Statistical Models for Improving the Rate of Advance of Buried Target Detection

Systems

by

Jordan M. Malof

Department of Electrical and Computer Engineering
Duke University

Date: __________________________
Approved:

___________________________
Leslie Collins, Supervisor

___________________________
Kishor Trivedi

___________________________
Loren Nolte

___________________________
John Board

___________________________
Galen Reeves

Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Electrical and Computer Engineering in the Graduate School of Duke University

2015
ABSTRACT

Statistical Models for Improving the Rate of Advance of Buried Target Detection Systems

by

Jordan M. Malof

Department of Electrical and Computer Engineering
Duke University

Date:_______________________

Approved:

Leslie Collins, Supervisor

Kishor Trivedi

Loren Nolte

John Board

Galen Reeves

An abstract of a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Electrical and Computer Engineering in the Graduate School of Duke University

2015
Abstract

The ground penetrating radar (GPR) is one of the most popular and successful sensing modalities that have been investigated for buried target detection (BTD). GPR offers excellent detection performance, however, it is limited by a low rate of advance (ROA) due to its short sensing standoff distance. Standoff distance refers to the distance between the sensing platform and the location in front of the platform where the GPR senses the ground. Large standoff (high ROA) sensing modalities have been investigated as alternatives to the GPR but they do not (yet) achieve comparable detection performance. Another strategy to improve the ROA of the GPR is to combine it with a large standoff sensor within the same BTD system, and to leverage the benefits of the respective modalities. This work investigates both of the aforementioned approaches to improve the ROA of GPR systems using statistical modeling techniques. The first part of the work investigates two large-standoff modalities for BTD systems. New detection algorithms are proposed in both cases with the goal of improving their detection performance so that it is more comparable with the GPR. The second part of the work investigates two methods of combining the GPR with a large standoff modality in order to yield a system with greater ROA, but similar target detection performance. All proposed statistical modeling approaches in this work are tested for efficacy using real field-collected data from BTD systems. The experimental results show that each of the proposed methods contribute towards the goal of improving the ROA of BTD systems.
Contents

Abstract ........................................................................................................................................... iv

List of Tables ...................................................................................................................................... ix

List of Figures ................................................................................................................................. x

Acknowledgements .......................................................................................................................... xxi

1. Introduction ................................................................................................................................... 1

2. Background ..................................................................................................................................... 9

   2.1 BTD overview ............................................................................................................................. 10

   2.1.1 Typical detection processing ................................................................................................. 10

   2.1.2 Scoring: the receiver operating characteristic curve ......................................................... 12

2.2 Anomaly Detection Algorithms - Prescreeners .......................................................................... 17

   2.2.1 The multi-scale blob detector ............................................................................................... 17

   2.2.2 The RX anomaly detector ..................................................................................................... 18

2.3 Classification Algorithms ............................................................................................................ 23

2.4 The Mean-Shift Algorithm .......................................................................................................... 25

2.5 Geometric Camera Modeling ....................................................................................................... 27

   2.5.1 The Pinhole Camera Model ................................................................................................... 27

   2.5.2 The Perspective Projection .................................................................................................... 30

   2.5.3 World-Space to Image-Space Projections ........................................................................... 32

2.6 Queuing theory background ......................................................................................................... 34

   2.6.1 Irreducible homogenous continuous-time Markov Chains ................................................ 37

2.7 Summary ..................................................................................................................................... 39
3. Seismo-acoustic detection ................................................................. 41
   3.1 Sensing methodology and detection challenges .......................... 41
   3.2 The seismo-acoustic dataset ..................................................... 44
   3.3 Seismo-acoustic background methods ...................................... 45
      3.3.1 Data preprocessing .......................................................... 45
      3.3.2 Fourier features ............................................................... 48
      3.3.3 Classification with human observers .................................. 50
   3.4 Spatiotemporal keypoint clustering ......................................... 52
   3.5 Experimental results & discussion .......................................... 55
   3.6 Conclusions ............................................................................ 61

4. Detection with a forward-looking infrared camera ....................... 64
   4.1 Background ............................................................................. 65
      4.1.1 The multi-look property of forward-looking infrared data ...... 67
      4.1.2 The baseline detector: Mean-shift RX ................................ 69
      4.1.3 Overview of this work ....................................................... 71
   4.2 The forward-looking infrared dataset ..................................... 73
   4.3 Experiment 1: Establishing the benefit of multi-look ............... 73
      4.3.1 Defining a look ................................................................. 74
      4.3.2 Isolating individual looks in the FLIR data ...................... 75
      4.3.3 Experimental design ......................................................... 79
      4.3.4 Results and discussion .................................................... 81
   4.4 Improving multi-look processing ............................................ 83
4.4.1 The plan-view RX algorithm ..........................................................84
4.4.2 The forward-looking RX algorithm ..............................................87
4.4.2.1 The forward-looking RX algorithm .........................................89
4.4.2.2 Creating FLRX filters ...............................................................91
4.5 Experiment 2: testing the proposed algorithms .................................98
4.6 Conclusions ..................................................................................100
5. Fusion of forward-looking infrared and ground penetrating radar .......102
5.1 Data ............................................................................................103
5.1.1 GPR Data ................................................................................104
5.1.2 FLIR Data ................................................................................104
5.2 Background ..................................................................................105
5.2.1 The FLIR Prescreener ...............................................................105
5.2.2 GPR Edge Histogram Descriptors ..............................................107
5.3 The FLIR-GPR Detector .................................................................108
5.4 Experimental Results & Discussion ..............................................113
5.5 Conclusions ..................................................................................119
6. A new multi-sensor management strategy for buried target detection systems ....122
6.1 A queuing model for a 2-sensor detection system .................................125
6.1.1 Derivation of the state transition rates .......................................128
6.1.2 Computation of ROA .................................................................131
6.2 Validating the sensor management strategy and queuing model ..........132
6.2.1 Dataset description .................................................................133
6.2.2 Detection algorithms.................................................................................. 134
6.2.3 Detection system experimental settings..................................................... 134
6.2.4 Experiment 1: validity of the queuing model.......................................... 135
6.2.5 Experiment 2: validity of sensor management strategy .......................... 139
6.3 Conclusions ................................................................................................. 147
7. Conclusions and future work......................................................................... 149
  7.1 Summary of contributions and future work.............................................. 151
Bibliography ..................................................................................................... 154
Biography .......................................................................................................... 160
List of Tables

Table 1: A tabulation of the area under the ROC curve (AUC) measured for the human observers, the SKC algorithm, and the two best human-informed Fourier features. Although the SKC algorithm doesn’t perform as well as the human observers, it performs better than the established Fourier features, and does so without making use of any human-provided information about the spatial location of the target, as is the case with the Fourier features. 61

Table 2: Equations for the FLRX algorithm. 97

Table 3  A table of the values of physical system parameters that are used in the experiments presented in this chapter. Values were chosen to be representative of what might reasonably be encountered in practice. 135
List of Figures

Figure 1.1. This is an illustration of a multi-sensor BTD system with two sensors. Sensor A is a large standoff sensor and therefore yields a higher rate of advance (ROA). This is similar to a forward looking GPR BTD system or a forward looking infrared (FLIR) BTD system. Sensor B has a short standoff and lower ROA: similar to a GPR BTD system. ....4

Figure 2.1. This figure shows a flowchart of a common BTD processing flow. Data from a sensor, often in the form of a volume, is fed to a prescreener (left). Prescreeners consist of a (computationally) fast detection algorithm that identifies individual locations in the data volume that are likely to contain a buried target, i.e., “alarms”. This reduces the amount of data that must be considered in the next step, feature processing, which is more computationally intensive but often yields better discrimination between true buried targets and false alarms. Feature processing (right) typically consists of extracting a variety of local statistics around each alarm location and using a classification algorithm to assign a decision statistic based on the data statistics............12

Figure 2.2. This figure provides an illustration of how detector alarms are assigned a label so that ROC analysis can be performed. The detection system output for a typical experiment consists of a set of locations on the earth, one for each alarm made by the system. These alarms are shown with black ‘x’ symbols. Each alarm must be labeled as a hit or a false alarm, indicating whether it refers to a true target or not. Usually the location of each buried object in the testing area is known in advance, and any alarm landing within a predefined radius, or halo, around a target is considered a hit. The target locations and their halos are shown in red on the figure. All alarms landing outside any halo are considered false alarms, as shown. In this example, there are a total of 3 hits, and 4 false alarms, 2 of each are labeled in the figure. ........................................13

Figure 2.3. An illustration of an ROC curve is shown. The area used to compute AUC is shown in light red and the area used for the pAUC measure is shown in darker red, between the values of $p_{FA0}$ and $p_{FA1}$. The ROC curve for chance, or random, detection is also shown along the diagonal.........................................................16

Figure 2.4. This figure provides an illustration of multi-scale laplacian-of-gaussian (LoG) processing. The left-most image shows a single frame of data where a blob-like target signature is visible. The image is convolved by several scaled LoG filters, shown in the middle, and then stacked as shown on the right side of the figure. Keypoints are obtained by choosing extrema in the data volume. These extrema yield a scale $\sigma_i$, a spatial location $(x, y)$, and a keypoint “confidence”, $L(x, y, \sigma_i)$........................................18
Figure 2.5. This figure shows an illustration of the RX detector applied for landmine detection in an image from a forward-looking infrared camera. The user chooses two window sizes: a target window (red) and a background window (black). At each location in the image, statistics are computed using the pixels within each window, respectively. The decision statistic for that pixel is computed based on the statistics computed from the windows (bottom right).

Figure 2.6. This figure illustrates the behavior of a linear classifier and a nonlinear classifier on a set of synthetic classification data. Each red dot corresponds to a datum from Class 1, while each green dot corresponds to a datum from Class 0. A Fisher Linear Discriminant would yield a solution in the form of a decision boundary (line) that bisects the feature space as shown in the figure (dashed black). A support vector machine can create a more complicated decision region (solid line). Although the SVM can provide more complex solutions, this can also lead to overfitting of the data. Therefore each of the two methods will work well under certain conditions. (This figure is a modified form of a figure from http://openclassroom.stanford.edu/MainFolder/DocumentPage.php?course=MachineLearning&doc=exercises/ex8/ex8.html)

Figure 2.7. This figure illustrates an example pinhole camera. A 3-dimensional cylinder (left) is imaged by the pinhole camera, and the resulting virtual image (2nd from left) and true image (right) are shown. The dot on the cylinder is shown to indicate the relative orientation of the object as it appears in each image.

Figure 2.8. This figure illustrates the major frames of reference in the pinhole camera model: world-space, image-space, and camera-space. The relative orientation and position of each frame of reference is indicated by the axes. It should be noted, however, that the world-space frame can take any orientation and shift relative to the other two frames.

Figure 2.9. This figure graphically illustrates how location coordinates are mapped from the world-space reference frame to the camera-space reference frame. In the camera-space reference frame (right), the camera is at the origin and pointing along the $i$-axis. The mapping from world-space to camera-space is determined by finding the mapping that moves the camera from its world-space location $(\pi x_w, \pi y_w)$ and orientation $(\theta w)$ to its camera-space location $0, 0$ and orientation $0 o$. The object on the left at $(x_w, yw)$ in world-space is located at $(xc, yc)$ after applying this mapping.

Figure 2.10. The panel on the left illustrates a simple queue model where jobs arrive at a station for processing. After processing there is some probability, $a$, that operation will halt for a short time and the queue will be cleared. If the job arrivals and departures
form Poisson processes with parameters $\lambda$ and $\mu$, respectively, then the queue can be represented equivalently as an irreducible homogenous CTMC (right-most panel). The \textit{ith} state of the CTMC has \textit{i} jobs in the queue, with the exception of the “Stop” state. Each state is indicated by a circle. The transition rates, $q_{ji}$ between states \textit{i} and \textit{j} (formally defined in the text) are indicated in the graphic with arrows and their values. This CTMC representation facilitates the use of many theoretical results that are used to derive expressions for queue properties. In Section 6.1 a 2-sensor BTD system is modeled as a queue (and CTMC), and expressions are derived for system characteristics, such as ROA.

Figure 3.1. An illustration of the seismic data collection system used in this study. An acoustic stimulus $a(t)$ is applied at a query location and the laser Doppler vibrometer (LDV) is focused at one particular spatial location $x, y$ on the ground where it collects a time-series of vibrational data at that location, $v_t, x, y$. This process is repeated several times at different spatial locations on a grid over a patch of earth. This results in the collection of a data volume over the query patch of earth (shown on the right). The red columns correspond to the time-series recorded at each spatial location by the laser vibrometer, shown on the left with corresponding red arrows. The transfer of energy from an acoustic waveform to vibrational energy (as measured by the LDV) is often modeled as a linear system that is completely characterized by an acoustic-to-seismic (A/S) transfer function, $h(t, x, y)$. Many existing detection methods rely on estimating $h(t, x, y)$ to detect the presence of a buried target.

Figure 3.2. An illustration of the preprocessing applied to each seismic data volume in the data collection. The seismic time series at each spatial location $v_t, x, y$ (two time series are indicated by the red columns in the center graphic) is cross-correlated with the stimulating acoustic signal, $a(t)$, yielding another volume where each time series is an estimate of the ground A/S transfer function at that location, $h(r, x, y)$. All detection methods operate on the processed data volume.

Figure 3.3. This figure shows the graphical user interface used in the human observer experiments. It allows the user to explore the preprocessed data volume with the slider at the bottom. Each image corresponds to all the spatial values at one instance in time. The user can record a rating at the bottom left indicating their level of confidence that a target is present. Some observations contained a bright fiducial object and these objects were removed from consideration with a large black box. An example is visible in the figure. This box was placed in every observation so it could not be used as a cue for the presence of a target. A red target box was also drawn in each observation by the authors. This box indicated where each human observer should look during testing. The target box was also placed in each observation whether it contained a target or not.
This red box was also used to control where the Fourier features were extracted in each observation.

Figure 3.4. This illustration depicts how target signatures only appear in relatively small portions of the preprocessed data volume. Most of the cross-correlation data is relatively uninformative (top right black image) while just a few images contain most of the informative target signature responses (bottom right). Targets tend to manifest as bright or dark blobs.

Figure 3.5. Each figure here plots the 10 strongest detected keypoints in the preprocessed data volume for a target observation (left) and a non-target observation (right). The size of each plotted point is proportional to the response of the LoG filter for those points. Two additional images are shown for each observation (on both the left and the right), corresponding to the images where the two strongest keypoints were detected for the target and non-target observations. The keypoint spatial location is also plotted in the subimages. In the target observation, the keypoints cluster tightly and have higher confidence. The blobs in the image data are clearly very strong in those frames of data. The non-target observation keypoints are further apart and are weaker, reflecting the fact that there is no target signature in the data, and no noticeable blob is observed in the subimages.

Figure 3.6. Plot of ROC curves for the human observers, the SKC algorithm, and the top two (human-informed) simple features. The human observers performed the best but not at all operating points: the SKC algorithm has better performance at lower false alarm rates. The simple Fourier features perform best at very high PD, but have high false alarm rates.

Figure 3.7. Here, two scatter plots are shown, each with a different combination of features. In the left-hand plot the two best Fourier features are scattered together. Targets are shown in red, non-targets in blue. The plot indicates that they are highly correlated, which is not surprising since they correspond to physically similar phenomenon. It is unlikely that fusing these would be beneficial. The right-hand plot shows the SKC algorithm along with one of the phase features. The black circles indicate non-target observations where one, but not both, of the two features indicates a high probability of a target. These features may yield better performance through fusion.

Figure 3.8. This figure is a plot of ROC curves for two detection algorithms along with their fusion results. The SKC algorithm is shown along with the best Fourier feature, Phase:155Hz. The output of these algorithms was fused using a linear classifier (FLD Fusion) and a nonlinear classifier (SVM Fusion). The results are shown in the dashed
and dotted lines, respectively. The fused decisions perform better over some parts of the ROC curve and worse in others.

Figure 4.1. This figure illustrates the way data is collected using a FLIR camera. The FLIR camera is typically mounted on a vehicle in a forward looking configuration as shown on the left and collects video frames as the vehicle drives down the road. A target is indicated by the blue circle in the ground. The data consists of a sequence (video) of infrared images as shown on the right. In the top right there is an example infrared image with a box around a target signature. The target appears as a bright circular disk.

Figure 4.2. An example of multi-look information that is present in FLIR video. A bright blob corresponding to a target appears in different locations of the video frames (image-space, left) however all points correspond to roughly the same location on the earth (world-space, right). The black boxes illustrate the notion of multiple looks at the object. Notice that the object appears at a different size and shape in each look (image).

Figure 4.3. This figure illustrates the basic operation of the baseline FLIR detector used in the work presented in this Chapter. It begins with a standard RX detector (left) which yields a list of alarm locations in image-space (center). It should be noted that the RX filter is characterized by a single target filter size and single background filter size which is used across all images and camera perspectives. The alarm coordinates in image-space are then mapped into world-space (GPS) coordinates where they are clustered using the mean-shift algorithm. The alarms output by the algorithm consist of the cluster centers found by mean-shift. The decision statistic for each cluster is computed by summing the decision statistics of its member alarms. In this way, the mean-shift algorithm attempts to exploit multi-look information available in the FLIR data to improve performance.

Figure 4.4. This figure illustrates the method used for isolating a single look at the lane. As the vehicle moves forward from time \( t \) to \( t + 1 \), the highlighted region of the ground passes under the camera. If this part of the image is retained at every time, it guarantees a complete look at the lane.

Figure 4.5. This figure provides an illustration of the regions used from the FLIR video frames to construct disjoint looks at the lane. There are eleven total regions shown here, although more can be made. The FLIR video data (left) is partitioned into 11 collections of sub-images. One collection of sub-images corresponds to taking a sub-image from every frame of data that is the size of Region 1. If this operation is performed across all images in the video, it results in one look at the lane.
Figure 4.6. This figure provides an illustration of the Depth Index parameter $k$. If $k = K$ then the regions from 1 up to $K$ are used for detection. Broadly speaking, the value of $k$ corresponds to the number of looks utilized for detection.

Figure 4.7. This figure shows the pAUC scores for RX (solid) and MSRX (dashed) algorithms on the FLIR data. Performance was computed as a function of the number of looks that were used, indicated by the Depth Index parameter, $k$.

Figure 4.8. This figure shows some of the results of plan-view processing on the FLIR dataset. Each of the eleven disjoint FLIR video regions (left) is used to construct a unique plan-view (right). These plan-views are then used for detection.

Figure 4.9. This figure provides an illustration of a pixel in image-space. A pixel is a square subset of the real image-space plane, $Pixyt$, where $(x, y)$ is the center location of the pixel and $t$ is the video frame in which the pixel exists.

Figure 4.10. This figure illustrates the process of mapping a pixel from image-space (left) to world-space (right). This is performed using the perspective projection operators described in Section 2.5.3.

Figure 4.11. This figure illustrates the process of creating the FLRX filter for the pixel located at $(x, y)$. The pixels surrounding $x, y$ are first mapped into world-space. The filter weight assigned to each mapped pixel corresponds to the amount of intersection between the pixel and the world-space target filter $Twxyt$. The final filter is shown on the bottom right.

Figure 4.12. This figure shows some examples of background and target FLRX filters created on the FLIR dataset. The locations of the filters are shown in the left pane of the FLIR image. They are labeled “one”, “two”, and “three” respectively.

Figure 4.13. This figure is a plot of the pAUC values computed for three detection algorithms as a function of the depth index parameter: MSFLRX (black), MSPRX (red), and MSRX (blue).

