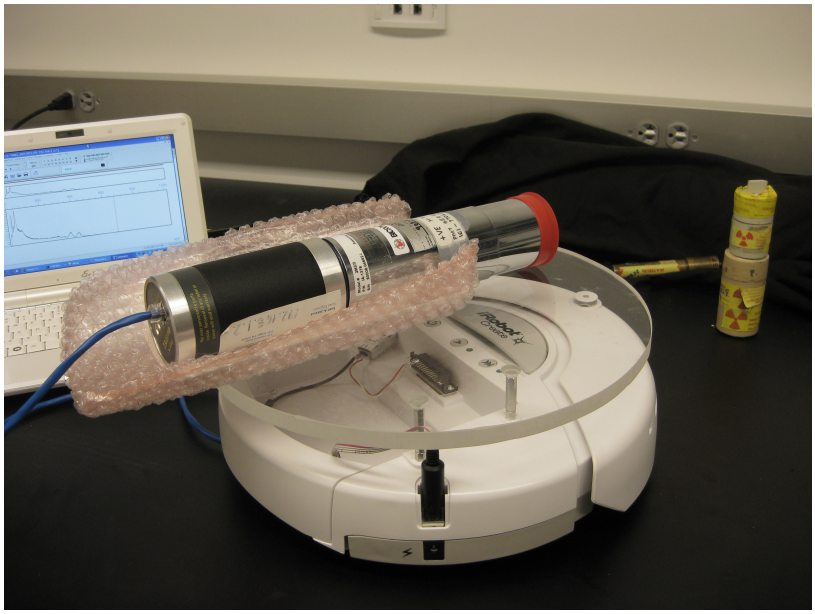
A photomultiplier tube (PMT) with a dsciSPEC digital spectrometer from ICx Technologies is mounted on the robot. The dsciSPEC is a high performance, digital gamma spectrometer housed in a 14-pin tube base [**?**]. Photons are collected and amplified inside the PMT as current, which is then recorded and analyzed inside the spectrometer. Power, communications and control for the spectrometer is provided through an Ethernet connection, also known as the power over Ethernet (PoE) interface. An interface IP protocol is available for querying sensor data over the network. An Acer Eee PC netbook is used as a relay between the sensor and a remote data server. Figure 5.1 shows how the system is set up.

5.2 Software

Player v2.6 is installed on the Verdex Pro XM4 microprocessor . *Player/Stage* is a free robotics software suite developed in USC [1]. *Player* is a device-independent server that provides network transparent robot control. *Stage* is a lightweight, highly configurable 2D robot simulator that supports large, heterogeneous populations. It also supports a wide array of device driver plugins for simulating responses from peripheral sensors such as SICK laser sensor. Figure 5.2 gives an example of versatility and flexibility of the *Player/Stage* setup. The client program talks to *Player* over a TCP socket, reading data from sensors, writing commands to actuators, and configuring devices in real time. *Player* supports a wide spectrum of robot hardware through a rich collection of device



(a) Create robot is controlled by gumstix microprocessor through a custom made serial cable. The microprocessor can be accessed from another computer through wired and wireless connection. Power is drawn from the robot to power the microprocessor.



(b) The dsciSPEC spectrometer is connected to a netbook through Ethernet. A program running on the netbook queries for sensor data once every second and sends the data packet to a remote server through wireless.

Figure 5.1: Hardware setup for radiation searching autonomous agent.