Figure 5.1. This figure shows an outline of the FLIR detection algorithm used in this work. An ensemble of size-contrast filters are used to filter the images and detect alarms in each frame (illustrated on left). Within each image these alarms are clustered into groups (center), and each group, indicated by a black dashed circle, is a new single image-space alarm. These alarms are then mapped from image-space (pixel coordinates) into world-space (UTM coordinates). Once mapped into world-space, the alarms are again clustered to create new alarms (right). These cluster centers are the final alarm locations designated by the FLIR detector.
Figure 5.2. This figure shows a GPR target signature (top) and how it is partitioned for EHD feature extraction. The signature is partitioned into several sub-images corresponding to each of the black boxes shown. Image gradients are extracted within each sub-image and placed into one of the 5 following categories: no edge, vertical, horizontal, diagonal, and anti-diagonal. The final EHD feature is a concatenation of the histogram for each sub-image as shown in the bottom panel. Note that the sub-images here are shown to be disjoint for simplicity, but in actuality they overlap by 50%.

Figure 5.3. This figure illustrates the difference between a conventional GPR data processing scheme (top panel), and the approach proposed in this work (bottom panel). In the conventional scheme, the GPR for a given query location is only processed once the GPR sensor has moved beyond the desired query location. This is illustrated in the left-most graphic in the top panel, where the query location is indicated with the gray circle. The subsurface data utilized for subsequent processing is highlighted in yellow, and the corresponding GPR data is shown in the right-most graphic. In the proposed processing scheme (bottom panel), the algorithm processing begins sooner: before all the data is collected. This increases the effective standoff distance, as illustrated by the green rectangles in the left-most graphic. However, it also yields less data for the subsequent detection processing, as indicated in the middle graphic. This reduction in available data may be detrimental to detection performance.

Figure 5.4. This figure provides an outline of the experiments conducted in this study. The FLIR and GPR testing data are shown on the left as input. The FLIR data is prescreened using the prescreener described in Section 5.2.1. This yields alarm locations (bottom left) where GPR EHD features are extracted from the GPR testing data (center). Several individual classification experiments are run, each using a different number of EHD feature sets: ranging from 1 set to 7 sets. Changing the number of sets used corresponds to changing the system standoff distance. These testing features are then classified using an SVM and then scored (right). The goal of the experiments is to illustrate the tradeoff between increasing standoff distance and detection performance.

Figure 5.5. This figure shows ROC curves for the different detectors that were applied to the FLIR-GPR dataset. The dashed lines show the performance of the two prescreeners. The black dashed line shows the performance of a set of alarms detected by a standard GPR prescreener on GPR data. The dashed red line shows the performance of a set of alarms detected by a standard FLIR prescreener on the same dataset using FLIR data. The 4 solid lines show the performance of an SVM classifier trained on EHD features extracted at the FLIR alarm locations. Each line corresponds to the performance achieved using some subset of the available EHD features at that location. As more EHD features are used, the performance increases as expected. However, there is a large
disparity between the best performance achieved by the EHD features extracted from the FLIR alarms and the performance of the GPR prescreener.

Figure 5.6. This figure compares the performance of all detection algorithms. The performance metric (y-axis) is pAUC, which is computed between a FAR of 0 and a FAR 0.02 $\text{FA/m2}$. The blue line corresponds to the performance of the GPR EHD features as standoff distance is increased. The left-most blue point corresponds to using all 7 sets of EHD features, while the right-most blue point corresponds to using just a single EHD feature set. Standoff distance increases as fewer EHD features are used, at the cost of lower detection performance (as measured by pAUC). However, the performance loss incurred from removing each EHD set is very low for the first several sets, making it feasible to increases in standoff with the system. The black dashed line is the performance of the FLIR prescreener as a stand-alone detector. A major limiting factor of the FLIR-GPR system overall is the performance disparity between existing GPR prescreeners and full FLIR detection systems. The red dashed line is the performance of a standard GPR prescreener as a stand-alone detector. Note that the FLIR prescreener has a very large added standoff distance (off the plot on the right side), while the GPR prescreener has 0 cm added standoff.

Figure 5.7. This figure shows the alarm locations for the GPR and FLIR prescreener plotted on the testing lane, along with target locations. Black circles indicate alarms from the GPR prescreener, red circles indicate FLIR detector alarms, and blue unfilled circles represent target locations. This illustrates how many more false alarms the FLIR prescreener incurs as compared to the GPR prescreener. This is despite the fact that the FLIR prescreener is operating at a 90% probability of detection while the GPR prescreener is operating at near 100%. This performance gap must ultimately be bridged in order for a large standoff FLIR-GPR system, like the one proposed here, to be a practically feasible detection system.

Figure 6.1. The panel on the left illustrates a 2-sensor BTD system that combines a large standoff FLIR detector with a short standoff GPR. The FLIR operation benefits from large standoff and high ROA, while the GPR operation mode has much better detection performance (e.g., a high $P_d$, given a fixed FAR) and a lower ROA. The panel on the right illustrates a new sensor management strategy where the system ROA is high with the FLIR only (GPR deactivated) until a suspicious location is identified by the FLIR BTD system, at which point the vehicle slows down and activates the GPR system to achieve better detection performance.

Figure 6.2. This figure illustrates the performance dynamics of the proposed sensor management strategy, in this case operating with a forward looking infrared (FLIR) camera and a downward-looking ground penetrating radar (GPR) system. FLIR and
GPR data are fed to the system (left) and each detection system is operated at a certain operating point on their respective operating curves. The FLIR system operates at probability and detection and false alarm of $P_d1$ and $P_f1$ respectively (shown in red). The alarms from the FLIR are fed to the GPR (operating point shown in blue). The overall system metrics are shown on the far right. Overall probability of detection, $P_d$, and probability of false alarm, $P_f$, are computed on the right. The system false alarm rate is computed by multiplying the $P_f$ by the FLIR alarm rate, $R_FA$, which is the number of alarms detected per unit of distance driven. The other metric of interest here is the rate of advance, ROA, but it is unclear how to compute it because system velocity changes over time.

Figure 6.3. The left-most illustration shows the FLIR-GPR system with several important processing quantities labeled. The right-most illustration depicts the corresponding birth-death model for the FLIR-GPR system, along with the arrival and processing parameters that were derived in Section 6.1.1. In this model, alarms from the FLIR detector represent jobs arriving to the queue, which is the GPR feature processing stage. The rate at which alarms arrive to the GPR from the FLIR detector depends on the false alarm rate of the FLIR ($FAR_1$, in false alarms per meter driven) and the velocity at which the system is moving down track in each state, $v_i$. The job processing rate is based on how quickly the GPR algorithms require to process each alarm, $T_{GPR}$.

Figure 6.4. In the left-most panel is a histogram of the FLIR alarm inter-arrival times, assuming a velocity of 4.2m/s. An assumption of the proposed BTD queuing model is that the distribution of the alarm inter-arrival times is exponential. A qualitative assessment of this assumption is made by fitting (via maximum likelihood estimate) an exponential distribution to the histogram (red). This shows that, while the exponential assumption is not perfect, it appears to be reasonable. Another assumption of the birth-death queuing model is that any two alarm inter-arrival times are independent of one another. On the right-most panel is a 2-D histogram showing the joint distribution between alarm inter-arrival given by $A_t$, and the inter-arrival times immediately preceding $A_t$, given by $A_t - 1$. This figure shows that any two consecutive inter-arrival times (i.e., $A_t$ and $A_t - 1$) are correlated, however this correlation is not very strong. This is indicated by the Pearson correlation coefficient shown in the title, along with its p-value.

Figure 6.5. These figures provide scatter plots of predictions made by the FLIR-GPR queuing model and the corresponding values estimated by simulation of the FLIR-GPR system using real data. The left-most plot is the most important. It illustrates the agreement between the model predictions of the system ROA and those obtained from simulations. This is quantified by the Pearson correlation coefficient and corresponding p-value in the title. The four smaller plots on the right illustrate some disagreement of
state probability predictions made by the model, and those from simulation. As the state index increases, the queuing model consistently underestimates the amount of time (by proportion) that the system spends in that state. This is likely due to the linear correlation between alarm inter-arrival times. Alarms tend to come in groups and cause the queuing model to more frequently occupy larger-index states than would be predicted by the queuing model because it assumes independent inter-arrival times.

Figure 6.6. The left-most panel of this figure illustrates the impact of $P_d1$ and $P_d2$ on the overall system ROA, for the specific case when the system detection sensitivity, $P_d = 0.65$. A much better ROA (red circle) can be obtained by properly choosing $P_d1$ & $P_d2$. The right-most panel shows the relationship between overall system detection sensitivity and ROA for three different systems: the FLIR-GPR system (red curve), a stand-alone FLIR system (blue curve), and a stand-alone GPR system (black curve). The results show that the FLIR-GPR system offers better ROA than the stand-alone FLIR for all detection sensitivities, but that the GPR offers better ROA at high sensitivities. This is because the FLIR-GPR system is fundamentally limited by the performance of its FLIR detector, but in principle, the FLIR-GPR can achieve better ROA even at high sensitivities if the FLIR detector (or any other large standoff detector) used in the system achieves better detection performance.

Figure 6.7. This figure expands on Figure 6.6, and shows the relationship between ROA and System $P_d$, as well as the system false alarm rate (given in false alarms per meter driven) for the FLIR system and the SMS System (i.e., FLIR-GPR). As the system $P_d$ increases the false alarm rate increases, and consequentially lowers the system ROA by increasing the amount of time spent driving slowly, or stopped.

Figure 6.8. This figure shows the relationship between the false alarm rate and the probability of detection of the SMS system (i.e., FLIR-GPR) and the standalone FLIR system. Each point, on each ROC curve, corresponds to a specific system ROA (in meters per second), which is written in black. Recall that the SMS system was optimized to achieve the best ROA for a given $P_d$. Therefore the SMS system (for these experimental conditions) always achieves a greater ROA than the standalone FLIR system for any choice of system $P_d$.

Figure 6.9. The leftmost panel shows the relationship between the FLIR $P_d$ (i.e., $P_d1$) and the GPR $P_d$ (i.e., $P_d2$). Each contiguous blue/green contour has a constant system $P_d$. The red line shows the values of $P_d1$ and $P_d2$ that optimize the system ROA for each choice of system $P_d$. The best balance between $P_d1$ and $P_d2$ changes depending on the desired overall system $P_d$. The right-most panel shows the system ROA as a function of the GPR false alarm rate (FAR), and the FLIR FAR. Note that any choice of $P_d1$ and $P_d2$ (as in the left-most panel) corresponds to a choice of GPR FAR and FLIR
FAR, respectively. This figure shows specifically how the system performs as ROA increases. Increases in FLIR FAR correspond to more frequent system slowdowns, while increases in GPR FAR correspond to more frequent stops.
Acknowledgements

This work was supported by the U.S. Army RDECOM CERDEC Night Vision and Electronic Sensors Directorate, via a Grant Administered by the Army Research Office under Grant W909MY-12-C-0044. A special thanks to the lab of Dr. James Keller at the University of Missouri for support with the infrared camera dataset.
1. Introduction

Buried explosive threats, such as landmines and improvised explosive devices (IEDs), impose enormous costs wherever they exist. They are indiscriminant killers, capable of killing and maiming any individual regardless of whether they are a combatant or a civilian. In 2012 there were an estimated 3628 mine-related injuries or deaths, of which nearly 50% were children [1]. Often landmines are placed during periods of war left behind after the conflict resolves, leaving the burden on local civilian populations. The situation is often exacerbated further because many humanitarian agencies will not operate in contaminated regions [2].

Because buried threats are such a costly and deadly problem, there is great interest in finding and removing them. However, this is a very difficult task. One class of solutions involves detecting buried threats at a distance through the use of remote sensing. Such remote sensing systems are usually mounted on a vehicle and collect data from the ground as the vehicle moves forward. As data are collected they are processed by detection algorithms: computer software that processes the data and alerts the vehicle operator if it determines there is a high probability of the presence of a buried threat. Such a system is referred to here as a buried target detection (BTD) system. Note that we use the term “target” in place of “threat” in the BTD acronym because BTD systems, and the work developed here, are often applicable for detecting a
variety of subsurface objects. The term target is therefore used to refer to any subsurface object of interest in a subsurface detection setting.

A substantial number of sensor modalities have been studied for BTD, including ground penetrating radar, electromagnetic induction, infrared imaging, seismo-acoustic sensing, and more [3]–[8]. Despite the variety of remote sensing technologies available, demining operations are still costly procedures. The cost to safely detect and remove a landmine is estimated to range from $300 to $1000 [9]. One major contributor to this cost is false alarms. A false alarm occurs when a buried target is thought to exist at a particular location, but upon excavation, it is discovered that there is not a landmine or IED there. Excavations are a costly and dangerous procedure, and each false alarm results in a needless excavation. Because of this relationship between false alarms and cost, a key metric for any BTD system is its false alarm rate (FAR) (given a fixed probability of detection, termed $P_d$). FAR refers to the number of false alarms that can be expected for each unit of area that is searched (e.g., squared meters). A higher FAR indicates more needless excavations of the ground, raising the costs of remediation and slowing its progress. Handheld electromagnetic induction detectors traditionally used by dismounted troops or non-governmental organizations for land mine detection, such as the PSS-12 and the ANPSS-14, suffered from very high false alarm rates because, in addition to sensing metal in landmines, they also sense all the metallic debris leftover from military activities [10], [11].
Probably the most successful contemporary remote sensing detection modality for vehicular-based sensing, in terms of high Pd and low FAR, is downward looking ground penetrating radar (GPR) (e.g., [12]–[14]). This has made downward-looking GPR the focus of numerous BTD studies and algorithm development efforts [5], [15]–[18]. One of the major drawbacks of GPR however is its low rate of advance (ROA) due to its short detection standoff distance [8], [15], [19], [20]. ROA refers to the expected (or average) system velocity over time. This measure often refers simply to the velocity of the vehicle while actively demining, but can also include the time required to stop for false alarms. The latter definition is employed in this work. Standoff distance (or standoff) refers to the distance between the vehicle wheels, on which the GPR is mounted, and the location in front of the vehicle where the GPR senses the ground. The short standoff of the GPR is illustrated below in Figure 1.1 with two sensors: one with a short standoff (sensor B) and one with a large standoff (sensor A). If the standoff is small, then the vehicle ROA must be sufficiently low so that, in the event of an alarm, the system operator can stop the vehicle before over-running the alarm location. This is the case for the GPR, where its short standoff results in a low ROA.

Due to the short standoff (and ROA) of the downward-looking GPR there has been substantial interest in developing a larger standoff alternative to GPR. Several modalities have been investigated in the literature for this purpose: seismo-acoustic sensing (e.g, [21]), forward looking infrared cameras (e.g. [8], [19]), and forward looking
GPR (e.g. [22]). The problem with these alternatives, however, is that their detection performance (e.g., $P_d$, given a fixed FAR), as yet, are much too low to make them suitable replacements for GPR. As a point of clarification, the term GPR will always refer to downward-looking GPR, and FLGPR will denote forward-looking GPR.

![Illustration of a standoff and rate of advance for two sensors](image)

**Figure 1.1.** This is an illustration of a multi-sensor BTD system with two sensors. Sensor A is a large standoff sensor and therefore yields a higher rate of advance (ROA). This is similar to a forward looking GPR BTD system or a forward looking infrared (FLIR) BTD system. Sensor B has a short standoff and lower ROA: similar to a GPR BTD system.

One direct way of addressing the performance limitations of large standoff sensors is to improve them via the development of improved signal processing and machine learning methods. Research across many BTD modalities over the last two decades has provided substantial evidence that improved signal processing and machine learning can lead to much better performance (e.g., [5], [23], [24]). This strategy is the basis of the first half of the work presented in this dissertation (Chapters 3 and 4). Algorithms based on machine learning and signal processing were developed to
improve detection performance (e.g., \( P_d \), given a fixed FAR) for two large standoff sensing modalities: the forward-looking infrared camera (the FLIR), and for a seismo-acoustic sensing system. Experiments are presented that employ real field collected BTD system data to examine the efficacy of the proposed methods. The results indicate that the proposed methods are effective for improving BTD detection, but that the performance for both of the BTD modalities investigated here, and for large standoff sensors in the literature in general, still falls far short of the performance of the GPR.

As BTD algorithms continue to improve over time, large standoff sensors may ultimately become suitable high ROA replacements for GPR, however this does not appear to be a viable solution to improve GPR ROA in the near future. Therefore, in the second part of this dissertation (Chapters 5 and 6), another general approach is investigated. This approach involves combining a large standoff sensor (higher ROA and poorer detection performance) with a GPR (lower ROA and higher detection performance) on a single detection platform so that it can leverage the advantages of each individual BTD sensor, yielding a system that can obtain detection performance (e.g., \( P_d \), given a fixed FAR) and ROA that are not obtainable by either of the two individual BTD systems if they are operated independently. Two methods are developed and investigated in this work to achieve this goal.

The first multi-sensor approach investigated is similar to classical sensor fusion which has been previously studied in the BTD literature [20], [25], [26]. In the sensor
fusion context, several sensors are placed on a single BTD platform and they are operated continuously and simultaneously. Detection is performed in such a system by combining (e.g., using classifiers) the data, features, or detection decisions from each BTD modality/system in order to improve overall detection performance (e.g., $P_d$, given a fixed FAR). While fusion often improves detection performance, the velocity of the system is limited by the slowest sensor (i.e., in this case the GPR) because all sensors must be operating at all times. The approach proposed in this work applies a strategy that uses a large standoff BTD system (in this work a FLIR system is always used) to help increase the standoff of the GPR system. The proposed approach is described in detail in Chapter 5, where experiments are conducted using real field-collected FLIR and GPR data. The results show that the approach effectively increases standoff, but only at the expense of detection performance. This limitation serves to motivate the work presented in Chapter 6.

In Chapter 6 a second multi-sensor management approach is proposed that is designed to overcome the limitations of the approach presented in Chapter 5. The primary cause of the limitation of the approach in Chapter 5 is that, as discussed, all sensors must operate at all time, and this limits the velocity of the multi-sensor system during operation. The multi-sensor system is limited by the ROA of the GPR system. In the new approach, a sensor management strategy is developed where the GPR remains inactive and is only activated when certain operating conditions are encountered.
During time when the GPR is inactive, the system velocity can be increased and this improves the overall system ROA. This strategy, while potentially very effective, introduces many challenges. One important challenge is that the proposed sensor management strategy makes it much more difficult to compute system ROA because the system velocity varies over time. To solve this problem, a new probabilistic model, based on queuing theory, is derived that makes it possible to quickly estimate ROA for such systems, and thereby facilitate practical use of the proposed strategy.

To recapitulate, this dissertation presents several scientific contributions to the science of BTD and BTD systems, with the specific aim of contributing to the development of a BTD system that achieves greater ROA than contemporary GPR-based systems, while still maintaining effective detection performance (e.g., $P_d$, given a fixed FAR). Two primary research strategies are adopted to achieve this goal: to improve the detection performance of large standoff (high ROA) BTD sensors so that they may replace GPR systems, and to combine the GPR system with a large standoff alternative so that the resulting multi-sensor system can achieve the aforementioned goals. With these research strategies, the following contributions are made in Chapters 3, 4, 5, and 6 respectively:

- Algorithms are developed for a seismo-acoustic and FLIR BTD systems to improve detection performance (as measured by ROC analysis)
A BTD system is developed that fuses a FLIR and GPR sensor to improve the standoff of the GPR.