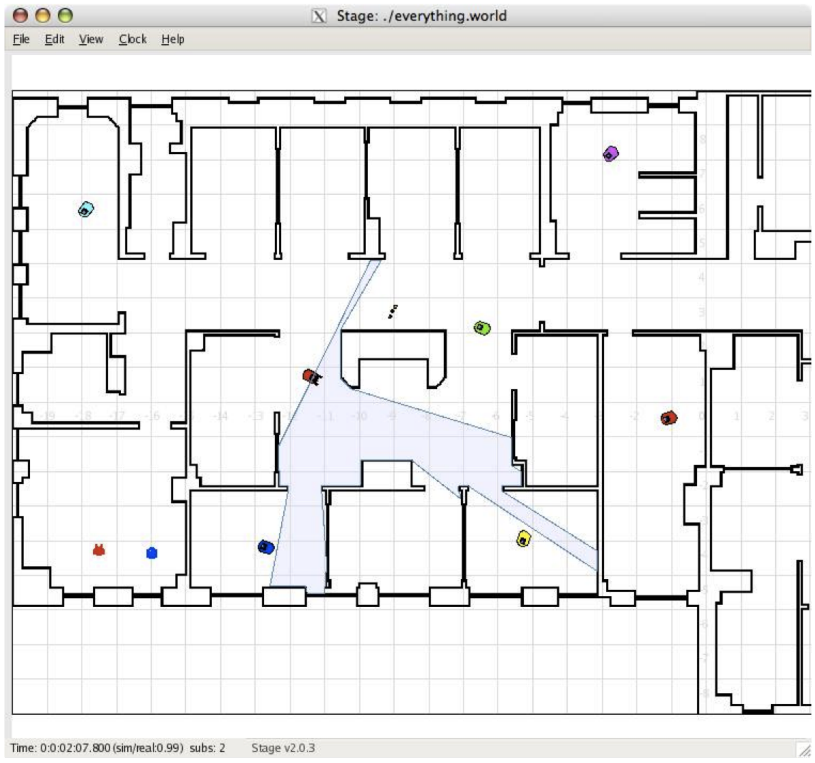


Figure 5.2: Screenshot of simulating a population of robots in *Player/Stage* .

drivers available. Likewise, *Stage* offers many plugins for sensor-based simulation. Clients developed on *Player* and verified on *Stage* can be easily transferred to the actual robot. Using this setup, a client running remotely can control the robot through the *Player* server installed on the microprocessor.

Communication between the on-board Netbook and the remote server builds on top of *Spread*. *Spread* is an open source toolkit that provides a high performance group communication system that is resilient to faults across local and wide area networks. It functions as a unified message bus for distributed applications, and provides highly tuned application-level multicast, group communication and point to point support [**?**]. Clients distributed across the network can either each subscribes to *Spread* servers running locally or to one on a single host [**?**]. This high level communication package is easy to scale. There are also client libraries support in multiple languages, including C++, Java, and Python. The client program running on the on-board Netbook sends an XML request to the dsciSPEC once every second through Ethernet connection. The request queries for the photon count histogram with respect to energy received since last request. The dsciSPEC returns an XML response that consists of detected spectrum in 1024 bins, the actual livetime, and timestamp. The program then parses the XML response and packages it into a single message to be sent to the remote server through the *Spread* network. Figure 5.3 shows pictorially how communication is set up between the agent and the server.

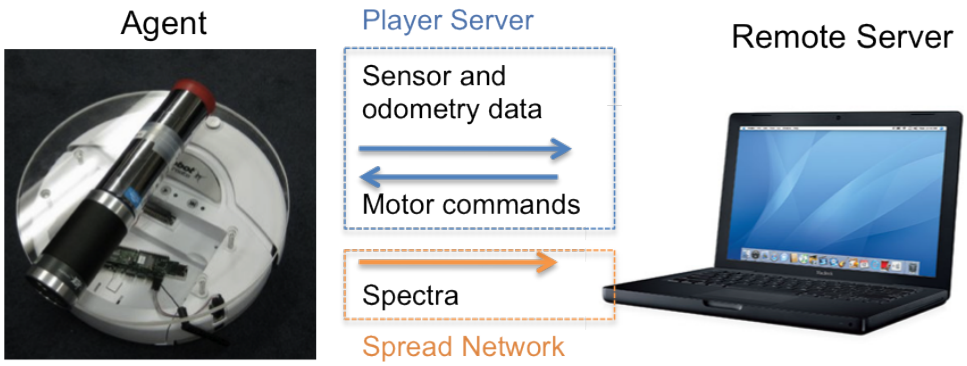


Figure 5.3: Software architecture pictorial diagram.

Sensor data is processed at the remote server for fusion at each time step. Bayesian parameter estimation is carried out to compute the posterior probability distribution of source location. This location is then transmitted back to the *Player* server on the robot as the new goal.

5.3 Robot Maneuvering

For the robot to maneuver in a realistic environment that contains obstacles, it has to keep track of its relative position to the environmental objects. If the environment is completely foreign to the robot, routines such as SLAM together with onboard range sensor (e.g. sonar, laser, ...etc..) are necessary for it to go from point A to point B without running into obstacles. However if the environment is previously mapped out, the robot only needs to know its position in the map coordinates. This position can be acquired from onboard GPS unit or approximated with robot's odometry, i.e. the number of wheel rotations and the sequence of angle changes since start. In reality, a combination of these techniques is often required to achieve good maneuvering, but here as an initial approach, we assume an ideal scenario that the robot has perfect knowledge of the world a priorly and perfect odometry readings. The problem is thus reduced to path planning between any two points on a map with well specified obstacles. We present an implementation of the *Sample-Based Motion Planner* in this section.