A sensor management strategy and probabilistic model are developed for combining large-standoff and short standoff BTD sensors on a single BTD system in such a way that it leverages their respective advantages: detection performance (e.g., a high $P_d$, given a fixed FAR) and ROA.

The organization of the remainder of this dissertation is as follows. Chapter 2 explores some of the pre-existing ideas and methods required to develop the proposed algorithms in this work. This includes a review of existing detection algorithms, detection scoring metrics, classification algorithms, and camera models. The next two chapters, Chapter 3 and Chapter 4, present the algorithms developed for the seismo-acoustic and FLIR BTD systems, as well as their performance results on real field-collected sensor data. Chapter 5 presents the multi-sensor BTD system that fuses a FLIR and GPR system. Results are presented showing the improved standoff distance over a stand-alone GPR system. Chapter 6 presents a development of the new BTD sensor management strategy and its application to real multi-sensor system involving a FLIR and GPR detection system. Finally, Chapter 7 discusses the conclusions and discusses potential future work.
2. Background

This chapter provides a brief review of the essential concepts required to develop
the methods and algorithms presented in subsequent chapters. In Section 2.1 an
overview of conventional BTD processing and algorithm testing is provided. This
includes a description of the major detection processing steps and terminology used in
the literature and throughout the remainder of this document. It also discusses how
detection experiments are conducted and what performance metrics are used to
compare algorithms. The work presented in each chapter uses these conventions except
where otherwise specifically stated. Section 2.2 reviews two major anomaly detection
algorithms: multi-scale blob detection [27] and the RX detector [28]. These two
algorithms will be used as the basis for new detectors proposed for the seismo-acoustic
and forward-looking infrared camera sensing modalities, respectively. Section 2.3
provides a brief description of supervised classification and describes two representative
algorithms that are used throughout this work. In Section 2.4 the mean-shift algorithm
[29] is described as a method that will be used for unsupervised data clustering. Mean-
shift will be employed as an important part of several algorithms used or developed in
this work. The last section, Section 2.5, describes camera models and perspective
transformations. This will be a very important concept for processing FLIR data, and for
developing the forward-looking RX algorithm described in Chapter 4. Finally, Section
2.6 discusses queuing theory, which is used to derive a probabilistic model for the BTD sensor management strategy proposed in Chapter 6.

2.1 **BTD overview**

Within the BTD literature there is no universally established experimental procedures or nomenclature, however, there are some widely used conventions. These conventions are used in this work, unless otherwise noted, and are reviewed in this chapter.

2.1.1 **Typical detection processing**

Data processing in BTD is often divided into two distinct processing steps: prescreening and feature processing (e.g., [5], [30]). This processing flow is outlined below in Figure 2.1. Often the raw or preprocessed set of sensor data is very large and it is difficult to apply sophisticated processing on all of it in a real-time setting (e.g. as the vehicle moves down a road). Prescreening is a processing step that relieves this bottleneck by using a fast detection algorithm to quickly discard most of the data and to identify specific locations considered to have a high likelihood of containing a target. More advanced processing (i.e., feature processing) can then be applied to these specific locations, dramatically reducing the computational load. Because they are computationally much more efficient however, prescreeners are usually less sophisticated and therefore less able to discriminate target and non-target data. In order to ensure a high likelihood of identifying most of the true targets, the prescreener
is operated with a high sensitivity at the cost of yielding many false alarms. It is then
the job of the feature processing step to perform more advanced processing and reduce
the number of false alarms.

Feature processing typically consists of extracting local statistics, or “features”,
around each prescreener alarm location and then applying a supervised classification
algorithm to assign a decision statistic to the location under consideration. It should be
clear that because feature processing only operates on prescreener alarms, it cannot
improve the probability of detection $P_d$ of the overall system. If some targets were
missed by the prescreener then they are not available for consideration in the feature
processing step. However, feature processing can dramatically reduce the number of
false alarms while missing very few of the true targets that exist among the prescreener
alarms [ref]. For this reason, feature processing is also sometimes referred to as false
alarm reduction.

This two-stage processing framework is common in BTD, especially for the GPR
modality, however, it is not universal. For example, it will not be an important concept
in Chapter 4 because that chapter is focused on prescreeners and no feature processing is
considered. However it will be very important with the multi-sensor GPR-FLIR BTD
systems presented in Chapters 5 and 6, respectively.
Figure 2.1. This figure shows a flowchart of a common BTD processing flow. Data from a sensor, often in the form of a volume, is fed to a prescreener (left). Prescreeners consist of a (computationally) fast detection algorithm that identifies individual locations in the data volume that are likely to contain a buried target, i.e., “alarms”. This reduces the amount of data that must be considered in the next step, feature processing, which is more computationally intensive but often yields better discrimination between true buried targets and false alarms. Feature processing (right) typically consists of extracting a variety of local statistics around each alarm location and using a classification algorithm to assign a decision statistic based on the data statistics.

2.1.2 Scoring: the receiver operating characteristic curve

As described in the previous section, the output of detection algorithms usually consists of a set of detector alarm locations and corresponding decision statistics. The objective of the receiver operating characteristic (ROC) curve is to concisely indicate how well the detector performed. ROC curves are a common metric for evaluating the performance of both detection and classification algorithms in the literature [31].

Before ROC analysis can be performed there is one additional processing step that must be performed on the detector alarms. Each alarm has to be labeled as to whether it corresponds to a true target or to a false alarm. Most BTD experiments,
including all the experiments presented here, are conducted so that the ground location of all targets is known in advance. With this information, each detector alarm can be labeled as a “hit”, or detection, if it is within some radius of a target location, referred to as a halo. Multiple alarms on the same target are usually merged into single alarm and the maximum decision statistic among the alarms is used to represent the group. All alarms that do not fall within a target halo are considered false alarms. An illustration of this labeling procedure is shown below in Figure 2.2. Once all of the alarms have been assigned as a hit or a false alarm, performance analysis can be conducted using a receiver operating characteristic (ROC) curve.

**Example detector output**

![Diagram of detector output showing two hits and two false alarms.](image)

Figure 2.2. This figure provides an illustration of how detector alarms are assigned a label so that ROC analysis can be performed. The detection system output for a typical experiment consists of a set of locations on the earth, one for each alarm made by the system. These alarms are shown with black ‘x’ symbols. Each alarm must be labeled as a hit or a false alarm, indicating whether it refers to a true target or not. Usually the location of each buried object in the testing area is known in advance, and any alarm landing within a predefined radius, or halo, around a target is considered a hit. The target locations and their halos are shown in red on the figure. All alarms landing outside any halo are considered false alarms, as shown. In this example, there are a total of 3 hits, and 4 false alarms, 2 of each are labeled in the figure.
With this processing complete, an ROC curve can be constructed for the given detector. The ROC curve is a plot of the tradeoff between the proportion of true targets that are detected by the detector and the number of false alarms that are incurred by the detector as a threshold, \( \gamma \), is raised or lowered on the decision statistic outputs from the detector. Each point on the ROC curve corresponds to one value of \( \gamma \). The location of the point in the ROC curve for that \( \gamma \) value is generated as follows: all the alarms with a decision statistic greater than \( \gamma \) are considered targets, and all others are discarded. Therefore some proportion of targets will be inadvertently discarded because their decision statistic was too low. Additionally, some number of false alarms will be included because their decision statistic is greater than \( \gamma \). Therefore, for each \( \gamma \), a detector can be assigned a probability of detection \( P_D \) and a probability of false alarm \( P_{FA} \). These two numbers indicate what proportion of the true targets and the total prescreener false alarms that were retained, respectively. The value of \( \gamma \) is varied continuously, generating all the sampled points for the ROC curve. An illustration of an ROC curve is shown below in Figure 2.3. An ideal detector will have one necessary point on the curve, at \( P_D = 1 \) and \( P_{FA} = 0 \). If the detector has no ability to discriminate between targets and non-targets then it will have a diagonal ROC curve. In practice, the \( \gamma \) value for a detector can be varied to achieve a desired operating point (a specific value of \( P_D \) and \( P_{FA} \)) based on how important it is to detect true targets versus incurring false alarms.
Often it is useful to compute a single statistic that summarizes the overall performance of a detector across all operating points. One such measure can be found by computing the area under the ROC curve [32], denoted by AUC. The AUC measure indicates the probability that two observations drawn randomly from each class of observations, respectively, will be properly ranked by their detector-assigned decision statistics. It is also often interpreted as percent correct.

Another similar measure is the partial area under the ROC curve [33], denoted here by pAUC. In this measure the area under the ROC curve is measured between two specific $P_{FA}$ levels, denoted here by $P_{FA0}$ and $P_{FA1}$. This statistic better summarizes the performance of a detector within a desired range of operation. An illustration of AUC and pAUC are shown in Figure 2.3 below. All pAUC measures are normalized by the total possible area between $P_{FA0}$ and $P_{FA1}$. The pAUC measure becomes significant in this work as we are often dealing with false alarm rate, and not $P_{FA}$. 
An illustration of an ROC curve is shown. The area used to compute AUC is shown in light red and the area used for the pAUC measure is shown in darker red, between the values of $P_{FA0}$ and $P_{FA1}$. The ROC curve for chance, or random, detection is also shown along the diagonal.

In many applications of ROC analysis the $P_{FA}$ values are converted into more meaningful physical units based on the application domain. One such example of this is in landmine detection where FARs are often scaled by the total number of alarms and divided by the total surface area ($m^2$) over which detection is performed [19], [34]. This results in a FAR measure in terms of false alarms per square meter ($\frac{FA}{m^2}$). The resulting ROC curves show performance in terms of $P_d$ and FAR, but are otherwise analogous to $P_D$ and $P_{FA}$. The remainder of this thesis will not distinguish between the two forms of the ROC curve.
2.2 Anomaly Detection Algorithms - Prescreeners

Anomaly detection algorithms are used to identify and localize a target signal that is embedded in a larger collection of (often) noisy data. Anomaly detectors are usually employed when a precise characterization of the target signal is unavailable. Rather than searching for the target signature, or features associated with a target signature, anomaly detectors detect data that is different, or anomalous, compared to the surrounding data. Although this is the main principle behind anomaly detection, some knowledge about the target is often exploited as well. These are usually very general characteristics such as the target’s average shape or scale. This section describes two popular anomaly detectors, multi-scale blob detectors and RX detectors, which are used as the basis for more advanced detectors proposed in subsequent section.

2.2.1 The multi-scale blob detector

Multi-scale blob detection is a popular technique that originated in the computer vision literature that was developed with the goal of detecting bright or dark blob-like objects in images [27], [35], [36]. It is well suited when the anomaly shape is unknown, or expected to be circular, and that it is expected to appear at multiple different sizes. This detector can exploit any knowledge about the range of sizes over which the anomaly is expected to appear by allowing the designer to specify a set of scales (or sizes) over which to search for anomalies. It will be employed in Chapter 3 as part of the keypoint clustering algorithm to find anomalies in seismo-acoustic data.
Multi-scale blob detection consists of convolving an input image with several Laplacian of Gaussian filters (see [37]) of varying scale, \( \sigma_i \). The scales are determined by the sizes at which target signals are expected to appear in the data. These filtered images are placed in a stack so that they collectively create a 3D block of data. This processing flow is illustrated below in Figure 2.4. Detector alarms, sometimes referred to as keypoints for this algorithm, are obtained by identifying extrema points in the 3D volume of data. These keypoints indicate a spatial location \((x, y)\), a scale \( \sigma \), and a “confidence” \( L(x, y, \sigma) \) at which a blob is detected.

\[
I(x, y) * \sigma_i \nabla^2 G(x, y, \sigma_i) = L(x, y, \sigma_i)
\]

Figure 2.4. This figure provides an illustration of multi-scale laplacian-of-gaussian (LoG) processing. The left-most image shows a single frame of data where a blob-like target signature is visible. The image is convolved by several scaled LoG filters, shown in the middle, and then stacked as shown on the right side of the figure. Keypoints are obtained by choosing extrema in the data volume. These extrema yield a scale \( \sigma_i \), a spatial location \((x, y)\), and a keypoint “confidence”, \( L(x, y, \sigma_i) \).

2.2.2 The RX anomaly detector

The RX anomaly detector is a popular and important anomaly detector in the multi-spectral image processing community [28],[38]–[40]. It is applied to find
anomalies at a single scale but allows a user-specified target signature shape to be defined. This is in contrast to the multi-scale blob detector, which is designed to find specifically blob-like shapes, but find them across multiple scales. RX was originally developed to detect anomalies in infrared and optical imagery and it is used for that purpose in this work in Chapter 4. The forward-looking RX algorithm and the integral image RX algorithms proposed in that chapter are based directly on the RX algorithm and it will be important to have a clear understanding of it. This section reviews the theory behind RX and how it is commonly employed in practice.

The RX algorithm is a probabilistic approach to anomaly detection that is motivated by the likelihood ratio test from the statistics literature. The likelihood ratio test is a statistical test that can be used to make optimal decisions about the true underlying model of some observed data, \( x \), given two competing hypotheses [41]. The two competing hypothesis, denoted by \( H_0 \) and \( H_1 \), are typically referred to as the null hypothesis and alternative hypothesis respectively. By the Neyman-Pearson theorem the optimal decision statistic is given by the likelihood ratio test

\[
\lambda(x) = \frac{p(x|H_1)}{p(x|H_0)},
\]  

(1)

where the distributions of the observed data under each hypothesis, \( p(x|H_1) \) and \( p(x|H_0) \), are assumed to be known. If \( \lambda(x) \) is greater than some threshold, \( \gamma \), then the null hypothesis is rejected. The specific threshold chosen is dependent upon a specific
user-defined performance criteria and is often application dependent, and again, ROC curves can be generated by calculating \( P_d \) and \( P_{FA} \) as \( \gamma \) varies.

In many real-world problems, \( p(x|H_1) \) and \( p(x|H_0) \) are not known exactly, or cannot be computed, and so the likelihood ratio test cannot be used. One common situation is that the functional form of the data distribution is known under each hypothesis, but the exact parameters of the distributions are unknown. One popular decision statistic used in this situation is the Generalized Likelihood Ratio Test (GLRT) [41], given by

\[
\lambda(x) = \frac{p(x|\hat{\theta}_{H1}, H_1)}{p(x|\hat{\theta}_{H0}, H_0)}
\]

where the \( \hat{\theta} \) parameters represent the maximum likelihood estimates of the unknown parameters under each hypothesis. In contrast to the likelihood ratio test, the generalized likelihood ratio test is not optimal. It is, however, a common approach used to design detectors for many practical applications, and can provide robust performance if the functional form of the distributions under each hypothesis is accurate.

The RX detector is based on the GLRT [28] with some specific assumptions about the functional form of the distributions under each hypothesis. Specifically it is designed to detect a signal with some unknown scalar amplitude \( b \) and some known template shape vector, \( s \), that is corrupted with additive white Gaussian noise, \( x^0 \). The two hypotheses for an observed data vector \( x \) are given by:
\[ H_0: \ x = x^0 \]
\[ H_1: \ x = x^0 + bs. \]  
(3)

As the formulation GLRT requires [41], the unknown parameters are estimated with their maximum likelihood estimates. This includes the variance of the white Gaussian noise vector \( x^0 \) as well as the unknown signal amplitude value \( b \). The resulting detector and decision statistic are given by

\[
\lambda(x) = \frac{(x^T s)(x^T x)^{-1}(x^T s)}{(s^T s)}. \tag{4}
\]

The decision statistic is a function of the data and the user-specified template vector, \( s \).

In this equation it is assumed that the sample mean of the vector \( x \), denoted by \( \hat{\mu}_B \) here, has already been removed.

In BTD applications the template vector \( s \) is specified based on the assumed shape of targets in the image data. Let us denote the maximum likelihood estimate of the mean within the template as \( \hat{\mu}_T \) and the maximum likelihood estimate of the variance of the entire dataset as \( \hat{\sigma}_B^2 \). Here the subscripts ‘T’ and ‘B’ refer to target and background respectively, referring to the regions of the data being used to compute the statistics. Now, observe that \( x^T s \) is a scaled estimate of \( \hat{\mu}_T \) and that \( (x^T x) \) is a scaled version of \( \hat{\sigma}_B^2 \). Substituting this notation into equation 8 above and ignoring scaling factors, the RX decision statistic \( \lambda_{RX} \) can be computed by

\[
\lambda_{RX}(x) = \left( \frac{\hat{\mu}_T - \hat{\mu}_B}{\hat{\sigma}_B} \right)^2. \tag{5}
\]
The scale factors can be ignored because they represent a monotonic transform of the decision statistic and do not impact the likelihood ratio.

The specific RX detector used in this work is based on an adapted version of the detector given in equation (4), which is commonly used in BTD research [8], [39]. It requires that the user specifies two windows: the target window and the background window. These windows are used to compute the mean and variance values given in (5). As before, the sizes of these windows are based on the expected sizes of target signatures encountered in the image data. A mismatch in size between these filters and the encountered target shapes/sizes can be very detrimental to performance. Improving this shape/size match between RX and target signatures in forward-looking video is the basis for the forward-looking RX algorithm proposed in Section 4.4.2. An example of the detector as it is applied to FLIR image data is illustrated in Figure 2.5 below.
Figure 2.5. This figure shows an illustration of the RX detector applied for landmine detection in an image from a forward-looking infrared camera. The user chooses two window sizes: a target window (red) and a background window (black). At each location in the image, statistics are computed using the pixels within each window, respectively. The decision statistic for that pixel is computed based on the statistics computed from the windows (bottom right).

### 2.3 Classification Algorithms

A classification algorithm is a function that takes a set of vectors and labels, where each vector has exactly one of two labels, denoted here as 0 or 1 corresponding to the two hypotheses being tested. The classification algorithm uses the data to learn a function that maps vectors, like those in the data set, into one of the two labels. In the context of this work, each vector in the dataset will be a set of statistics collected from sensor data around detection alarm locations. The classification algorithms will be trained to determine whether the statistics correspond to a target or to a non-target. In
this regard, classification algorithms are similar to detection algorithms, except they make fewer a-priori assumptions about the data and learn models for each class of observations directly from the data. Often classification algorithms can be split into two classes based on whether their learned functional form is linear or non-linear. See Figure 2.6 below for an illustration of linear and nonlinear classifiers in the following discussion.

Linear classifiers are simpler and computationally less expensive but may be unable to map vectors into their correct class labels if the true underlying functional relationship is complex (e.g., nonlinear). In this case, nonlinear classifiers are often good choices. However, because nonlinear classifiers learn more complex mappings they are also prone to incorporate structure in the data that is purely the result of noise and not representative of the true class variations. This is referred to as overfitting and it can be very detrimental to performance in practice (see [42], [43]). Therefore, depending on the nature of the data, either linear or nonlinear classifiers can perform best. There are many classification algorithms in the literature, but in this work two classifiers are used, one to represent each category of classifier: support vector machines (SVM, non-linear) [42] and the Fisher Linear Discriminant (FLD, linear). These two algorithms were chosen as representatives because they are relatively well understood and established in the literature (e.g., [42] [44]).
2.4 The Mean-Shift Algorithm

The mean-shift algorithm is a non-parametric technique for identifying the mode(s) of a probability distribution given discrete samples from that distribution [29]. In practice, however, it is often applied for unsupervised clustering of data [45]. It will be an important part of many proposed methods in this work.
The mean-shift algorithm attempts to cluster data points by assigning every point to exactly one cluster. The number of clusters is not user-specified but is determined automatically during the clustering process. Mean-shift achieves this by sequentially creating new clusters and assigning nearby points to them until every point belongs to a cluster. The location of each new cluster is randomly initialized and then moved to its final position through an iterative re-estimation process. Let \( x_i \) denote the \( i^{th} \) observation vector in the data, and let \( \mu_t \) indicate the cluster center location estimate at iteration \( t \). The estimate of the cluster center at the next iteration, given by \( \mu_{t+1} \) is given by

\[
\mu_{t+1} = \frac{\sum_{x_i \in N(\mu_t)} K(x_i - \mu_t) x_i}{\sum_{x_i \in N(\mu_t)} K(x_i - \mu_t)}.
\]  

(6)

Here \( N(\mu_t) \) is a user-defined neighborhood of the current mode location and \( K(\cdot) \) is a kernel function that controls the weight of a data point in influencing the new mode estimate based one that points distance to the current mode estimate. In this work \( K(\cdot) \) is a Gaussian kernel which is a well-studied and popular choice in the literature (e.g., [45], [46]). The Gaussian kernel has a width parameter, \( \sigma \), which can be optimized based on the data, or chosen by hand based on the expected size of clusters in the data. Both methods will be used in this work.