5.3.1 Sample Based Motion Planner

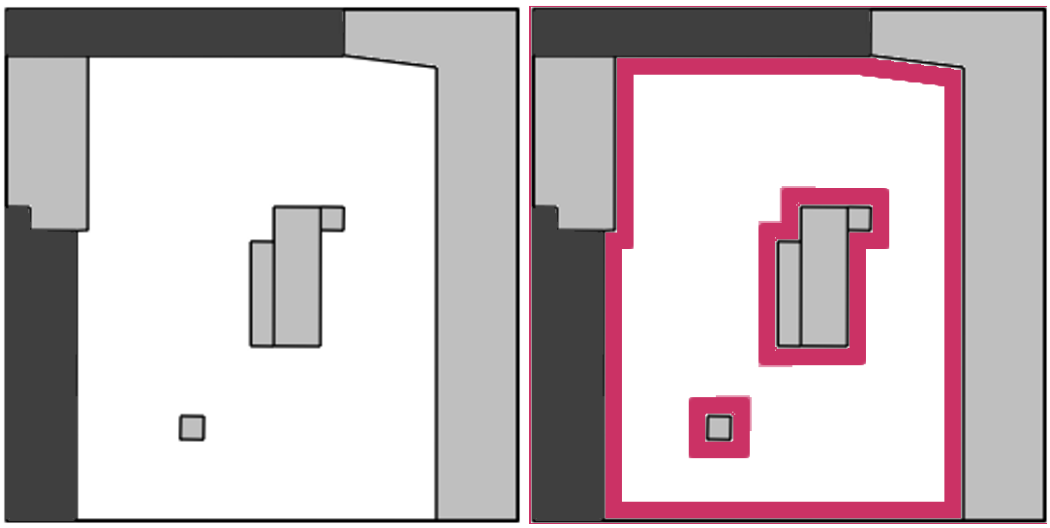
We implement a sample based motion planner in C++ for a disc-like robot (e.g. iRobot Create) operating in 2-dimensional environment with objects of arbitrary shape. The program takes the map of an bounded environment, calculates the configuration space (C-space) based on the diameter of the robot, samples the C-space using n points, constructs a connected roadmap from the sampled points, and finally outputs a collision-free path in the C-space given start and end points (V_{start}, V_{goal}). This

implementation takes a *multiple query* approach and stores the roadmap for future request. The result is demonstrated in *Player/Stage* simulation.

5.3.2 Procedures

1. Read environment map: The program reads in the layout of the environment as a *png* picture, whose name is provided at command line. The picture is true to scale, the real width in meters is also provided at command line. Obstacles in the environment are all colored. The *png* file is processed using an open source graphics library - *pngwriter* [**?**], which allows access to each pixel in the picture as well as some basic plotting tools. Figure 5.4(a) is an example of input file.

2. Construct C-space: The diameter of the disc robot is read in from command line arguments. The construction of C-space is typically hard, but the disc shape greatly simplifies the calculations. The algorithm first determines the edges of the obstacles, and extends each edge into the the empty space for the amount of the robot radius. The environment is assumed to be closed, and the surrounding boundaries are also treated as obstacles. The run time for this algorithm is $O(pw \times ph)$, where pw and ph are the pixel width and height. Figure 5.4(b) shows the C-space constructed from Figure 5.4(a).



(a) The original map (b) The C-space is constructed and colored in pink

Figure 5.4: Construction of C-space from environment map

3. Sampling in the C-space: The number of points sampled is given as command line argument. Each sampled point v_i is part of the Halton sequence described below [**?**]:

$$v_i = (r(i, p_1), r(i, p_2))$$

where $p_1 = 2$, $p_2 = 3$. The $i - th$ sample is the base-p representation for i, which has the form $i = a_0 + pa_1 + p^2a + p^3a_3 + \ldots$. $v_i \in [0, 1]$ is obtained by reversing the order of $p's$ and moving the decimal points.

$$r(i, p) = \frac{a_0}{p} + \frac{a_1}{p^2} + \frac{a_2}{p^3} + \frac{a_3}{p^4} + \ldots$$

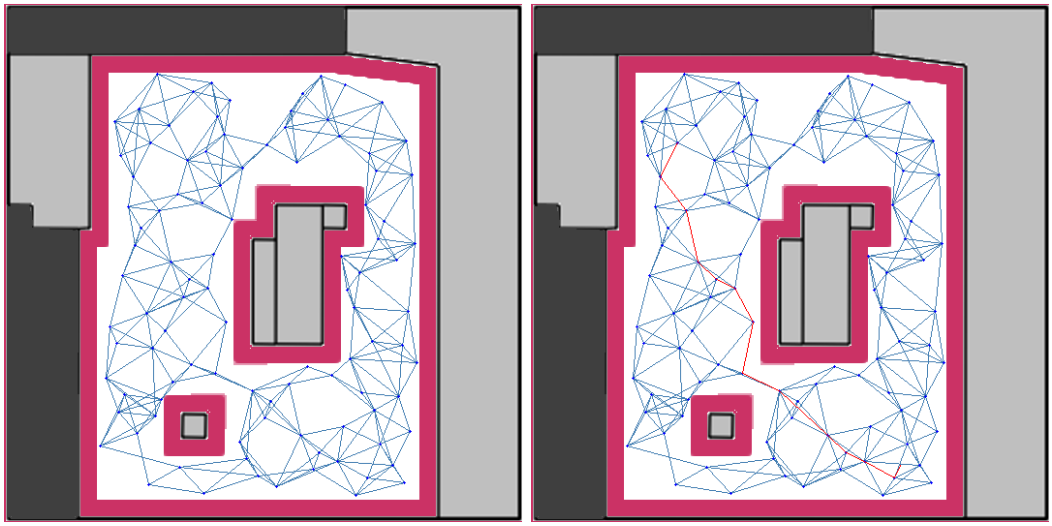
v_i are then scaled proportional to the width and height of the map. The drawing is done in sequential order. Points that fall inside the C-space are discarded. This implementation uses the routines implemented by John Burkardt [**?**]. The routine always returns the same sequence. To get a more randomized sequence, one can choose to seed at different positions other than 0 and/or to leap through sequence. The running time for this is $O(n)$, where n is the number of samples.

4. Construct roadmap: Each valid sample points in the world is treated as a vertex. The algorithm iteratively connects each vertex to n closest neighboring vertices, where n is a command line argument. The connection is bidirectional, that is, if $Edge(i, j)$ exists, $Edge(j, j)$ also exists. If v_j is one of the n closest neighbors to v_i and $Edge(i, j)$ already exists, then the next closest vertex v_k is chosen. This algorithm connect clustering vertices to the rest of the graph. The result in Figure 5.5(a) uses 100 vertices each with 3 degree of connectivity. This implementation has running time $O(n^2d)$, where d is the degree of connectivity.

5. Graph search: The above procedures are one-time operations, the resulting roadmap is stored in memory for future queries. Each query consists a P_{start} and P_{goal} pair. The program firsts determines the closest two vertices v_s and v_g such that the paths connecting from P_{start} to v_s and P_{goal} to v_g lie in C-free. The program then uses A* graph searching algorithm for constructing shortest path form v_s to v_g, using euclidean distance to v_g as heuristic. Figure 5.5(b) plots the path using $v_s = (-1.0, 1.0)$ and $v_g = (1.1, -2.0)$. The running time of this step can be approximated with the that of Dijkstra's algorithm, which uses $O(|E| + |V|log|V|)$. $|E|$ and $|V|$ are the number of edges and vertices. Translate that to the notations here, it is equivalent to $O(nd + nlogn)$.

5.3.3 Results

Using the above environment as an example. Below are the screenshots taken in *Stage* for a *Roomba* like robot. The red line sticking out from the robot shows where it is facing. The gray trail is the