Re-estimation of a given cluster stops when the distance between two successive center estimates \( \mu_{t+1} - \mu_t \) becomes small. When the cluster center converges, all the points within some neighborhood of the cluster center are assigned to that cluster. In
this work, the neighborhood corresponds to a radius around the cluster center determined by the width of the Gaussian kernel. After convergence is reached, a new cluster center is randomly initialized and estimated based on any remaining un-clustered points. This is repeated until all points belong to at least one cluster.

### 2.5 Geometric Camera Modeling

Geometric camera modeling is used to describe the way objects in 3-dimensional (3D) scenes appear when they are captured as a 2-dimensional (2D) image in a camera. As its name suggests, these models use geometric laws and model light rays as lines in 3D space to generate functions that can map coordinates in the 3D scene into a 2D image, given some information about the camera. One particular mapping, the perspective projection, is widely used for this purpose and will be an important part of the detection algorithms proposed for the FLIR camera data in Chapter 4. It will also be necessary to map the locations of alarms in the FLIR video, given in pixel locations, to their corresponding locations on the earth. This section outlines the basic geometric camera model, the pinhole camera, and then describes how the perspective projection is derived from this model and used to map between 3D world coordinates and 2D image coordinates.

#### 2.5.1 The Pinhole Camera Model

The pinhole camera model provides a basic geometric model for image acquisition that is used in many applications, including many previous FLIR detection
studies [8], [34], [39]. The pinhole model was inspired by the dynamics of a pinhole camera. An example of a simple pinhole camera imaging a cylinder is shown in Figure 2.7 below. Notice that the image of the cylinder is reversed compared to its orientation in the world. Objects do indeed appear reversed in a real pinhole camera, however when using the pinhole model it is often assumed that there is a separate image plane called the virtual image plane that exists between the scene and the pinhole. The virtual image is effectively the same as the true image plane except that the objects are not reversed. When referring to the image of an object, it will be assumed that the virtual image is being used.

![Basic Pinhole Camera](image)

**Figure 2.7.** This figure illustrates an example pinhole camera. A 3-dimensional cylinder (left) is imaged by the pinhole camera, and the resulting virtual image (2nd from left) and true image (right) are shown. The dot on the cylinder is shown to indicate the relative orientation of the object as it appears in each image.

The basic objective of the pinhole model is to describe the relationship between a point in the real world and its location in an image. To define this relationship mathematically requires a mathematical description of location. In most imaging
models a Euclidean coordinate description is used for this purpose. An important concept in camera modeling is frame of reference, which defines the relative orientation and scale of a coordinate system compared to other coordinate systems. In general a single point will have different coordinate values in two different reference frames. One of the main goals of the pinhole model is to relate points in different frames of reference to each other mathematically.

There are three important frames of reference in the pinhole imaging model: the image frame, the camera frame, and the world-frame. The image reference frame (referred to as image-space) describes the location of an object within an image. The top left corner of the image is considered the origin and coordinates become positive as they move down and to the right. The camera frame of reference (referred to as camera-space) describes points based on their location relative to the camera. The world frame of reference (referred to as world-space) describes where objects are located in some real-world coordinate system such as global positioning system (GPS) coordinates. In this work the three axes of any reference frame will be denoted by $i, j,$ and $k$ respectively. Subscripts will indicate which reference frame the axes correspond to, as shown in Figure 2.8 below. In general each reference will have a different origin, scale, and orientation. A point in any reference frame can be described using a Cartesian coordinate, $X = \{x, y, z\}$, however typically $X \in \mathbb{R}^2$ for an image-space coordinate.
Coordinates will be given a subscript to denote their frame of reference, in similar fashion to the axes shown in Figure 2.8.

![Frames of Reference](image)

Figure 2.8. This figure illustrates the major frames of reference in the pinhole camera model: world-space, image-space, and camera-space. The relative orientation and position of each frame of reference is indicated by the axes. It should be noted, however, that the world-space frame can take any orientation and shift relative to the other two frames.

### 2.5.2 The Perspective Projection

Although relatively simple, the notation above combined with the pinhole geometry are enough for modeling imagery in many practical situations. From the pinhole model it is possible to derive a useful mathematical tool known as the perspective transform [37], [45]. The perspective transform is a linear coordinate transformation that has many applications in image modeling. The major application of interest here is its use in projecting 3D points from camera-space into their
corresponding 2D image-space coordinates. If the perspective projection matrix is denoted by \( K \), then the fundamental relationship between a 3D camera-space point and its corresponding 2D image-space coordinate is given by

\[
\begin{bmatrix}
    x_i \\
    y_i \\
    1
\end{bmatrix} = K
\begin{bmatrix}
    x_c \\
    y_c \\
    z_c \\
    1
\end{bmatrix},
\]

where the subscript \( c \) denotes a coordinate in camera-space and the subscript \( i \) denotes a coordinate in image-space.

Although this relationship is relatively simple, obtaining the matrix \( K \) requires detailed knowledge of the camera, which is not always available. In such cases it is possible to estimate the matrix \( K \) by other means. This estimation, known as geometric camera calibration, usually involves collecting imagery of some object with known camera-space coordinates and image-space coordinates [37]. With this data the camera calibration basically can be formulated as an optimization problem where the 12 values in the 3x4 projection matrix \( K \) are optimized to minimize the error produced when projecting the known world-space coordinates of the object into their known image-space counterparts.

In many practical situations where the camera parameters are unknown, there may be a reasonable alternative approach to trying to estimate a projection matrix. Consider a situation where the imaged scene is a flat surface that is well represented by the \( z \)-plane in camera-space. This is basically the situation where the earth is flat, and
the camera is level; it is referred to as the flat-earth assumption. In FLIR-based BTD applications this has been proven to be a practical assumption under some circumstances [39]. The effect of this assumption is effectively to remove one of the dimensions in camera-space so that the transformation is now given by

\[
\begin{bmatrix}
    x_i \\
    y_i \\
    1
\end{bmatrix} = K \begin{bmatrix}
    x_c \\
    y_c \\
    1
\end{bmatrix}.
\]

The new projective transformation is now a 3x3 matrix with only 9 parameters rather than 12 parameters as before. Having fewer parameters naturally makes the problem of estimating the camera parameters more tractable.

### 2.5.3 World-Space to Image-Space Projections

One of the primary purposes of the perspective projection is to transform coordinates from camera-space into image-space. To use the perspective projection matrix \( K \), for this purpose requires knowledge of the coordinates of some point, relative to the camera's position. In many situations the location of an imaged object is only known in some world-space reference frame which is not relative to the camera. Therefore an additional transformation must be used to translate the world-space coordinate into its camera-space equivalent.

One example of such a situation is in FLIR-based BTD [8], [34], [39]. Target coordinates are usually provided in 2D GPS coordinates. In order to map them into their image-space coordinates they must first be transformed into camera-space. In
previous FLIR detection literature, and in this work, a flat-earth assumption can be made, in which case the transformation from world-space to camera-space is simply composed of a translation and a rotation operation in 2 dimensions. Given knowledge of the world-space orientation, \( \theta_w \), and location, \( \pi_w = [\pi x_w \ \pi y_w] \), of the camera this transformation can be computed by

\[
\begin{bmatrix}
\hat{x}_c \\
\hat{y}_c \\
1
\end{bmatrix} = \begin{bmatrix}
\cos \theta_w & -\sin \theta_w \\
\sin \theta_w & \cos \theta_w
\end{bmatrix} \begin{bmatrix}
x_w \\
y_w \\
1
\end{bmatrix} - \begin{bmatrix}
\pi x_w \\
\pi y_w \\
1
\end{bmatrix},
\]

(9)

where the subscript \( w \) denotes a known world-space coordinates of some imaged object. This process of rotation and translation is illustrated in Figure 2.9 below. We can use the augmented form of the vector similar to (9) and write this equation as follows

\[
\begin{bmatrix}
\hat{x}_c \\
\hat{y}_c \\
1
\end{bmatrix} = \begin{bmatrix}
\cos \theta_w & -\sin \theta_w & -\pi x_w \\
\sin \theta_w & \cos \theta_w & -\pi y_w \\
0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
x_w \\
y_w \\
1
\end{bmatrix} = T \begin{bmatrix}
x_w \\
y_w \\
1
\end{bmatrix},
\]

(10)

![World-Space to Camera-Space](image)

Figure 2.9. This figure graphically illustrates how location coordinates are mapped from the world-space reference frame to the camera-space reference frame. In the camera-space reference frame (right), the camera is at the origin and pointing
along the $i$-axis. The mapping from world-space to camera-space is determined by finding the mapping that moves the camera from its world-space location $(\pi x_w, \pi y_w)$ and orientation $(\theta_w)$ to its camera-space location $(0, 0)$ and orientation $(0^\circ)$. The object on the left at $(x_w, y_w)$ in world-space is located at $(x_c, y_c)$ after applying this mapping.

Assuming that a projective transformation is available, via knowledge of the camera or due to estimation, then the matrix in (10) can be combined with the perspective transformation to yield a complete transformation from world-space into image-space, given below as

$$
\begin{bmatrix}
  x_i \\
  y_i \\
  1
\end{bmatrix} = K T \begin{bmatrix}
  x_w \\
  y_w \\
  1
\end{bmatrix} = K' \begin{bmatrix}
  x_w \\
  y_w \\
  1
\end{bmatrix}.
$$

(11)

This provides a compact mathematical formula for converting between these coordinate systems. It is important to keep in mind that knowledge of the camera location and orientation is required, and that the matrix $K'$ is a function of these values.

### 2.6 Queuing theory background

This section provides a review of queuing theory [47], and queuing theory results required to derive the multi-sensor BTD queuing model in Section 6.1. Queuing theory is the study of queues (waiting lines), where it is assumed that there is a “station” where “jobs” arrive to receive service. The rate (per unit time) that jobs arrive is known as the arrival rate, and the rate at which they finish processing and depart the station is known as the service rate. Queuing models can become very complicated, however, the aforementioned components are the essential components, which are always present. A
queuing model corresponding to this basic structure is depicted below in the left-most panel of Figure 2.10, however, with one atypical feature. This atypical feature is included because it is needed for the BTD queuing model in Section 6.1. The queue in Figure 2.10 assumes that, after each job completes processing at the station, there is some probability, \( \alpha \), that the queue halts operation for some average time, \( T_{stop} \), and that all existing jobs are discarded before continuing operation as normal. There is likewise a probability of \( \bar{\alpha} = 1 - \alpha \) that the queue operation continues without interruption. The motivation for this modification is presented in more detail in Section 6.1 when it is utilized to model a BTD system, however, in this section it is assumed that such a queue is useful, and the focus is on the characteristics of such a queue.

Much of the utility of queuing models (i.e., computing useful queue properties) requires that the models meet two criteria: the job arrival times and job service times must form Poisson processes, respectively [47]. In other words, the probability of the number of job arrivals, or jobs serviced, in some fixed amount of time can be described by a Poisson distribution. This is also equivalent to requiring that the inter-arrival, and inter-service, times for jobs are independent and exponentially distributed [47]. If these conditions are met, then it can be shown that the queuing model has an equivalent representation as a stochastic process, called an irreducible homogenous continuous-time Markov Chain (CTMC) [47]. CTMCs model queues with discrete states of operation in which the queue randomly moves between during operation. A state-
diagram for the CTMC representation of the queue described here is shown below in the right-most panel of Figure 2.10. CTMCs are described in more detail in the next section.

With the exception of the state labeled "Stop", the state of the queue depends on the number of jobs in the queue (i.e., the CTMC is in state $i$ when there are $i$ jobs in the queue). The rate of transition between states (which is formally defined in Section 2.6.1) depends upon the rate parameters of the Poisson processes (see queue in Figure 2.10). For the "Stop" state it is assumed that the waiting times (how long the system is stopped) is exponentially distributed with a mean of $T_s = 1/\mu_{stop}$. This implies that the waiting times are mutually independent (memoryless) and form a Poisson process with rate $\mu_{stop}$. This assumption is made because few statistics about stopping times are available and the aforementioned assumptions are very weak: simply assuming mutual independence of stopping times. In the models presented here, the transition rate parameters are the same for each state, but the model and results presented here are also valid if each state has unique arrival and service rates. This is an important generalization because it is used for the multi-sensor BTD queuing model presented in Section 6.1.

In Section 6.1 a 2-sensor BTD system will be modeled as a queue, and this will lead to an equivalent representation of the BTD as a CTMC. This representation, in turn, will make it possible to derive expressions for useful quantities for the BTD system, specifically the system ROA. Deriving these quantities will require the use of some
specific theorems and results for CTMCs. The following Sections provide a brief review of the most important theorems that are needed for deriving the BTD models that are presented in the Chapter 6. For more details on queuing theory, and the theorems presented here see [47].

### 2.6.1 Irreducible homogenous continuous-time Markov Chains

An irreducible homogenous continuous time Markov chain is a particular type of stochastic process that has a number of special properties that make it very useful for practical problems. Let $X(t)$ be the random variable of a stochastic process, indexed by $t$, that can assume values over a discrete state-space, indexed by $I = \{0,1,2,\ldots\}$. Assume the index parameter, $t$ is defined over $T = [0, \infty)$. This is a continuous time stochastic process and if the distribution over any given state in the process is only dependent upon the most recent state of the process, then it is a Continuous-time Markov Chain (CTMC). A homogenous CTMC is one where the conditional probability of being in state $j$ at time $t + \nu$ after being in state $i$ at time $\nu$ is constant over time, and given by

$$p_{ij} = P(X(t + \nu) = j | X(\nu) = i), \forall \nu, t \geq 0,$$  \hspace{1cm} (12)

The transition rate for a homogenous CTMC from a state $i$ to another state $j$ is constant over time, and is given by:

$$q_{ij} = \frac{\partial}{\partial t} p_{ij}(\nu, t)|_{\nu=t}. \hspace{1cm} (13)$$
A state $j$ is said to be reachable from state $i$ if for some $t > 0, p_{ij}(t) > 0$. A CTMC is said to be irreducible if every state is reachable from every other state. It can be shown for an irreducible CTMC, the limits

$$
\pi_j = \lim_{t \to \infty} p_{ij}(t) \quad i, j \in I
$$

always exist and are independent of the initial state $i$. In other words, over time the state probabilities approach particular steady-state values, which will be denoted here by the vector $\pi = [\pi_0, \pi_1, ...]$. Further, and more importantly, computing the values of $\pi$ for irreducible homogenous CTMCs can be accomplished by constraining that the state probabilities sum to one, and solving the following system of linear equations:

$$
\pi Q = 0
$$

Here $Q$ is called the generator matrix of the CTMC and is given by $Q = [q_{ij}]$. This result illustrates that once the transition rates are set (the values of $Q$), then the state probabilities can be computed by solving a simple system of linear equations. Once $\pi$ is obtained, numerous useful properties of a queue can be computed using what is known as the Markov reward model. This model computes an expected reward (which is typically a desired property of the queue) as a weighted sum of the state probabilities, given by

$$
expected \ reward = E_X[r] = \sum_i r_i \pi_i.
$$
Here the weight for each state $i$ is given by $r_i$. A simple practical application of the reward model is the computation of the average queue length by assigning, $r_i = i$. By modeling the FLIR-GPR system as a CTMC and applying the Markov reward model it will be possible to compute the average system ROA.

![A simple queuing model and CTMC model for the queue](image)

Figure 2.10. The panel on the left illustrates a simple queue model where jobs arrive at a station for processing. After processing there is some probability, $\alpha$, that operation will halt for a short time and the queue will be cleared. If the job arrivals and departures form Poisson processes with parameters $\lambda$ and $\mu$, respectively, then the queue can be represented equivalently as an irreducible homogenous CTMC (right-most panel). The $i^{th}$ state of the CTMC has $i$ jobs in the queue, with the exception of the “Stop” state. Each state is indicated by a circle. The transition rates, $q_{ij}$ between states $i$ and $j$ (formally defined in the text) are indicated in the graphic with arrows and their values. This CTMC representation facilitates the use of many theoretical results that are used to derive expressions for queue properties. In Section 6.1 a 2-sensor BTD system is modeled as a queue (and CTMC), and expressions are derived for system characteristics, such as ROA.

### 2.7 Summary

This chapter reviewed the background material necessary to develop the methods proposed in the subsequent chapters of this work. An introduction was given to the basic methodologies for algorithm development in the context of BTD. These conventions are used throughout the remainder of this work to test and compare the
performance of many of the proposed algorithms to the performance of existing algorithms. The last few sections described some of the basic algorithms from the literature that will be used, or provide the basis for, proposed algorithms in this work. This included anomaly detection algorithms, supervised classification algorithms, and unsupervised clustering algorithms. The pinhole camera model was introduced as a method to model the changes imposed on collected data as a function of the perspective of the camera collecting it. This is primarily for the development of the algorithms for FLIR data, but is an important concept therein. Lastly, a review of queuing theory and some of its important theoretical results were presented, which are needed for the derivation of the sensor management model in Chapter 6.

The next chapter begins describing the work that has thus far been completed in pursuit of increasing standoff in BTD. This begins with the improvement of detection algorithms for two large-standoff detection modalities: seismo-acoustic sensors and the FLIR camera. Seismo-acoustic sensing and detection is described first in the next chapter.
3. Seismo-acoustic detection

Seismo-acoustic sensing is remote sensing modality for BTD that may eventually offer a large standoff alternative to downward-looking GPR. A major challenge leading to this objective is the development of target detection algorithms with higher Pds and lower FARs. This chapter describes the development of algorithms to improve BTD using seismo-acoustic data. First, some background information about the seismo-acoustic sensing modality is presented, followed by a description of some background topics that are specific to this seismo-acoustic work. In the third chapter we present our algorithm, spatiotemporal keypoint clustering, that addresses some of the challenges associated with seismo-acoustic-based detection. Experimental results are presented in the last chapter that illustrate how spatiotemporal keypoint clustering improves over the performance of existing baseline seismo-acoustic detection methods.

3.1 Sensing methodology and detection challenges

In recent years, seismo-acoustic measurement has been investigated as a means to remotely detect buried targets. As presented here, seismo-acoustic measurement refers to seismic (vibrational) measurements of the ground taken by a laser Doppler vibrometer (LDV) at a query location after it is insonified with an acoustic source. Detection using this modality relies on the existence of mechanical differences between the object of interest, in this case a buried target, and the surrounding earth. These mechanical differences then manifest in the data as a change measured vibration of the
ground in response to the acoustic stimuli [48]–[51]. An illustration of this remote sensing framework is shown below in Figure 3.1.