(a) Roadmap constructed using 100 vertices and connectivity of 3 degree

(b) A* graph search for path finding

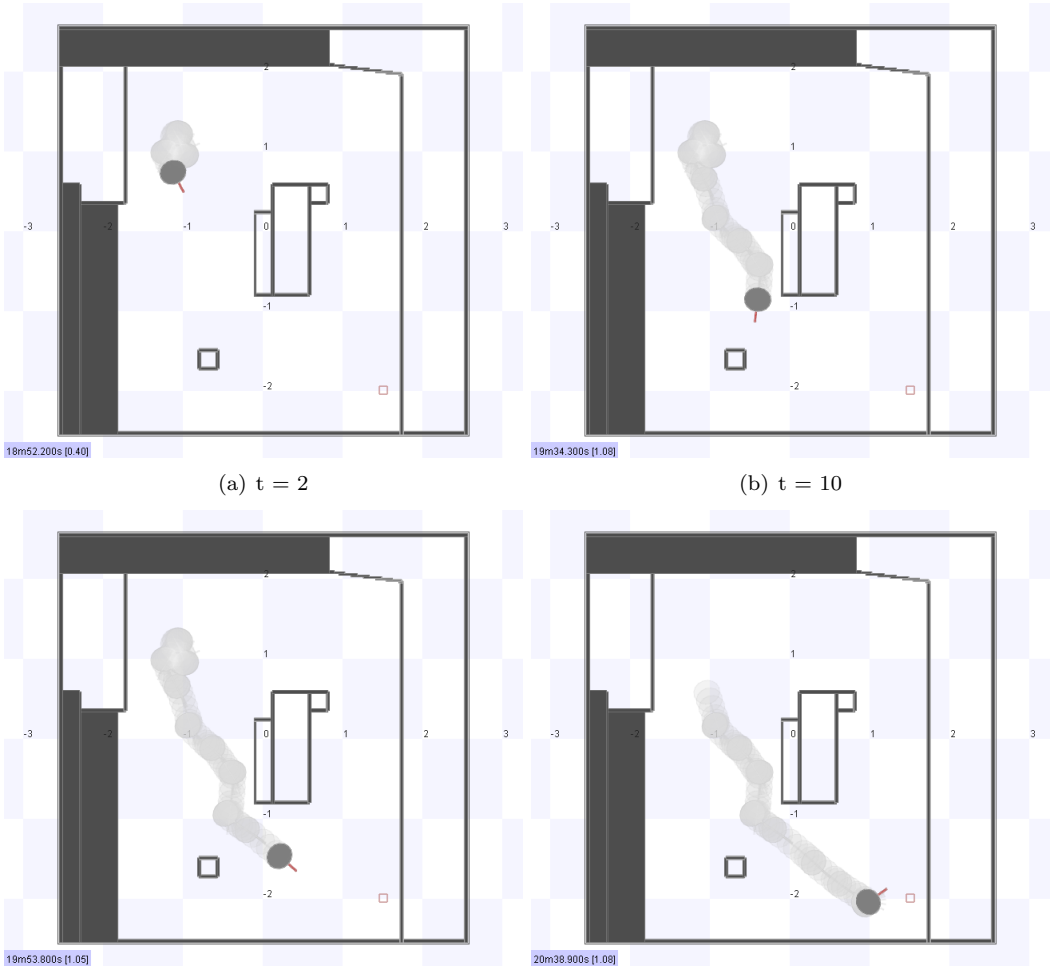
Figure 5.5: Roadmap construction and graph search

footprint of the robot.

5.3.4 Experiments

The following experiments use a much larger and more complex map. When using 500 samples and degree of 3, the resulting path, as seen in Figure 5.7(a), is much longer than expected and includes several backtracking. This is because the shortest path passes through a part of the graph that is not connected to the rest and is thus not visible to the graph search algorithm. This is especially common for compartments with narrow entry way. There are two ways to fix this problem - increasing the number of samples, and/or increase the degree of connectivity.

By increasing the degree of connectivity from 3 to 5, the graph becomes connected, as shown in Figure5.7(b). The same results is achieved by increasing the number of samples from 500 to 1000 in Figure 5.7(c). For the level of complexity of this example, the difference in speed of the two approaches is negligible. However on a closer look at the running time analysis. This implementation has running time $O(n + n^2d + nd + nlogn)$. The n^2 terms dominates the running time, therefore increasing the number of samples is actually more expensive. Of course, this greatly depends on the implementation. The effectiveness of the two approaches should also be considered. Unfortunately, this cannot be as easily compared as running time analysis.



(a) t = 2 (b) t = 10

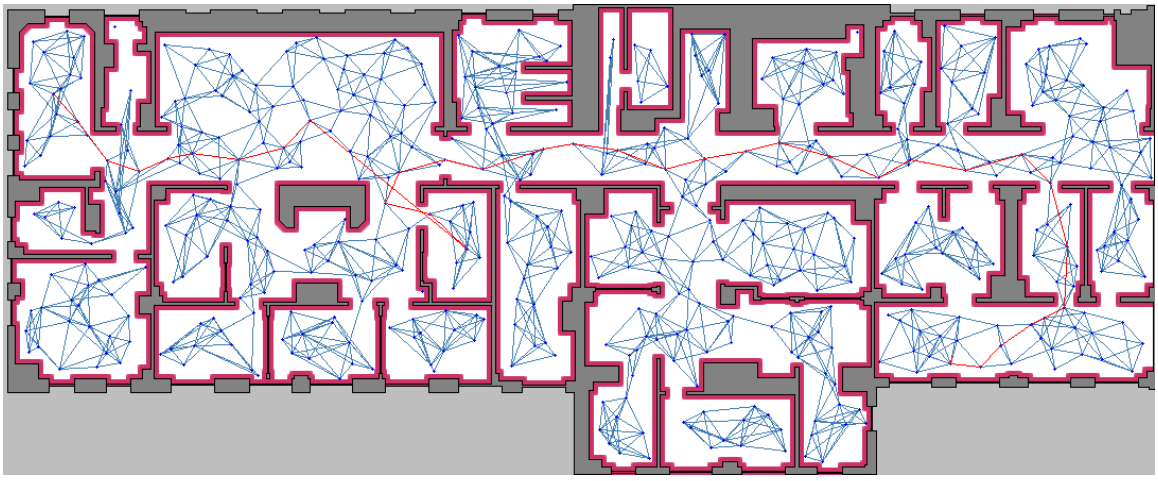
(c) t = 15 (d) t = 30

Figure 5.6: Roadmap construction and graph search

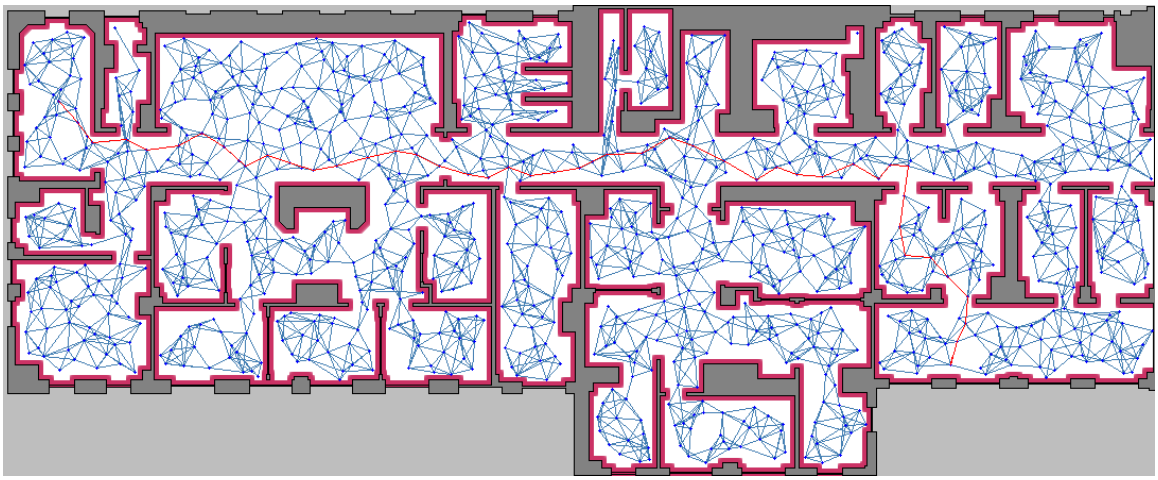
5.3.5 Possible Improvements

Dynamically select n and d The program treats number of samples (n) and degree of connectivity (d) as user input. However, as shown in the experiments, it doesn't always find a path between two reachable points. It is not a big step up to dynamically choose n d by testing graph connectivity. Though this inevitably adds to the overhead.

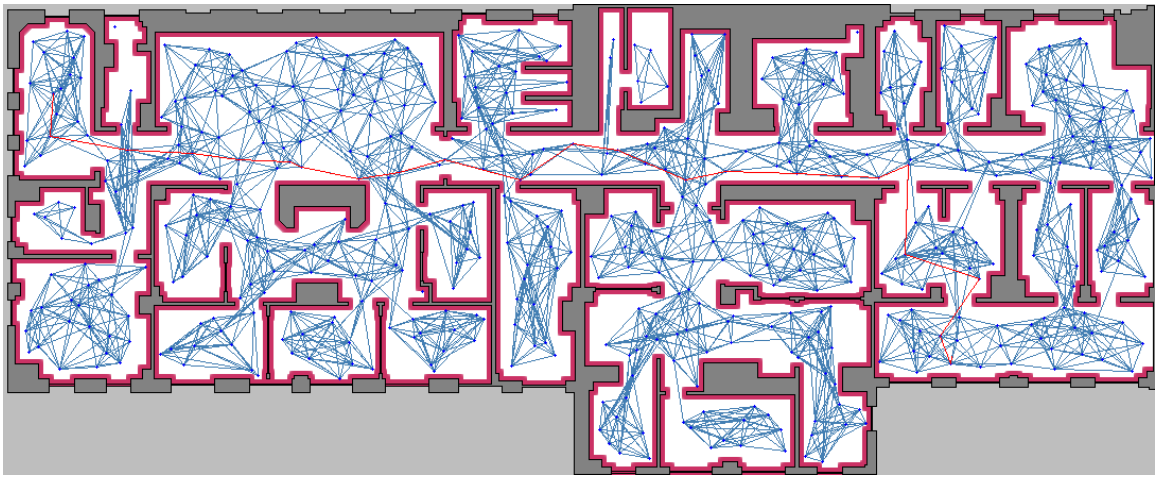
Greedy Sampling The Halton sequence sampling method gives a pretty good representation of the environment. However, on a closer observation, one can see that not all points are equally important in the graph. For example, points at entry way are important, where multiple points in one compartment are redundant. Conceptually, it is best to pick a set of points that optimally reduces the "entropy" in the environment. (i.e. if we only 20 points in the example above, most points will lie in the "entry way"). However, to actually acquire this set may be too computationally intensive.