**Figure 3.1.** An illustration of the seismic data collection system used in this study. An acoustic stimulus \( a(t) \) is applied at a query location and the laser Doppler vibrometer (LDV) is focused at one particular spatial location \((x, y)\) on the ground where it collects a time-series of vibrational data at that location, \( v(t, x, y) \). This process is repeated several times at different spatial locations on a grid over a patch of earth. This results in the collection of a data volume over the query patch of earth (shown on the right). The red columns correspond to the time-series recorded at each spatial location by the laser vibrometer, shown on the left with corresponding red arrows. The transfer of energy from an acoustic waveform to vibrational energy (as measured by the LDV) is often modeled as a linear system that is completely characterized by an acoustic-to-seismic (A/S) transfer function, \( h(t, x, y) \). Many existing detection methods rely on estimating \( h(t, x, y) \) to detect the presence of a buried target.

Many studies have investigated the potential effectiveness of this sensing modality for detection, often with positive results. For signal processing purposes, the remote sensing process is often modeled with a linear acoustic-to-seismic (A/S) transfer function, wherein the LDV-measured seismic waveform is assumed to be a scaled and phase-shifted version of the stimulus acoustic signal [21], [52]. Detection then becomes a matter of estimating the A/S transfer function and detecting the changes in it that are characteristic of the presence of a buried target. In many early studies this was
accomplished by computing statistics associated with the frequency response (Fourier domain) of the raw seismic data [48]–[51], [53]. In more recent studies, the data is preprocessed by convolving the recorded seismic data with the stimulating acoustic signal [21], [52], [54]. The result of this preprocessing is an estimator of the A/S transfer function.

Although established Fourier-domain statistics have been shown to have some discriminative ability, they have several limitations that could potentially be addressed to improve detection performance (e.g., $P_d$ given a fixed FAR). One obvious limitation is that the features do not capture spatiotemporal correlations that exist in the data among target signatures. In other words, target signatures tend to appear across narrow ranges of time and space, but the Fourier features are applied across all time and space within the data. This tends to average out the target signature with large amounts of noisy uninformative data. Detection could potentially be improved by first localizing anomalous objects in space and time, and only then performing detection.

We have proposed a novel detection algorithm, called spatiotemporal keypoint clustering (SKC) [25], to address the limitations of established Fourier features by localizing target signatures in space and time. The SKC algorithm operates by first detecting anomalous locations, called keypoints, in space and time and then clustering them to find the most likely location where a target might exist in the data. The likelihood of a cluster corresponding to a target is based on the number and strength of
its cluster member keypoints. The performance of the proposed algorithm is compared with a set of established linear Fourier-domain features on a large collection of LDV-based seismic observations. The performance of human observers is also assessed in order to establish an estimate of the achievable performance on this problem. The results indicate that the SKC algorithm outperforms the established Fourier features and results in detection performance closer to that achievable by human observers.

3.2 The seismo-acoustic dataset

The detection experiments in this work were conducted on seismic data collected at 98 separate patches of earth under varying conditions at an Eastern U.S. test site. Targets were buried in 65 of the 98 patches. In each case the objects were buried near the center of the patch, and appear near the center of the corresponding seismic data block.

At each location seismic measurements were collected over a grid of points with approximately 5cm spatial resolution. At each grid point seismic vibration data was collected in the form of instantaneous surface velocity, using the LDV in response to a template acoustic stimulus. After this process was completed for each grid point in a given patch, the result is a 3D volume of instantaneous velocity measurements. This collection process is illustrated in Figure 3.1 for a single patch. The acoustic stimulus in this case was a linear chirp waveform ranging from 50 Hz to 250 Hz.
3.3 Seismo-acoustic background methods

This Chapter describes several concepts and methods that are based directly on previous work in the seismo-acoustic literature. First an important data preprocessing step is discussed that is performed on all data in this work. After that, a set of Fourier features are described for use in detection that are based on established features from the literature. In the last part of this Chapter, a method is described for estimating the performance of human observers specifically for seismo-acoustic detection.

3.3.1 Data preprocessing

This Chapter describes an important preprocessing step that is applied to all of the measured seismic data. The main goal of this preprocessing is to estimate the A/S transfer function. As discussed in the introduction, a significant body of prior work on seismo-acoustic data has focused on modeling the ground as a linear system that operates on the acoustic stimulus signal and results in the seismic (vibrational) output data recorded by the LDV. In this framework the ground, and anything in it, is completely characterized by an A/S transfer function. This conceptualization is illustrated and described in Figure 3.1 above. BTD within this framework then relies on first estimating the A/S transfer function and that is the goal of the data preprocessing described here.

The method of estimating the A/S function used here was developed by Lee [55] and has been used in several recent seismo-acoustic studies [21], [52], [54].
is obtained by cross-correlating the seismic measurement data (shown in Figure 3.1) with the acoustic stimulus waveform. This process is illustrated in Figure 3.2 below. An estimate of the acoustic stimulus waveform was obtained by measuring it with a microphone that was placed on the ground near the insonified patch of earth.

This procedure can be described mathematically as follows. Let the stimulus signal waveform be given by \( a(t) \) and the measured LDV data at location \((x,y)\) be given by \( v(t,x,y) \). Then the A/S transfer function at that location \( h(\tau,x,y) \) can be estimated by:

\[
\hat{h}(\tau,x,y) = a(t) \otimes v(t,x,y) = \int a(t)v(\tau + t,x,y)dt
\]

(17)

This is a relatively unintuitive procedure and so the derivation is outlined here in the context of this seismo-acoustic detection problem. This derivation can also be found in more detail in Polge et al., [56]. The A/S estimate relies on an assumption that the measured vibration time-series, \( v(t,x,y) \) can be modeled by the following convolution:

\[
v(t,x,y) = a(t) * h(t,x,y) + \eta(t,x,y) * h(t,x,y)
\]

(18)

Where \( \eta(t,x,y) \) is some noise or disturbance that is statistically independent of the acoustic signal \( a(t) \). Let the cross-correlation between \( a(t) \) and \( v(t,x,y) \) be denoted by \( \phi_{av}(\tau) \), then we have

\[
\phi_{av}(\tau) = a(t) \otimes [a(t) * h(t,x,y) + \eta(t,x,y) * h(t,x,y)]
\]

(19)
\[ \phi_{aa}(t) * h(t,x,y) + \phi_{aa}(t,x,y) * h(t,x,y) \]  
(20)

Where the subscripts of each \( \phi \) indicate which time-series are being cross-correlated.

Now, note that if \( a(t) \) and \( \eta(t) \) are independent and zero-mean, then the second term in (21) is always equal to zero, leaving

\[ \phi_{av}(\tau) = \phi_{aa}(t) * h(t,x,y). \]  
(22)

Now, if \( \phi_{aa}(t) = k\delta(t) \), where \( \delta(t) \) is the unit impulse, this leaves

\[ \phi_{av}(\tau) = kh(\tau), \]  
(23)

which is a scaled version of the impulse response. This occurs if the acoustic stimulus signal is white Gaussian noise with a spectral density equal to \( k \). The acoustic stimulus here is not white Gaussian noise, but it is a linear chirp that approximates the frequency response of white Gaussian noise because its spectral density is uniform (although band-limited). This processing is applied at every location \((x,y)\) in a given observation to yield the processed data volume and is used before all other processing discussed in subsequent sections.
Figure 3.2. An illustration of the preprocessing applied to each seismic data volume in the data collection. The seismic time series at each spatial location $v(t, x, y)$ (two time series are indicated by the red columns in the center graphic) is cross-correlated with the stimulating acoustic signal, $a(t)$, yielding another volume where each time series is an estimate of the ground A/S transfer function at that location, $h(\tau, x, y)$. All detection methods operate on the processed data volume.

### 3.3.2 Fourier features

This section describes a set of Fourier domain statistics that will be used in this work to establish a baseline detection performance (e.g., $P_d$ given a fixed FAR) against which more advanced processing proposed is compared. These features are based on similar established spectral features from the literature where statistics were computed on the frequency response of the raw seismic data [48]–[51], [53], or more recently, on the frequency response of the estimated A/S transfer function [21]. In this work the Fourier features are computed on the frequency response (Fourier transform) of the estimated A/S function (as estimated using the preprocessing described in the previous section).

The first step towards computing the Fourier features is the estimation of the A/S transfer function using the method described in Section 3.3.1 above. To obtain the
frequency response of the A/S transfer function, the Fourier transform is taken of the estimated A/S time series at each spatial location \((x, y)\) within each observation. A set of Fourier features is then computed at each spatial location using windowed averages of the amplitude and phase of the frequency response. Let the Fourier transform of the processed vibration data at a spatial location \((x, y)\) be given by \(\tilde{H}(f, x, y)\), where \(f\) denotes the frequency in Hertz. The \(i^{th}\) pair of phase and amplitude features at spatial location \((x, y)\) is then computed by

\[
Amplitude_{f_i}(x, y) = \frac{1}{10} \int_{f_i-5}^{f_i+5} |\tilde{H}(f, x, y)|, \quad (24)
\]

\[
Phase_{f_i}(x, y) = \frac{1}{10} \int_{f_i-5}^{f_i+5} \arctan \left( \frac{\text{Im} \left( \tilde{H}(f, x, y) \right)}{\text{Re} \left( \tilde{H}(f, x, y) \right)} \right), \quad (25)
\]

where \(f_i\) denotes the center frequency of the window. As indicated in the equations, the windows used for feature computation were 10Hz wide. These windowed averages were taken from 50Hz to 250Hz with a 5Hz overlap between neighboring windows. The frequency range of 50Hz to 250Hz corresponds to the frequency range of the linear acoustic chirp signal used to stimulate the seismic response.

Once these features were computed at every spatial location \((x, y)\) in a given observation, they were spatially averaged to create a single feature vector that was used to represent the entire observation. However, only a subset of spatial locations was used in the spatial average. In each observation a human annotator manually identified a red box (i.e., a rectangular spatial region), indicating where the annotator believed the
target signature was within the given observation. This was performed whether or not a target was present within the observation. Only pixels from this red box were used in the spatial feature average. The placement of these boxes is described in more detail in Section 3.5 below. The red boxes were used to determine the predictive power of the features under more ideal circumstances where the target signature is localized spatially.

This approach is not practical in real applications but it provides a way to determine the full potential efficacy of the features for target/non-target discrimination, and therefore answers an important question about whether more effort should be invested into investigating the features, or other Fourier-based features.

### 3.3.3 Classification with human observers

This section describes the experiments used to estimate the ability of humans to visually discriminate targets and non-targets in the preprocessed seismic data (the estimate of the A/S transfer function). Human performance is sometimes used for pattern recognition problems as an estimate of the achievable performance [57]. Although automatic methods may ultimately perform better than humans, it is useful to establish some performance level that is known to be achievable. In these experiments, three human observers were each independently presented with the same set of processed seismic data and asked to assign a value between 1 and 10 indicating their confidence that a target is present. A graphical user interface was created to allow the user to easily visualize and examine each data volume and submit a rating.
There were a total of 98 observations available for the experiment and this set was split into a training set and a testing set. The training set consisted of 20% of the observations and was representative of the conditions present in the testing set. Each human observer had the same training and testing set, however, the testing observations were presented in randomized order. After ratings were collected for all three human observers, their ratings for each observation were averaged together to obtain a final decision statistic for each seismic observation. Note that the human results utilize 79 testing observations (80% of the data), rather than the full 98 which are used to estimate performance for the other algorithms.

Figure 3.3. This figure shows the graphical user interface used in the human observer experiments. It allows the user to explore the preprocessed data volume with the slider at the bottom. Each image corresponds to all the spatial values at one instance in time. The user can record a rating at the bottom left indicating their level of confidence that a target is present. Some observations contained a bright fiducial object and these objects were removed from consideration with a large black box. An
example is visible in the figure. This box was placed in every observation so it could not be used as a cue for the presence of a target. A red target box was also drawn in each observation by the authors. This box indicated where each human observer should look during testing. The target box was also placed in each observation whether it contained a target or not. This red box was also used to control where the Fourier features were extracted in each observation.

Some observations contained a fiducial object that could bias the results of the human observers. Therefore these objects were removed from consideration with a black box. An example is visible in Figure 3.3. This box was placed in every observation, whether it contained a target or not, so it could not be used as a cue. A red target box was also drawn in each observation by a human observer. This box indicated where each human observer should look during testing, and reflected the best estimate of the spatial location of the target by the human observer. These red target boxes were also placed in each observation whether or not they contained a target. This red box was also used to control where the Fourier features were extracted in each observation.

3.4 Spatiotemporal keypoint clustering

The preprocessed seismic data theoretically contain all the information that is useful for detection; however, it also contains large amounts of uninformative data. Typically, most of the images in the data contain little, or no, evidence of a target: they appear noisy and empty. The target signatures appear only in a localized region of the data in time and space. This is not necessarily a challenge except that the specific frames of data in which the target appears changes from collection to collection, as well as the
extent of time over which it appears. However, within this region, the signature is very apparent and can be detected by eye. Figure 3.4 illustrates this characteristic of a target-containing data volume with examples of informative and uninformative images. The target signature is clearly visible in the informative bottom image as a bright blob. Target signatures typically appear as blobs that alternate intensity (bright and dark) across images.

![Cross-correlation of stimulus & vibration data at a single spatial location](image)

**Figure 3.4.** This illustration depicts how target signatures only appear in relatively small portions of the preprocessed data volume. Most of the cross-correlation data is relatively uninformative (top right black image) while just a few images contain most of the informative target signature responses (bottom right).

Targets tend to manifest as bright or dark blobs.

Because most of the data volume is uninformative, any statistics collected over the entirety of the data volume are likely less effective for detection. This is the case for many existing seismo-acoustic features in the literature. These features rely on the A/S transfer function estimate that is computed using a large time series that is mostly empty of target signal. After these features are computed at each spatial location, they are potentially further degraded by aggregating over regions of space because most of the
data over space is also empty. The SKC algorithm is motivated by the need to address this issue and localize the target signature in time, space, and scale prior to making a target/no-target decision.

The SKC algorithm consists of two parts: identifying each location in time and space that may contain a target signature, and then localizing them by clustering each identified suspicious location. To identify target signatures a multi-scale blob detector (described in Section 2.2.1) is applied. The scales for the LoG filters are chosen based on the expected size range of target signatures in the data. The rationale is that, given the presence of a target, the blob detector will produce strong keypoints that tend to lie in a relatively narrow temporal and spatial region. In contrast, non-target observations will produce weaker keypoints that are more evenly distributed across time/space.

After keypoints are generated, localization is achieved by clustering the keypoints using the mean-shift algorithm described Section 2.4. Each cluster will be considered a new keypoint with a new confidence. In order to make use of all of the information provided by the cluster members, the new confidence for each cluster is assigned to be the sum of the keypoint responses of all of its members. This will give more weight to clusters that contain both, higher confidence keypoints, and more keypoints. The decision statistic for a given observation is taken to be the confidence of the largest confidence cluster obtained in that run. The SKC algorithm is illustrated for
two separate observations in Figure 3.5: one target observation and one non-target observation.

![Extracted keypoints from a target observation](image1)

![Extracted keypoints from a non-target observation](image2)

Figure 3.5. Each figure here plots the 10 strongest detected keypoints in the preprocessed data volume for a target observation (left) and a non-target observation (right). The size of each plotted point is proportional to the response of the LoG filter for those points. Two additional images are shown for each observation (on both the left and the right), corresponding to the images where the two strongest keypoints were detected for the target and non-target observations. The keypoint spatial location is also plotted in the subimages. In the target observation, the keypoints cluster tightly and have higher confidence. The blobs in the image data are clearly very strong in those frames of data. The non-target observation keypoints are further apart and are weaker, reflecting the fact that there is no target signature in the data, and no noticeable blob is observed in the subimages.

### 3.5 Experimental results & discussion

This section presents the experimental results for three detection methods: human observers, Fourier features, and the SKC algorithm. All performance estimates were computed using the data described in Section 3.2. The human performance was computed on 80% of the data (20% was used for training), whereas the Fourier features and SKC results are computed on 100% of the data. Additionally, for clarity, only the results for the best two Fourier features are reported. As a result of this performance-
based sampling, it is likely that the performance estimates of the Fourier features are optimistic.

The results for all experiments, presented in Figure 3.6, indicate that no particular detection method results in performance that dominates across all ROC curve operating points. The human observers almost always perform better than the two Fourier features, with the exception of the last few targets (i.e., high probability of detection). The SKC algorithm is notable because it outperforms humans at detecting the first 75% of the target observations. The SKC algorithm has nearly half as many false alarms as humans at the point when it detects 75% of the targets. The remaining 25% of targets are very difficult for SKC to detect and this method accrues many false alarms before detecting the remaining targets.
Figure 3.6. Plot of ROC curves for the human observers, the SKC algorithm, and the top two (human-informed) simple features. The human observers performed the best but not at all operating points: the SKC algorithm has better performance at lower false alarm rates. The simple Fourier features perform best at very high PD, but have high false alarm rates.

One potential method to improve algorithmic detection and improve over human performance is through feature fusion. The SKC algorithm and the phase features cue on fundamentally independent information in the data, motivating investigation of their potential for fusion. Figure 3.7 below shows two scatter plots, each plot showing a single combination of the three feature sets (not including humans). The plot on the left-side shows the two top performing Fourier features. These two features are highly correlated, reflecting the fact that they are designed to cue on very similar physical phenomenon, and therefore carry similar information. The right-hand side plot shows one of the phase features (the better performing of the two by area under the
ROC curve (AUC)) scattered with the SKC algorithm output. Here the features are much less correlated, as expected. The black circles in the plots mark non-target observations where one detection method assigned a high target probability, and the other (correctly) assigned a low target probability. This illustrates that in many cases the two methods do indeed cue on different information in the data.

In order to test this hypothesis, the SKC algorithm and Phase feature were fused together using two classification methods: a fisher linear discriminant (FLD), and a support vector machine (SVM) (see [42] for details). These two algorithms were used because they are standard classifiers for linear and nonlinear classification problems, respectively. FLD classifiers often work well when there is little data available, such as the situation here. SVMs are excellent classifiers for nonlinear decision problems, and based on the right-hand scatter plot in Figure 3.7, a nonlinear classifier may be more appropriate. Leave-one-observation-out cross-validation was used for training and testing the algorithms. The results are shown below in Figure 3.8. The results indicate that both fusion algorithms improve performance in some regions of the ROC curve and degrade it in others. However, the nonlinear SVM fusion tends to perform much better than the linear FLD fusion. As mentioned before, given the nonlinear distribution of the data shown on the right in Figure 3.7, this is not a surprising result. Although SVM’s are better suited for nonlinear problems, there is very little data available in this fusion problem, making it more difficult to learn a non-linear hyper plane. It is plausible that
more data may lead to much better non-linear fusion performance, although more investigation is required to confirm this.

Figure 3.7. Here, two scatter plots are shown, each with a different combination of features. In the left-hand plot the two best Fourier features are scattered together. Targets are shown in red, non-targets in blue. The plot indicates that they are highly correlated, which is not surprising since they correspond to physically similar phenomenon. It is unlikely that fusing these would be beneficial. The right-hand plot shows the SKC algorithm along with one of the phase features. The black circles indicate non-target observations where one, but not both, of the two features indicates a high probability of a target. These features may yield better performance through fusion.
Figure 3.8. This figure is a plot of ROC curves for two detection algorithms along with their fusion results. The SKC algorithm is shown along with the best Fourier feature, Phase:155Hz. The output of these algorithms was fused using a linear classifier (FLD Fusion) and a nonlinear classifier (SVM Fusion). The results are shown in the dashed and dotted lines, respectively. The fused decisions perform better over some parts of the ROC curve and worse in others.