(a) n = 500, degree = 3



(b) n = 500, degree = 5



(c) n = 1000, degree = 3

Figure 5.7: Roadmap construction and graph search

One possible approach is to choose the points "greedily". That is, at each step of generating samples, look ahead x number of samples, choose the one that maximally increase the "coverage" of the graph and discard the rest. It is however unclear how well this heuristic works without further investigation.

Chapter 6

Conclusion

6.1 Contributions

In this thesis, we made three major contributions in the field of radiation detection that were not previously addressed in related works.

First of all, we provided analysis on the characteristics of a radiation sensor network. We studied in details the relationship between true positive rate, false positive rate, sensor density, and the time to detect and showed that data fusion improves the system performance in terms of detection and further quantified this improvement. We showed that a typical grid layout of sensors in an open field is not optimal based on rigorous simulation results and described two algorithms for computing better layouts.

Secondly, we built a simulation framework based on a multi-player virtual gaming environment that is powerful enough to address complex physical interactions between photons, obstacles, and sensors. This virtual environment allows us to simulate sensor response in highly dynamic, close to reality environments that produces non-uniform background noise. We developed a powerful parameter estimation method based on Bayesian model for detection and localization that takes advantage of the world model and consumes it as prior information. The multiplayer aspect also makes it possible for us to observe realistic adversarial behaviors and develop team counter act measures through game plays.

Lastly, we demonstrated that sensor mobility significantly improves the system performance in terms of detection and localization and showed that slow motion is sufficient. Furthermore, we described an initial work on incorporating autonomous robotic agent into the system architecture and presented implementation for simple robot searching and maneuvering in a well known environment.

6.2 Future Work

The radiation detection problem in a distributed sensor network is extremely challenging. Numerous simplifications are required in order to begin understanding the problem. For future work, there are several interesting directions to pursue.

Improving algorithm performance Our work has shown that different algorithms have different strengths. For example, a simple triangulation type of algorithm for localization may perform better than Bayes localization because the latter has a much larger parameter space. It is possible to use the results from other algorithms as prior information in the Bayesian method to achieve better performance.

Considering sensor directionality Some of the higher end sensor can provide directionality data of received photons. But none of the current algorithms in the field of radiation detection takes advantage of this information, which can potentially significantly improves sensor SNR. It is important to understand whether directionality helps in a DSN and how it may changes sensor redeployment strategy.

Considering multiple moving sources Most of the work done in radiation detection assumes single static source. In reality, there are likely multiple moving, benign sources that we must account for in our algorithm. Our current algorithm is capable of tracking slow moving source (¡ 2 m/s). In the future, we will look into more robust methods such as Kalman filter.

Designing sensor motion There are three types of sensor motions that have been considered: 1) random, 2) pre-defined path following, and 3) directed motion. The only type that has been studied is directed real-time redeployment, which is a suitable model for autonomous agents but not for human agents since they have much higher fidelity than robots. We have begun to quantify the benefit from human random motion in terms of detection. Another interesting problem is to design a set of paths to follow for the scenarios where human agents need to search through a large open field.

Constructing a group of aerial sensors The work on ground mobile sensor is ideal for indoor search and surveillance. For outdoor environment of uneven terrains, such as border protection, aerial vehicles are more ideal. We can extend what we learned from the work on ground robot to aerial mobile devices for better maneuvering ability.

Bibliography

[1] M Anderson, L Thaete, and N Wiegand. Player/stage: A unifying paradigm to improve robotics education delivery. pages 1–5, Jun 2007.

[2] B Boser, I Guyon, and V Vapnik. A training algorithm for optimal margin classifiers. *Proceedings of the fifth annual workshop on ...*, Jan 1992.

[3] A Mielke D Torney S Brennan. Radiation detection with distributed sensor networks. pages 1–3, Jul 2004.

[4] S Brennan, A Mielke, D Torney, and A Maccabe. Radiation detection with distributed sensor networks. *COMPUTER*, Jan 2004. simulation tool for DSN.

[5] M Chandy, C Pilotto, and R McLean. Networked sensing systems for detecting people carrying radioactive material. *Networked Sensing Systems*, Jan 2008.

[6] J Chin, D Yau, N Rao, Y Yang, and C Ma. Accurate localization of low-level radioactive source under noise and measurement errors. *... of the 6th ACM conference on Embedded network sensor ...*, Jan 2008.

[7] A Gunatilaka, B Ristic, and R Gailis. On localisation of a radiological point source. *Information*, Jan 2007.

[8] A Gunatilaka, B Ristic, and R Gailis. Radiological source localisation. *dspace.dsto.defence.gov.au*, Jan 2007.

[9] R L Heath. Scintillation spectrometry. pages 1–539, Mar 1997.

[10] J Howse, L Ticknor, and K Muske. Least squares estimation techniques for position tracking of radioactive sources. *Automatica*, Jan 2001.

[11] K Jarman, L Smith, and D Carlson. Sequential probability ratio test for long-term radiation monitoring. *2003 IEEE Nuclear Science Symposium Conference ...*, Jan 2003.

[12] Glenn F. Knoll. Radiation detection and measurement. page 754, Jan 1989.

[13] A Krause, A Singh, and C Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *The Journal of Machine Learning Research*, Jan 2008.

[14] A H Liu, M M Wu, K M Chandy, D Obenshain, M Smith, and R McLean. Design tradeoffs for radiation detection sensor networks. pages 1–12, Apr 2009.

[15] M Morelande, B Ristic, and A Gunatilaka. Detection and parameter estimation of multiple radioactive sources. *Information Fusion*, Jan 2007.

[16] K E Nelson, T B Gosnell, and Knapp D A. The effect of gamma-ray detector energy resolution on the ability to identify radioactive sources. *Radiological And Nuclear Countermeasures Program*, pages 1–50, Mar 2009.

[17] J S Dreicer D C Torney T T Warnock R J Nemzek. Distributed sensor networks for detection of mobile radioactive sources. *IEEE Transactions on Nuclear Science*, 5(4):1693–1699, Sep 2004.

[18] HF Ji T Thundat L Pinnaduwage. Moore's law in homeland defense: An integrated sensor platform based on silicon microcantilevers. *IEEE Sensors Journal*, 5(4):774–785, Jul 2005.

[19] N Rao, M Shankar, J Chin, D Yau, and S Srivathsan .... Identification of low-level point radiation sources using a sensor network. *Information Processing in Sensor Networks*, Jan 2008.

[20] N Rao, M Shankar, J Chin, D Yau, and Y Yang. Localization under random measurements with application to radiation sources. *International Conference on Information Fusion*, Jan 2008.

[21] B Ristic, A Gunatilaka, M Rutten, and M DSTO. An information gain driven search for a radioactive point source. *Information Fusion*, Jan 2007.

[22] B Ristic, M Morelande, and A Gunatilaka. A controlled search for radioactive point sources. *Information Fusion*, Jan 2008.

[23] B Tribelhorn and Z Dodds. Evaluating the roomba: A low-cost, ubiquitous platform for robotics research and education. *Robotics and Automation, 2007 IEEE International Conference on*, pages 1393 – 1399, Mar 2007.

[24] Ben Tribelhorn and Zachary Dodds. Roomba and mac os x: Cross-platform vision and robotics for ai. *American Association for Artificial Intelligence*, pages 1–6, Oct 2006.

[25] Wikipedia. Compton scattering — Wikipedia, the free encyclopedia, 2010. [Online; accessed 28-May -2010].

[26] Wikipedia. Radiation — Wikipedia, the free encyclopedia, 2010. [Online; accessed 28-May -2010].

[27] S Wold. Principal component analysis. *Chemometrics and intelligent laboratory systems*, Jan 1987.

[28] M Wu, A Liu, and M Chandy. Virtual environments for developing strategies for interdicting terrorists carrying dirty bombs. *International Society of Crisis Response and Management Conference*, pages 1–5, May 2008.

[29] M Wu and D Obenshain. Using submodular function optimization for mobile radiation detector path planning, 2009. [Report].

[30] K. P Ziock. The lost source, varying backgrounds and why bigger may not be better. 632:60–70, Jan 2002.