The final results for all experiments, including fusion, are summarized below in Table 1 with AUC measures. Not surprisingly, the human observers perform the best overall. While the SKC algorithm does not achieve human performance levels, it does perform much better than the established Fourier features while also being fully automatic. Recall that the SKC algorithm does not utilize the red human-drawn boxes. It also has the benefit of providing estimates of the spatiotemporal location of target signatures as part of its output. This information may be very useful as an initial step for other processing in the future. These estimates could potentially allow other more sophisticated detection statistics to be extracted at the most likely target locations in space and time.
Table 1: A tabulation of the area under the ROC curve (AUC) measured for the human observers, the SKC algorithm, and the two best human-informed Fourier features. Although the SKC algorithm doesn’t perform as well as the human observers, it performs better than the established Fourier features, and does so without making use of any human-provided information about the spatial location of the target, as is the case with the Fourier features.

<table>
<thead>
<tr>
<th>Method</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean human rating</td>
<td>0.87</td>
</tr>
<tr>
<td>SKC Algorithm</td>
<td>0.84</td>
</tr>
<tr>
<td>SVM Fusion: SKC+Phase_{160Hz}</td>
<td>0.84</td>
</tr>
<tr>
<td>FLD Fusion: SKC+Phase_{160Hz}</td>
<td>0.83</td>
</tr>
<tr>
<td>Phase_{160Hz}</td>
<td>0.80</td>
</tr>
<tr>
<td>Phase_{155Hz}</td>
<td>0.78</td>
</tr>
</tbody>
</table>

### 3.6 Conclusions

In this work three detection methods were applied to discriminate between target and non-target observations on a collection of 98 acoustically stimulated seismic measurements. First, the detection performance (ROC curves) of three human observers was evaluated, providing an estimate of the performance that can potentially be achieved on this detection problem. The second evaluated the detection ability of several established features based on computing statistics on the Fourier transform of the data. Because they are simple, these features provided a performance baseline by which more complicated processing can be compared for efficacy. The last method was a new method that I proposed which involved a keypoint clustering algorithm that searched for blob like locations in the data and then clustered them to obtain final alarm locations and confidences.
The performance of each method was compared using ROC analysis and area under the ROC curve (AUC). Based on their comparison several conclusions can be drawn.

SKC outperforms established Fourier features in terms of AUC and requires no human annotations (e.g., the red human-drawn boxes used in this work for feature extraction).

SKC provides important spatiotemporal information that can be used as first step for additional processing.

Nonlinear fusion of Fourier and SKC information may be beneficial, but further study is required to confirm this.

Future study could focus on developing more sophisticated feature extraction algorithms. Human observers still outperform automated algorithms, suggesting there is more information in the data that can be utilized by automatic detection algorithms. Future work will be focused on further improving the performance of this sensor in an effort to ultimate bring its performance closer to the performance of existing state-of-the-art modalities, namely downward-looking GPR. The next chapter in this document explores my work in improving detection algorithms for another large-standoff modality, the FLIR camera. Similar to seismo-acoustic sensing, the FLIR camera has
several challenges which can be addressed to improve performance in the effort to provide a large standoff alternative to downward-looking GPR.
4. Detection with a forward-looking infrared camera

This chapter discusses my work developing BTD algorithms for use with FLIR cameras. The main objective of this work is very similar to my work in the previous chapter with the seismo-acoustic modality. FLIR represents a potential large standoff alternative to downward-looking GPR that is limited by the FAR’s of existing detection algorithms. We aim to develop algorithms that improve the detection performance (e.g., $P_d$, given a fixed FAR) of the FLIR camera so that it may eventually act as practical large-standoff detection modality.

This chapter is organized as follows. The first section describes some background about detection using data from a FLIR camera. An important characteristic of FLIR data is described in the background section, called multi-look, which will be an important motivation for the new algorithms we propose. The background section also describes a popular FLIR detection algorithm that will act as a baseline detector against which all proposed algorithms are compared. The background Section concludes with an outline of the experiments and contributions made by my work. After this, Section 4.2 describes the FLIR dataset used for all experiments conducted in this chapter. This is followed by Section 4.3, which presents a method for controlling multi-look information in the FLIR data and an experiment that uses this method to show how multi-look information is utilized by the baseline detector to improve detection performance (e.g., $P_d$, given a fixed FAR). This fact, that the baseline
detector is using multi-look, then motivates the development of methods that better exploit it to improve detection performance. New algorithms proposed for this purpose are presented in Section 4.4: the plan-view RX algorithm [58] and the Forward-looking RX algorithm. Section 4.5 then describes experiments conducted with the new algorithms and their results, showing that the new algorithms yield a performance improvement over the baseline algorithm. The last section presents some conclusions and future work.

\section{4.1 Background}

Infrared cameras form images by passively measuring the infrared energy being emitted from the ground surface. The image formation process is very similar to visible video cameras, except that the pixel intensities correspond to electromagnetic energy measured in the infrared frequency band (700nm – 1mm) rather than the visible light band. IR-based detection has received attention in recent research because it can easily be placed in a forward-looking configuration where the camera points at an angle towards the earth in the direction of travel (e.g., [8], [20], [39]). This particular configuration with an infrared camera is what is referred to as the FLIR modality. The basic FLIR data collection process is illustrated below in Figure 4.1. The camera collects video frames as the vehicle travels down the road, ultimately resulting in a 3-dimensional set of video data, indexed by time and space. As discussed in Chapter 1, in
contrast to popular downward looking modalities like downward-looking GPR, FLIR has a large standoff distance and the associated ROA benefits.

Detection with FLIR cameras relies upon taking advantage of the thermal characteristics (heat conductivity, capacity, etc) of landmines and other buried targets, which tends to be different from surrounding soils [39], [59]. Because of this difference, the presence of a target in the subsurface alters the heating and cooling processes of the soils that immediately surround it. This results in temperature differences between soils near the landmine and soils elsewhere in the ground. This difference is captured in the IR imagery and buried targets typically manifest themselves in the imagery as anomalies: specifically either as bright or dark regions. Further, because only the soil directly surrounding the buried target is affected by its presence, the anomalous imagery will tend to emulate the shape of the buried target itself. An example of a bright target signature is shown in Figure 4.1 in the example infrared image on the right.

![Diagram](image.png)

**Figure 4.1.** This figure illustrates the way data is collected using a FLIR camera. The FLIR camera is typically mounted on a vehicle in a forward looking configuration as shown on the left and collects video frames as the vehicle drives down the road. A target is indicated by the blue circle in the ground. The data consists of a sequence...
(video) of infrared images as shown on the right. In the top right there is an example infrared image with a box around a target signature. The target appears as a bright circular disk.

4.1.1 The multi-look property of forward-looking infrared data

FLIR data has several unique characteristics that present challenges for detection algorithms, one of which is termed multi-look. Multi-look refers to the fact that IR data from each physical location in a scene is usually recorded in multiple frames of the FLIR video, each time at a different pose relative to the camera. In other words, each time a subsurface object is imaged, its position and orientation relative to the camera is different. The result of this is that an object in the scene appears differently in each of the images in which it is recorded. Each sub-image (within a single video frame) corresponding to an object is referred to as a single look at that object (e.g., [19], [20], [23], [60]). This phenomenon is illustrated below in Figure 4.2. It should be noted that there is no rigorous definition of a look provided in the literature, however, for the experiments presented later in this chapter it will be important to provide more precision to its meaning. Section 4.3.1 therefore describes the meaning of a look that is adopted in this work for the purposes of experimental design and discussion.

This multi-look characteristic of FLIR presents challenges for algorithm development that may contribute to lower detection performance (e.g., $P_d$, given a fixed FAR). One difficulty is the changing shape and size of target signatures in the video. Many algorithms applied to the FLIR-detection problem are adapted from standard IR detection techniques, and many of them make some assumptions about the size and
shape of targets. Some examples of this include morphological operators [61], multi-scale blob detectors [8], maximally stable extremal regions [20], and the popular RX algorithm [20], [34], [39]. These assumptions are less appropriate for the multi-look FLIR data where the target signatures change shape and size in each frame.

Another difficulty (and potential opportunity) is raised by the existence of multiple looks at each object. There is evidence that combining multiple looks together can improve detection performance [8], [39] however, until my work on the topic (Section 4.3 below), there was no explicit investigation of this effect. It is also unclear if additional looks are useful when the camera is far away from the object, for example when a large standoff distance is used.

My research has focused on developing new algorithms that address these particular challenges of FLIR data in order to improve detection performance. This work significantly leverages an existing detection algorithm that has been applied in numerous FLIR BTD studies. This algorithm will act as the baseline algorithm against which all other proposed methods are compared. Before describing an outline of this work in detail, it will be important to review the basic functionality of this baseline algorithm.
Figure 4.2. An example of multi-look information that is present in FLIR video. A bright blob corresponding to a target appears in different locations of the video frames (image-space, left) however all points correspond to roughly the same location on the earth (world-space, right). The black boxes illustrate the notion of multiple looks at the object. Notice that the object appears at a different size and shape in each look (image).

4.1.2 The baseline detector: Mean-shift RX

The baseline FLIR detection algorithm consists of two main parts: an RX detection algorithm (see Section 2.2.2), and an alarm clustering operation with the mean-shift algorithm (see Section 2.4). For this reason the detector is referred to here as the mean-shift RX (MSRX) algorithm. Figure 4.3 below shows an illustration of how the MSRX algorithm operates. First, each video frame is filtered with the RX detection algorithm (left), which yields a list of alarms for each image (center). As a reminder, the RX filter is characterized by a single target filter size and single background filter size which is used across all images and camera perspectives. Addressing this limitation is the main objective of the proposed algorithms in subsequent Sections.
Once the RX alarms are extracted from each image, the coordinates for each alarm location in image-space are mapped to their corresponding locations in world-space coordinates. This mapping is achieved using the perspective projection mappings discussed in Section 2.5.2. In this work Universal Transverse Mercator (UTM) coordinates are used to identify locations in world-space. Due to the multi-look property of FLIR data, many alarms from different video frames will correspond to the same object (see Figure 4.2). When they are mapped into world-space they will map to nearby locations as shown in the right in Figure 4.3. To exploit this fact, a mean-shift clustering operation is performed in world-space to identify locations where many alarms are located. All individual alarms from the images are then discarded and replaced with the mean-shift cluster centers. These centers become the alarm locations output by the MSRX algorithm. The decision statistic for each cluster center is computed by summing the decision statistics for all of its member alarms. The assumption of this clustering operation is that targets will tend to appear consistently across many video frames and will have strong clusters (many alarms with high decision statistics) in world-space. Clustering and summing alarm confidences then takes advantage of this multi-look effect to improve performance. In subsequent experiments we show that the mean-shift operation does indeed exploit this effect.

The MSRX algorithm was chosen as the main focus of my work because it is a main component in many FLIR BTD algorithms proposed in the literature [19], [23], [34],
Additionally, mean-shift and RX, the two main components of this algorithm, are well-established in the broader signal processing and machine learning literature (e.g., [37], [45], [62]). Any improvements to these methods are likely to be useful for the broader research community.

**The baseline FLIR detector**

Figure 4.3. This figure illustrates the basic operation of the baseline FLIR detector used in the work presented in this Chapter. It begins with a standard RX detector (left) which yields a list of alarm locations in image-space (center). It should be noted that the RX filter is characterized by a single target filter size and single background filter size which is used across all images and camera perspectives. The alarm coordinates in image-space are then mapped into world-space (GPS) coordinates where they are clustered using the mean-shift algorithm. The alarms output by the algorithm consist of the cluster centers found by mean-shift. The decision statistic for each cluster is computed by summing the decision statistics of its member alarms. In this way, the mean-shift algorithm attempts to exploit multi-look information available in the FLIR data to improve performance.

### 4.1.3 Overview of this work

The broad goal of the work presented in this Section is to improve the performance of FLIR sensor based detection. The strategy for doing this is to improve the way the MSRX detection algorithm exploits multi-look information in the FLIR data. The first step in this process is to ensure that MSRX does indeed use multi-look
information. Heuristically, the purpose of the mean-shift operation in MSRX is to aggregate the multi-look information across FLIR images to improve detection performance, as measured by ROC analysis. Although it is established that the mean-shift operation improves performance over a stand-alone RX detector [8], [39], it is unclear if it is actually due to the utilization of multi-look information. Section 4.3 presents an experiment that shows that this is indeed the case.

Section 4.4 then presents two new algorithms, forward-looking RX (FLRX) and plan-view RX (PRX), which are intended to replace RX in the MSRX algorithm to better exploit multi-look information in FLIR data. The main idea motivating both of these algorithms is to better account for the varying shape and size of target signatures in FLIR data. Recall that RX requires the algorithm designer to specify two filter sizes: a target filter size and a background filter size. These filter sizes are chosen based on the assumed size/shape of target signatures in the data. Any mismatch between the filters and the actual target signatures is likely to be detrimental to detection performance. Clearly one filter size is not well suited for the changing target signatures that appear in FLIR data. FLRX and PRX mitigate this problem by altering the RX filter shapes or altering the data before it is processed by RX, respectively. The presentation and testing of these new algorithms is the focus of Section 4.4 and Section 4.5 respectively.
4.2 The forward-looking infrared dataset

The experiments conducted in this work use an FLIR video dataset provided by the Night Vision and Electronic Sensors Directorate (NVESD) of the United States Army. It has been used in several previous studies for detection algorithm development [19], [34], [39]. The video data was collected using an uncooled Long Wave IR camera with a spectral response from 8-12 micrometers at fifteen frames per second. Each frame has 8 bits per pixel and a resolution of 640x480 pixels. The camera was hard-mounted on a vehicle along with a differential GPS receiver and an Inertial Measurement Unit (IMU).

The imagery was collected at an arid United States Army test site over two lanes. A total of 6 passes were made over each lane under varying conditions: time of day, weather, and driving direction. A series of high-metal and low-metal target objects were buried in the lanes at various depths. In total the FLIR video data contains 534 target encounters over a total of 43724 m² of lane area. The GPS coordinates of each target were recorded for scoring purposes. This dataset represents a modified version of the dataset used by Stone et al., [39], and more details can be found there.

4.3 Experiment 1: Establishing the benefit of multi-look

This section describes an experiment to test the hypothesis that multi-look information is explicitly utilized by the mean-shift clustering step in the MSRX algorithm to improve BTD performance. This will be accomplished by measuring the performance of MSRX as the quantity of multi-look content in the FLIR data is varied in
a controlled fashion. Thus far the term ‘multi-look’ has been used in a very qualitative way, however, in order to control multi-look content it will be necessary to have some more precise definition of what one look is, and what it means to add another look to the FLIR data. This definition is provided in the next section. This is followed by a description of a method for isolating individual looks in the FLIR dataset so that looks may be added or removed from the data in a controlled fashion. The last section presented an experiment where the performance of MSRX is measured as the multi-look content of the FLIR dataset is varied.

### 4.3.1 Defining a look

As discussed, the term multi-look refers qualitatively to the fact that each object in the scene appears in the video multiple times. For the purposes of the experiments here however, it will be necessary to assign a more precise definition to multi-look, and what it means to add or remove a look from a video. For this purpose we define a look as a collection of images such that, collectively, they form a video wherein every object in the scene appears exactly once. A look therefore must be defined with respect to some scene area on the earth. In this context, the scene refers always to the lane over which the detection system is being driven and being scored. Creating a single look can be achieved relatively simply using the FLIR dataset presented in the previous section, and this is described in the next section.
Now that a look is defined, it is possible to describe what it means for a video to contain \(N\) looks at the scene (multi-look). Let the variable \(N\) denote the number of looks in a multi-look video. If a video contains \(N\) looks at the scene then it means that every object in the scene appears exactly \(N\) times in the video: no more and no less. A necessary condition for this to be true is that the video can be partitioned into \(N\) disjoint video sequences where each video constitutes a single look at the scene. An illustration of how a video with multi-look might be partitioned into its constituent looks is illustrated in Figure 4.5 below and is described in the next section.

4.3.2 Isolating individual looks in the FLIR data

In this section, a method of isolating the individual looks available in the FLIR video dataset is described. This method provides the control needed for the controlled multi-look experiment and its subsequent analysis. Again, in this context all looks will be defined with respect to the lane detection scoring area. The question now becomes how to partition the FLIR video in such a way as to create individual looks at the lane. In other words, for each look we need to choose sub-images from the FLIR video in such a way that together they form one look at the lane. The method presented here for choosing this sub-image is based on an important key observation: that a complete look at the whole lane is guaranteed by choosing, from each video frame, a sub-image which corresponds to the lane area passing underneath the camera and out of view.
To understand this, consider two frames of a camera and their respective fields of view. This is illustrated in Figure 4.4 below, where a side profile of the FLIR camera is shown at two instances in time. The two arrows pointing towards the ground from the camera indicate the camera’s field of view (FOV) along the direction of travel. The FOV of the camera at time $t + 1$ is shifted forward by the distance the vehicle travelled since the time $t$. This causes the FOV to shift by the same distance, and for some of the ground area to pass under the camera and out of the FOV permanently. This ‘escaping’ region is illustrated in image-space by the shaded region of the image at time $t$. If this shaded region of image is retained as a sub-image, it corresponds to a single image of all the lane area that passed out of view between the frame at time $t$ and $t + 1$. Now extrapolate this sub-image capture process if the camera were to continue moving along the entirety of the lane at a constant velocity; then only the surface area passing out of view will be captured in each frame. Each area of the ground will only pass out of view once so therefore it will only be captured once (assuming the vehicle doesn’t go backwards). This ensures that all lane area is captured just once. Repeating this procedure at each frame therefore ensures that every object in the lane is captured exactly once.
Figure 4.4. This figure illustrates the method used for isolating a single look at the lane. As the vehicle moves forward from time $t$ to $t + 1$, the highlighted region of the ground passes under the camera. If this part of the image is retained at every time, it guarantees a complete look at the lane.

This process of isolating a single look from the FLIR dataset is illustrated in Figure 4.5 below on actual FLIR data. On the left side is a set of images corresponding to the FLIR video dataset. Using the method just described, the light blue area called Region 1 can be retained in every frame of video as shown and it will create one *look* at the lane. This can be further understood by looking on the right of the figure: if the blue region is mapped into world-space using a perspective projection, it will cover the entire lane exactly once. In this way, the mapped regions create a “cover” of the lane in world-space.

This process can be used to create another *look* at the lane whose sub-images are disjoint to the first *look*. This can be accomplished if the process is repeated after the sub-images corresponding to the first *look* (Region 1) are removed from consideration.
This is illustrated with the area denoted as Region 2. This process can be continued until all of the image data is allocated to exactly one look. Partitioning the video data in this way will be the method by which multi-look content is controlled during detection. The video will be partitioned into disjoint looks at the lane, and a select number of looks will be fed to the MSRX algorithm for performance estimation.

Figure 4.5. This figure provides an illustration of the regions used from the FLIR video frames to construct disjoint looks at the lane. There are eleven total regions shown here, although more can be made. The FLIR video data (left) is partitioned into 11 collections of sub-images. One collection of sub-images corresponds to taking a sub-image from every frame of data that is the size of Region 1. If this operation is performed across all images in the video, it results in one look at the lane.
4.3.3 Experimental design

Now that disjoint looks can be constructed, they can be included for detection in a controlled fashion. To test the hypotheses for this experiment, it will be sufficient to begin with 1 look and measure performance as additional looks are added. In other words, detection is performed using only the image data in Region 1 in each of the video frames. Next, progressively higher index regions (looks) are included into the detection process, so that detection is performed on an increasingly larger region of the FLIR frames. To describe this process, the Depth Index parameter, $k \in \{1, \ldots, 11\}$ is introduced, where $k = K$ indicates that all image data from region 1 up to region K was used in detection. In this approach the pixels closer to the camera are incorporated first, and more pixels are taken into consideration as $k$ increases. When $k = 11$, all of the regions are used. This value was chosen because it was sufficient for addressing the experimental goals of this work. In particular, increasing this value did not yield improve performance for any algorithms. Since each region was designed to model a separate look at the scene, direct evaluation of the value of multi-look data as a function of the distance to the camera can be assessed.
An illustration of depth index, $k$, on a forward-looking infrared image

Figure 4.6. This figure provides an illustration of the Depth Index parameter $k$. If $k = K$ then the regions from 1 up to $K$ are used for detection. Broadly speaking, the value of $k$ corresponds to the number of looks utilized for detection.

Performance in these experiments is measured with a pAUC score and is computed by measuring the area under the ROC curve between 0 and 0.02 false alarms per square meter (see Section 2.1.2). This is a typical operating range for landmine detection and has been used previously [19], [23], [63]. All pAUC values were scaled by the maximum possible pAUC to be between 0 and 1. The convention was used that any alarm located within a 1 meter halo of the world-space target location was considered a true detection. This is also a convention used in previous FLIR detection studies [60]. If multiple alarms are located within a single halo then the maximum alarm statistic across all included alarms is used. All other alarms were considered false alarms.
Two detectors were used to perform detection in the experiment described above: a stand-alone RX detector and the MSRX detector. The RX detector size was optimized over the entire dataset. By convention in the literature [60], the mean-shift algorithm used in these experiments utilized a uniform kernel and its neighborhood size was determined via lane-based cross-validation. A cluster-size threshold was also learned via cross-validation. This removed clusters if they contained a number of RX alarms below the learned threshold.

4.3.4 Results and discussion

The results of this experiment are shown below in Figure 4.7, where the performance results from each detection algorithm are plotted as a function of \( k \). The RX algorithm is shown in a solid blue line and the mean-shift of the RX alarms, referred to as MSRX is shown in light blue. The RX algorithm performance initially outperforms MSRX but is exceeded after \( k = 2 \). The RX algorithm performance begins decreasing steadily after \( k = 2 \). The MSRX algorithm performance improves rapidly up to \( k = 4 \), after which it tends to decline.

There are several important implications of these results. The first is a confirmation of the hypothesis that mean-shift is utilizing multi-look information to improve performance. This is indicated by the fact that mean-shift initially performs poorly and as additional looks are added, in a controlled fashion, the performance of MSRX improves. This is further validated by the performance of RX without the
mean-shift operation. Aside from the first look, the performance of RX steadily declines as more looks are added because RX cannot exploit them. The additional image data mostly serves to create more false alarms for RX, whereas it is exploited by MSRX to improve performance. It is also notable that MSRX doesn’t outperform RX when only one look is available in the data. In this case the mean-shift step is actually detrimental to performance. This is further evidence that mean-shift is utilizing the multi-look information, because otherwise it performs worse than the stand-alone RX algorithm.

The other important implication of these results is that the multi-look information may not always be useful. Although MSRX improves in performance at first, its performance quickly levels off and then eventually begins to decrease, although it is difficult to say if this is a downward trend at $k = 11$. It is plausible that performance may degrade when $k$ becomes large because the looks collected at greater standoff are less informative. Performance also may degrade because the RX filter size is not designed to operate well at large standoff. This motivates the development of the PRX and FLRX algorithms that are described in the next section. These algorithms try to increase exploitation of multi-look by altering the processing so that RX can take better advantage of looks that are collected at varying standoff distance.
Figure 4.7. This figure shows the pAUC scores for RX (solid) and MSRX (dashed) algorithms on the FLIR data. Performance was computed as a function of the number of looks that were used, indicated by the Depth Index parameter, $k$.

### 4.4 Improving multi-look processing

This section presents two algorithms that were designed to improve the use of multi-look information in the MSRX detector. Both algorithms attempt to modify the standard RX detection algorithm (see Section 2.2.2) to adjust for the changes in target shape/size as a function of the camera perspective. Recall that RX requires the algorithm designer to specify two filter sizes: a target filter size and a background filter size. These filter sizes are chosen based on the assumed size/shape of target signatures in the data. Any mismatch is likely to be detrimental to detection performance. It is
clear that any single filter size specification therefore cannot be appropriate for FLIR because each target changes size and shape across frames as the camera perspective changes. The PRX algorithm attempts to mitigate this problem by transforming all of the data into a common camera perspective (world-space perspective) where targets sizes no longer change as a function of the camera perspective from which they were collected. Then RX detection can be performed using a single filter size. The FLRX algorithm takes an alternative approach by performing detection in image-space but changing the RX filter sizes as a function of the image-space location which is being filtered.

4.4.1 The plan-view RX algorithm

The goal of the PRX algorithm is to transform the FLIR image data in such a way as to remove the size/shape change that occurs for each scene object as the camera perspective changes. One way to achieve this is to use the geometric camera models described in Section 2.5 to map all of the image data (pixels) into a common reference frame, in this case world-space, and then interpolate the pixels over a uniform grid in that space. Once the data is transformed in this way, a single set of RX filter sizes will work well regardless of the camera perspective from which any object was imaged. One potential problem that arises with this processing strategy is that substantial portions of the image data will map to the same world-space location. This is the same thing that occurs with detector alarms when they map to (roughly) the same world-space
locations. One way to address this issue is to simply map all of the pixel coordinates into world-space and then interpolate over one uniform grid. This will yield a single image of the entire lane in world-space.

Another approach, and the one taken here, is to make several images, called ‘plan-views’ here, corresponding to each look at the lane. This is a straightforward extension of the processing performed in the previous experiment because the pixels needed to form each look (plan-view) are already known. Figure 4.8 below illustrates how this process works as well as what plan-views look like for each look of the FLIR video. This also fits well into the MSRX algorithm. RX detection can be performed separately on each plan-view, and the resulting alarms can then be clustered directly in world-space with mean-shift. The only step that needs to be performed is interpolation of the pixels once they are mapped into world-space. Notice in Figure 4.8 that the target (bright blob) appears at roughly the same shape and size in each of the world-space images. This is the intended effect, allowing one RX filter to be appropriate regardless of the perspective at which the targets are imaged originally by the camera.
Figure 4.8. This figure shows some of the results of plan-view processing on the FLIR dataset. Each of the eleven disjoint FLIR video regions (left) is used to construct a unique plan-view (right). These plan-views are then used for detection.
4.4.2 The forward-looking RX algorithm

The design of the FLRX algorithm was motivated by the large computational costs of PRX processing. As discussed, PRX transforms the video data to account for the effects of the changing camera perspective. To do this, all of the FLIR data (pixels) are mapped into world-space and then are interpolated over a grid. All of this must be performed as the data is collected, and before any standard RX detection processing can take place. This creates a large and impractical computational cost. FLRX is designed to achieve approximately the same processing but to do it in a computationally less expensive manner that is also better suited to real-time applications.

FLRX takes an alternative perspective on the plan-view processing approach. While the plan-view transforms the FLIR data to account for the camera perspective, the FLRX algorithm involves transforming the RX filters to account for the changes in perspective. In other words, the FLRX filter shapes and sizes are adjusted depending on the pixel on which they are centered so that, effectively, the image objects no longer vary as a function of the camera perspective. To do this, the confidence of each pixel is computed by a unique background and target filter shape and size.

The key observation for designing all the image-space filter shapes is that they must all correspond to the same size/shape in world-space. This idea is best illustrated with a single example RX filter. Consider a square RX filter centered over some location \((x, y)\) in world-space. The objective of PRX is to compute a sample mean over the world-
space area that the RX filter shape covers. Now assume that this area of the ground surface in world-space is imaged using the FLIR camera, and that the resulting video pixel intensities are available. The intensity of each pixel can be modeled as a sample mean computed over some small region of world-space. Some of these pixels will correspond to parts of world-space covered by the RX filter at \((x, y)\). If these assumptions can be made, it will be shown that there are some subset of the image pixels that, together, cover the area of the world-space RX filter. It is possible to compute the sample mean over this world-space area using an appropriately weighted sum of the covering image-space pixels. This is the mechanism by which each FLRX filter is designed. This will be shown mathematically in the next section. An important consequence of this design is that a different set of filter weights is needed for each pixel that is filtered in image-space.

There are several important consequences of this approach. Since (in general) each pixel in the video has a unique filter associated with it, the filter creation process is computationally very expensive; however, in contrast to processing in the plan-view, filter creation can be performed offline before filtering for detection. This makes it much better suited for real-time online applications than the plan-view RX detector. Also, it will be shown shortly that under some circumstances the need for a unique filter for each pixel can be relaxed.
Since the filter creation can be performed offline, the algorithm is only limited by the speed of the filtering process. Although this makes it very similar computationally to the original RX algorithm, the FLRX algorithm may not always achieve the same speeds. This is because many image filtering algorithms, such as those using the fast Fourier transform, require fixed filter shapes. This is not possible for FLRX and thus it cannot take advantage of such algorithms. Nonetheless, the computational cost of the FLRX algorithm is substantially better suited for real-time detection applications than the plan-view approach.

4.4.2.1 The forward-looking RX algorithm

The FLRX filter relies on modeling image pixels as averages of the infrared energy over trapezoidal regions of the earth (world-space). This approach will be introduced here before the FLRX algorithm is presented. Let us define each image pixel, \( p_{i}^{xyt} \), as a square region of the image-space plane. The subscript denotes that the pixel resides in the image-space frame of reference. The superscripts \( x \) and \( y \) indicate the image-space coordinate of the pixel center location, and the superscript \( t \) denotes the frame of the video in which it exists. Note that the pixels themselves are subsets of the real plane, \( \mathbb{R}^2 \), and that the superscripts are simply an indexing system. This distinction is illustrated below in Figure 4.9. To distinguish operations on sets versus operations on points we will use the convention that capital symbols refer to sets and lower-case symbols refer to points, unless otherwise specified.
Figure 4.9. This figure provides an illustration of a pixel in image-space. A pixel is a square subset of the real image-space plane, $P_{i}^{xyt}$, where $(x, y)$ is the center location of the pixel and $t$ is the video frame in which the pixel exists.

This image-space set can also be mapped into a corresponding world-space set, $P_{w}^{xyt}$, using perspective projection operators. As defined in Section 2.5.2, the perspective projection is a point-wise operation and generally would not be able to operate on sets. Fortunately, however, perspective projections have the property that any straight line is preserved through its mapping [45]. The pixels are all parameterized by four corner points, or equivalently, the four lines connecting these four points around the perimeter of the pixel. As a result, in order to map a pixel, we need only to map the four coordinates corresponding to the four corners of the pixel. We can map these four corners and then reconnect the lines between them in the same order. The new region is given by the region enclosed by the lines. For example, to transform $P_{i}^{xyt}$ to its corresponding world-space region, we need only to identify the four corner points of $P_{i}^{xyt}$, transform those points into world-space, and then reconnect the lines as they were connected in image-space. Since the corner points can be determined by knowledge of
the pixel index \((x, y)\) then this can be easily performed for any pixel. This is illustrated below in Figure 4.10.

Figure 4.10. This figure illustrates the process of mapping a pixel from image-space (left) to world-space (right). This is performed using the perspective projection operators described in Section 2.5.3.

4.4.2.2 Creating FLRX filters

Now that the basic conception of FLRX has been established, a method for actually designing the FLRX filters can be presented. The key observation for designing the filters is that they all correspond to the same size in world-space. The FLRX filters need to be designed so that they effectively compute the \textit{world-space} mean and variance computations over this fixed-size area, but using only the pixels in \textit{image-space}. Towards this end it will be shown that, given some reasonable assumptions about the image data, this is indeed possible. First the method for computing a world-space mean-value by
using the image pixels will be outlined, and then all of the new FLRX filtering equations are defined.

The main assumption required for the FLRX algorithm is that the intensity of each image pixel represents the average intensity that would be measured over the area of the earth spanned by that pixel’s world-space projection, \( P^{x,y,t}_w \). It will be assumed that the intensity values in world-space are given by a continuous function \( I: \mathbb{R}^3 \to \mathbb{R} \). This function maps a location in world-space at each time (3 domain coordinates) to an intensity value. These intensities can be likened to the IR radiation emitted by each infinitesimal area of the earth measured by the FLIR camera.

Now, assume that the pixel intensities in the image are an average over the world-space intensities to which they correspond. This can be defined mathematically by a function \( i(x,y,t) \) that is given by

\[
i(x,y,t) = \frac{1}{A(P^{x,y,t}_w)} \int_{P^{x,y,t}_w} I \, dP^{x,y,t}_w.
\]  

(26)

This function gives the relationship between the intensity of each pixel in the video, \( i(x,y,t) \), and the area over which the pixel at \( (x,y,t) \) is measured on the earth, \( P^{x,y,t}_w \). Note that in (26), the notation \( dP^{x,y,t}_w \) in the integral represents an infinitesimal unit of the real plane at time, \( t \). The value \( A(P^{x,y,t}_w) \) corresponds to the area of the plane defined by \( P^{x,y,t}_w \). To be precise this area operator for \( P^{x,y,t}_w \) is given by
\[ A(P_w^{xyt}) = \int_{P_w^{xyt}} dP_w^{xyt}. \]  

(27)

As mentioned, the FLRX filter requires a filter for each pixel. These filters will be indexed by their center location, specified by \((x, y)\), over each pixel. Define the world-space projection of this filter by \(T_w^{xyt}\). This region is easily obtained by projecting the coordinate \((x, y, t)\) into world-space and then using the (assumed known) world-space target filter size to define the perimeter of the filter. Now, assume further that we can find a family of pixels \(D\) that form a world-space partition of \(T_w^{xyt}\). A partition is a family of sets whose members are pairwise disjoint and have the relationship that

\[ T_w^{x'y't'} = \bigcup_{(x, y, t) \in D} P_w^{xyt}. \]  

(28)

Note that the indices with superscripts, \((x', y', t')\), are simply to distinguish the two sets of indices being used in the equation. This assumption regarding \(D\) is a reasonable one as long as the point spread function of the camera is approximately a delta function. Using this definition the mean value over \(T_w^{x'y't'}\) can be computed using a weighted sum of the image-space pixels in \(D\).
\[
\mu(T_{w}^{x',y',t'}) = \frac{1}{A(T_{w}^{x',y',t'})} \int_{T_{w}^{x',y',t'}} I dT_{w}^{x',y',t'}
\]

\[
= \frac{1}{A(T_{w}^{x',y',t'})} \left[ \sum_{(x,y,t) \in D} \int_{p^{x,y,t}} I dP_{w}^{x,y,t} \right]
\]

\[
= \sum_{(x,y,t) \in D} \frac{1}{A(T_{w}^{x',y',t'})} \left[ \int_{p^{x,y,t}} I dP_{w}^{x,y,t} \right]
\]

\[
= \sum_{(x,y,t) \in D} \frac{A(p_{w}^{x,y,t})}{A(T_{w}^{x',y',t'})} \left[ \frac{1}{A(p_{w}^{x,y,t})} \int_{p^{x,y,t}} I dP_{w}^{x,y,t} \right]
\]

\[
= \sum_{(x,y,t) \in D} \frac{A(p_{w}^{x,y,t})}{A(T_{w}^{x',y',t'})} i(x,y,t)
\]

\[
= \sum_{(x,y,t) \in D} \tau^{x,y,t}(x',y',t')i(x,y,t)
\]

This derivation begins with the basic calculation of the sample mean within the region defined by \(T_{w}^{x',y',t'}\). Then, substituting the pixel partition of the region and reordering terms yields the final equation in (34) which shows that the mean can be calculated by a weighted sum of the pixels in the partition. The weight for each pixel, given by \(\tau^{x,y,t}(x',y',t')\), is based on the area of intersection each pixel has \(T_{w}^{x',y',t'}\). This is normalized by the total area of the region, meaning that the weights sum to one. This shows how to compute a mean-value, but can easily be extended to variance computations. The variance computation is shown below in Table 2. The important observation from this equation is that the mean is now a weighted sum of image-space

94
pixel values, thus ensuring that it can be implemented as a filter. The process of creating a single FLRX filter is illustrated below in Figure 4.11.

![FLRX Filter Creation Diagram]

Figure 4.11. This figure illustrates the process of creating the FLRX filter for the pixel located at \((x, y)\). The pixels surrounding \((x, y)\) are first mapped into world-space. The filter weight assigned to each mapped pixel corresponds to the amount of intersection between the pixel and the world-space target filter \(\mathcal{T}_w^{xyt}\). The final filter is shown on the bottom right.

There are two additional considerations that need to be addressed before the FLRX filter equations can be written out explicitly. First is a simplification of the equations. The derivation outlined above assigns a different filter to each pixel location and time. In general this may be necessary but here this need not be the case. This is because the camera perspective projection matrix that is used is not time-dependent (described in Section 2.5.3). This means that, although the world-space location of pixels will change over time, their locations relative to one another do not change over time.
Therefore, given a filter center pixel, the overlap of the surrounding pixels with the RX filter in world-space does not change over time. As a result, the notation for the RX filter world-space regions can be modified to $T_{w}^{x'y'}$ and all resulting filter creation is independent of time. It is important to keep in mind that this is a simplification that can only be made due to the static perspective projection. In general the FLRX filters may be time varying.

The second consideration regarding the FLRX filters is related to the assumption above that the pixels in $D$ form a perfect partition of $T_{w}^{x'y'}$. In reality this will not be the case. Although the pixels are approximately disjoint, some of the pixels do not perfectly overlap with the filter region. This is the case for pixels on the edge of the world-space filter region. As a result, the equation for the pixel weights, given a world-space filter region $T_{w}^{x'y'}$, is given by

$$
\tau^{xy}(x',y') = \frac{A\left(P_{w}^{xy} \cap T_{w}^{x'y'}\right)}{A(T_{w}^{x'y'})).}
$$

One consequence of this change is that it is no longer necessary to specify a family of pixels $D$ that partition $T_{w}^{x'y'}$. Any pixels that do not intersect with the filter region will simply be assigned a weight of zero.

At this point the final equations for the FLRX algorithm can be presented. Thus far we have specified $T_{w}^{x'y'}$ for the target filter. Let us now specify $B_{w}^{x'y'}$ for the background world-space filter region. The pixel weights for this filter are then denoted
by $\beta^{xy}(x', y')$. With this notation, the new FLRX filter equations are shown below in Table 2. These equations are analogous to those shown for the RX algorithm in (5). Of course, they are now dependent upon the current pixel to which the filter is applied, $(x', y')$.

Table 2: Equations for the FLRX algorithm.

<table>
<thead>
<tr>
<th>The FLRX equations for image-space pixel $(x', y')$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tau^{xy}(x', y') = \frac{A \left( P_w^{xy} \cap T_w^{x'y'} \right)}{A(T_w^{x'y'})}$</td>
</tr>
<tr>
<td>$\beta^{xy}(x', y') = \frac{A \left( P_w^{xy} \cap B_w^{x'y'} \right)}{A(B_w^{x'y'})}$</td>
</tr>
<tr>
<td>$\mu_T(x', y') = \sum_{v(x,y)} \tau^{xy}(x', y')i(x, y)$</td>
</tr>
<tr>
<td>$\mu_B(x', y') = \sum_{v(x,y)} \beta^{xy}(x', y')i(x, y)$</td>
</tr>
<tr>
<td>$\sigma_B^2(x', y') = \left( \sum_{v(x,y)} \beta^{xy}(x', y')i(x, y)^2 \right) - \left( \mu_B(x', y') \right)^2$</td>
</tr>
<tr>
<td>$\lambda_{FLRX}(x', y') = \frac{\left( \mu_T(x', y') - \mu_B(x', y') \right)^2}{\sigma_B^2(x', y')}$</td>
</tr>
</tbody>
</table>
4.5 Experiment 2: testing the proposed algorithms

The experiments presented in this section are designed to examine the efficacy of the FLRX and PRX algorithms that were proposed in the previous section. This experiment evaluates the performance of three detectors: RX with meanshift (MSRX), PRX with mean-shift (MSPRX), and FLRX with mean-shift (MSFLRX). The experimental design is similar to Experiment 1, which was presented in Section 4.3. The performance (pAUC) of each algorithm is measured as the Depth Index parameter is increased. The main hypothesis being tested in this experiment is that the MSFLRX and MSPRX algorithms perform better than the original MSRX algorithm.

As before, the RX filter size for MSRX was determined by optimizing it using pAUC over all of the data. The PRX filter size was determined in the same fashion on plan-view data. FLRX filters were created using the filter sizes computed by optimizing the PRX filter. Theoretically, the PRX and FLRX filter sizes should be the same. Some examples of the target and background FLRX filters are shown below in Figure 4.12. All mean-shift parameters were again optimized using lane-based cross-validation.
Figure 4.12. This figure shows some examples of background and target FLRX filters created on the FLIR dataset. The locations of the filters are shown in the left panle of the FLIR image. They are labeled “one”, “two”, and “three” respectively.

The detection results for all algorithms are shown below in Figure 4.13. The performance of the MSFLRX detector (black) increases rapidly until \( k = 5 \), after which it tends to decline for the remaining values of \( k \). The MSPRX detector (red) behaves in a similar fashion, reaching its peak value at \( k = 6 \). The MSRX algorithm was shown in previous results and is presented here again for comparison in blue.

The main hypothesis, that the modified RX detectors perform better, is largely supported by the experimental results. With the exception of \( k = 11 \), the MSPRX and MSFLRX algorithms consistently outperform the MSRX detector. The performance improvement is relatively narrow but it is consistent as more data is added.

One counter-intuitive result of the experiment is the difference in performance of the MSFLRX and MSPRX algorithms. The MSFLRX algorithm generally outperforms the MSPRX algorithm despite their similar processing. This is likely caused by the imprecision of the plan-view creation step. The grid of points over which the plan-view
images were created was more coarse than the pixel spacing. Therefore some image information was always lost when transforming the image data, which would likely cause some degradation in performance.

Figure 4.13. This figure is a plot of the pAUC values computed for three detection algorithms as a function of the depth index parameter: MSFLRX (black), MSPRX (red), and MSRX (blue).

4.6 Conclusions

FLIR video is an important potential modality for BTD due to its ability to provide larger standoff distances than downward looking sensors. The work presented here was developed to improve the detection performance (as measured by partial AUC
of the ROC curve) of FLIR algorithms and bring them closer to that of a downward-looking GPR system. To do this, an investigation was performed of the multi-look characteristic of FLIR data. A method was developed to isolate and control multi-look information in FLIR data and it was used to confirm that a popular algorithm, MSRX, utilized multi-look information to improve detection performance. This motivated the development of new algorithms that attempted to make better use of multi-look information. Two algorithms were developed and presented, called MSPRX and MSFLRX respectively. Experiments were presented showing that these algorithms do indeed improve performance over the standard MSRX algorithm that was used from the literature.

While these performance improvements do indeed bring the FLIR detection algorithms closer to the performance of downward-looking GPR, there is still a large disparity. The next chapter presents work that pursues the same basic objective as the preceding two chapters: to improve BTD standoff. However, the strategy taken is much different. Rather than trying to replace downward-looking GPR with a large-standoff detector, a system is proposed where a high performance downward-looking GPR is combined with a large-standoff FLIR detector in order to obtain the benefits of both systems.
5. Fusion of forward-looking infrared and ground penetrating radar

The goal of the work presented in this chapter is to improve the ROA of the GPR system, however the strategy for achieving this goal is quite different from the preceding chapters. The strategy in the two preceding chapter was to develop improved detection algorithms for large standoff detection systems in an attempt to make their performance comparable to that of a downward-looking GPR system. In both previous chapter, algorithms were proposed that improved over the detection performance (as measured by ROC analysis) of existing methods, however, a large performance disparity still existed between downward-looking GPR detection systems and the proposed systems. For this reason, an alternative strategy is taken in this chapter. The strategy here was to achieve greater standoff (and thereby greater ROA) for a GPR system by combining it with a large standoff FLIR detector on a single detection system. Specifically, the goal was to yield a system that benefits from the larger standoff than the GPR by leveraging the large standoff of the FLIR, while still achieving detection performance that is similar to the original GPR system.

To achieve this outcome, the proposed detection system uses a full FLIR detection system to mark suspicious locations on the ground at a very large standoff distance. This will effectively move the responsibility of typical GPR prescreening from the GPR system over to a large-standoff FLIR detection system. Once alarms are obtained with the FLIR prescreener, downward looking GPR data can be collected at
each alarm location when the vehicle moves closer. Standoff distance for the system as a whole is altered by changing the amount of GPR data that is collected around each FLIR alarm location. The details of this methodology are described in Section 5.3. The consequence of this processing framework is a tradeoff between standoff distance and detection performance (e.g., $P_d$, given a fixed FAR), and experiments are conducted to measure the tradeoff that is achievable with the proposed system. The results show that a substantial increase in standoff distance can be achieved with a modest loss in detection performance.

The next section describes the FLIR and downward-looking GPR datasets that were used for this work. This is followed in Section 5.2 by a presentation of some methods from the literature that will be used in the proposed FLIR-GPR system. Specifically, this includes a description of the FLIR detection system that will act as the prescreener, and the GPR features that will be used for feature processing with the GPR data. Following this, the details of the FLIR-GPR system are presented in Section 5.3. Experimental results are presented in Section 5.4 and final conclusions are drawn in Section 5.5.

### 5.1 Data

The experiments in this section utilize two sets of data: GPR and FLIR data. In each case there was a set of data that was used to train any classification algorithms that was kept disjoint from the data used for testing. The testing data consisted of GPR and
FLIR data collected over the same lane. The details regarding each set are described below.

5.1.1 GPR Data

All GPR data was collected from a NIITEK 51-channel array, with 6cm and 5cm crosstrack and down-track sampling respectively. The training data was collected over several lanes at an arid western U.S. test site with several hundred buried targets of varying type and composition. Feature processing is performed using this GPR at alarm locations that were marked in advance by the F1V4 prescreener [64], [65]. This resulted in 3640 training target observations and 2533 false alarms observations. The testing data was also collected at a western US test site over approximately 4000 $m^2$ of lane area and 44 target encounters.

5.1.2 FLIR Data

The FLIR data was all collected using a vehicle-mounted forward looking infrared camera with 640x480 pixel resolution and 16-bit pixel intensity resolution. The camera was fitted with a differential Global Positioning System and an IMU unit was placed directly next to the camera in order to record the vehicle GPS and IMU information. Together this information was used to map image-space alarms into world-space coordinates using methods previously used for this data [60], [66].

The training and testing data was collected at an arid Western U.S. location. The training data was collected using several runs over a single lane with area 4000 $m^2$ with
50 unique buried targets. This data was used to train the prescreener described in Section 5.2.1 below. The testing data was collected at the same lane as the GPR data: a lane with 44 target encounters over approximately 4000 m$^2$ of area.

5.2 Background

This section describes some algorithms and features from the BTD literature that will be used as part of the FLIR-GPR system.

5.2.1 The FLIR Prescreener

The FLIR detection algorithm used here was developed in two recent FLIR landmine detection studies [19], [39] and subsequently used in numerous others studies (e.g., [23], [34]). The FLIR detector begins by filtering each FLIR image with an ensemble of size-contrast filters. Size-contrast filters are anomaly detectors that are tuned to detect anomalous objects of a particular size in the data. They are based closely on the RX anomaly detection algorithm originally proposed by Reed and Xu [28]. Size-contrast filters as used here are specified by two windows: an inner window and an outer window. The basic idea is to compare the pixels from the inner window to the pixels in the outer window, using statistics of each, to estimate how anomalous the inner region is compared to its surroundings. The inner window size is chosen based on the expected size of anomalies in the data, while the outer window is chosen to capture information about the surrounding background region. An ensemble of filters is often used to capture anomalies of different shapes and sizes. The output of the filter ensemble is a
list of alarm pixel locations and the response of each location to the size-contrast filter that detected it.

The next step in the algorithm involves two consecutive clustering operations using the mean-shift algorithm [29]. The first clustering operation is performed on size-contrast anomalies that are detected in each image. Mean-shift identifies the major clusters of anomalies that exist in each FLIR frame. This reduces the number of anomaly locations in each frame of the data. These cluster centers are then mapped from pixel coordinates into locations on the ground to which they correspond. Specifically, the pixel locations are mapped into Universal Transverse Mercator (UTM) coordinates using a well-established mapping technique applied in numerous FLIR studies [60], [66]. This places the anomalies from each FLIR image onto a common coordinate system. The second clustering operation then clusters these anomalies in UTM-space to identify locations on the earth that consistently appeared anomalous across multiple FLIR images. The final output of this algorithm is a list of alarms in UTM coordinates along with a decision statistic indicating the likelihood of a target at that location. The full FLIR detector process is shown and described below in Figure 5.1.
Figure 5.1. This figure shows an outline of the FLIR detection algorithm used in this work. An ensemble of size-contrast filters are used to filter the images and detect alarms in each frame (illustrated on left). Within each image these alarms are clustered into groups (center), and each group, indicated by a black dashed circle, is a new single image-space alarm. These alarms are then mapped from image-space (pixel coordinates) into world-space (UTM coordinates). Once mapped into world-space, the alarms are again clustered to create new alarms (right). These cluster centers are the final alarm locations designated by the FLIR detector.

5.2.2 GPR Edge Histogram Descriptors

Edge histogram descriptor (EHD) features [30] are extracted from GPR data at all of the FLIR alarm locations marked by the FLIR prescreener that was described in the preceding section [30]. This EHD descriptor is typically extracted on a patch of GPR data surrounding a prescreener alarm location. The patch of data is separated into 7 overlapping sub-images and, within each sub-image, the image gradients are computed and a histogram is created by placing each gradient into one of five categories: no edge, vertical edge, horizontal edge, diagonal edge, and antidiagonal edge. The histogram for each sub-image is referred to here as one “set” of EHD features, and typically, seven sets
are extracted (corresponding to the 7 sub-images) and concatenated to yield a full EHD descriptor for each alarm location. This process is illustrated in Figure 5.2.

Figure 5.2. This figure shows a GPR target signature (top) and how it is partitioned for EHD feature extraction. The signature is partitioned into several sub-images corresponding to each of the black boxes shown. Image gradients are extracted within each sub-image and placed into one of the 5 following categories: no edge, vertical, horizontal, diagonal, and anti-diagonal. The final EHD feature is a concatenation of the histogram for each sub-image as shown in the bottom panel. Note that the sub-images here are shown to be dis-joint for simplicity, but in actuality they overlap by 50%.

5.3 The FLIR-GPR Detector

As discussed in the introduction, downward looking GPR has excellent performance but suffers from short standoff distance. Several detection modalities exist
that offer larger standoff, but unfortunately they suffer from high false alarm rates and are therefore impractical replacements for downward looking GPR. The goal of the proposed FLIR-GPR detector is to combine the best qualities of each modality to create a single detection system with greater standoff than GPR, but with a practical false alarm rate.

Before describing the proposed FLIR-GPR system, a method is presented that can be used to increase standoff distance for a stand-alone GPR BTD system (i.e., without a FLIR). This approach was proposed in a previous investigation [15], and it is the basis of the FLIR-GPR system proposed in this work, and therefore it is beneficial to introduce it first. As described in the background (see Section 2.1.1), a typical GPR processing chain begins with prescreening. Each GPR datum is processed by a prescreening algorithm as it is received, and high confidence spatial locations are designated as alarms. Each alarm location is subsequently further evaluated in the feature processing stage. Prescreener computation time imposes that there is a delay between the time feature processing begins for a given alarm location, and the time when the GPR collects data at that location. This is illustrated below in the top panel of Figure 5.3. The left-most graphic shows the relative location of a query location (indicated by the gray circle) and the GPR sensor, at the point when feature processing begins for that query location. The yellow region indicates the spatial region over which GPR data are provided to the feature processing algorithm, for the given query location. The corresponding GPR
data is shown in the right-most graphic. The approach in [15] proposes to increase standoff distance by reducing the amount of data collected by the GPR before prescreening or feature processing begin. This is illustrated in the bottom panel of Figure 5.3. In this modified processing scheme, detection processing begins sooner, which increases the effective standoff distance by a distance indicated by the green rectangles in the left-most graphic. However, a drawback of this approach is that it also yields less data for detection processing, as indicated in the middle graphic.

**Conventional processing scheme**

![Conventional processing scheme diagram](image)

**Proposed processing scheme, to increase standoff**

![Proposed processing scheme diagram](image)

Figure 5.3. This figure illustrates the difference between a conventional GPR data processing scheme (top panel), and the approach proposed in this work (bottom panel). In the conventional scheme, the GPR for a given query location is only
processed once the GPR sensor has moved beyond the desired query location. This is illustrated in the left-most graphic in the top panel, where the query location is indicated with the gray circle. The subsurface data utilized for subsequent processing is highlighted in yellow, and the corresponding GPR data is shown in the right-most graphic. In the proposed processing scheme (bottom panel), the algorithm processing begins sooner: before all the data is collected. This increases the effective standoff distance, as illustrated by the green rectangles in the left-most graphic. However, it also yields less data for the subsequent detection processing, as indicated in the middle graphic. This reduction in available data may be detrimental to detection performance.

It was shown in [15] that substantial increases in standoff distance could be obtained with very modest losses in detection performance (e.g., $P_d$, given a fixed FAR). However, the experimental results in [15] assumed that the prescreener was provided with all data, and that only the feature processing stage was affected when standoff distance was increased. This is not possible in a real fielded GPR BTD system, and this was a major limitation of the results that were presented. In this work, a FLIR detection system is used as a prescreener to provide alarm locations for GPR feature processing, making this approach feasible for a real BTD system. This is because a FLIR BTD system collects data at a very large standoff distance already, and can provide alarm locations to the GPR system. Therefore, any increase in standoff distance will only affect the GPR feature processing, and the proposed strategy to increase standoff could potentially be used in a real system.

Although FLIR detection systems have false alarm rates (given a fixed $P_d$) that are too large to compete with a full GPR BTD system (i.e., a prescreener followed by
feature processing), the FAR of a GPR prescreener is much more comparable, and therefore it is more feasible to replace a GPR prescreener with a FLIR BTD system.

In this work experiments are conducted to examine the tradeoff between detection performance (as measured by ROC analysis) and standoff distance for the proposed FLIR-GPR system. Specifically, a FLIR BTD system was used to identify alarm locations in real field collected data. GPR data was also collected over the same lane, and GPR feature processing was applied to the GPR data at each spatial location designated by the FLIR BTD system. The EHD features described in Section 5.2.2 were extracted around each FLIR alarm location and classified using an SVM classifier like the one described in Section 2.3. The FLIR BTD system and the GPR BTD system were trained using real field collected data that was completely disjoint from the testing data.

The tradeoff between detection performance (as measured by ROC analysis) and standoff distance was examined by varying the amount of GPR data, and therefore the amount of EHD features, used by the GPR BTD system. Classification of an alarm using GPR EHD features typically utilizes seven sets of EHD features [30]. Each EHD set is extracted at a different spatial location relative to the alarm location. The first EHD set is extracted on GPR data closest to the detection system and the last set, set seven, is extracted on data furthest from the system. This is illustrated in Figure 5.2 above. As each set is removed the standoff distance is increased by the length of the GPR sub-image used for extracting those EHD features. For example, rather than collecting
enough GPR data to extract all seven sets of EHD features around an alarm, only
enough data is extracted to create the first five sets. This allows processing to begin
earlier and therefore yields an increase in standoff distance. This approach is used to
vary standoff distance of the FLIR-GPR system.

5.4 Experimental Results & Discussion

This section describes experiments performed in order to investigate the tradeoff
between detection performance (as measured by ROC analysis) and standoff distance
that is achievable using the FLIR-GPR system described in Section 5.3. Detection
performance is measured as standoff is increased (i.e., fewer EHD features are used for
classification).

**FLIR-GPR Experiments**

![Diagram](image)

*Figure 5.4. This figure provides an outline of the experiments conducted in
this study. The FLIR and GPR testing data are shown on the left as input. The FLIR
data is prescreened using the prescreener described in Section 5.2.1. This yields alarm
locations (bottom left) where GPR EHD features are extracted from the GPR testing
data (center). Several individual classification experiments are run, each using a
different number of EHD feature sets: ranging from 1 set to 7 sets. Changing the
number of sets used corresponds to changing the system standoff distance. These*
testing features are then classified using an SVM and then scored (right). The goal of the experiments is to illustrate the tradeoff between increasing standoff distance and detection performance.

In order to classify EHD feature vectors of different lengths, a separate SVM was trained using the different subsets of EHD features. All parameters and algorithms were trained using the training data described in Section 5.1. Likewise, testing was conducted on the co-located GPR and FLIR data described in the same section. ROC curve analysis (see Section 2.1.2) is used to evaluate the performance of the FLIR-GPR system using each specified subset of EHD features. The performance of two other algorithms is also reported for reference and discussion. The detection performance of the FLIR prescreener was scored as a stand-alone detection algorithm and reported. Additionally, a standard GPR prescreener for the NIITEK radar, called F1 [5], was run and scored on the testing data so that the FLIR-GPR system could be compared to existing state-of-the-art GPR processing systems.

The ROC curves from all of the experiments are shown below in Figure 5.5. The dashed red and black lines correspond to the FLIR prescreener and the GPR prescreener respectively. The green lines each correspond to using different subsets of the EHD features extracted at the FLIR alarm locations. From the results, it is clear that using more EHD features yields better performance; however, the performance saturates very quickly. The performance using three sets of EHD features is very similar to the
performance using five or seven sets. In most cases only a small performance loss is incurred per unit of added standoff distance.

Figure 5.6 summarizes the experimental results using a partial area under the ROC (pAUC) measure as compared to the additional standoff that each EHD feature subset uses. The y-axis shows the partial area under the ROC curve (pAUC) corresponding to the area between 0 false alarms per meter squared FA/m² and 0.02 FA/m². The x-axis shows the increase in standoff distance achieved. The left-most point uses all seven sets of EHD features while the right-most point uses just one set of EHD features (the most added standoff). Each EHD feature set uses 75 cm of GPR data, and each EHD feature-set overlaps with its neighbors by 50%. Therefore, the removal of one set of EHD features adds 37.5 cm of standoff. Note that the GPR prescreener has no added standoff distance. The results show that greater standoff distance may be achieved with a relatively low loss in performance per unit of standoff distance added. The first two sets of EHD features can be removed without almost any noticeable performance loss. Further, performance doesn’t drop below 98% of that of the full-feature set until only two or one of the seven EHD feature sets are used.
Figure 5.5. This figure shows ROC curves for the different detectors that were applied to the FLIR-GPR dataset. The dashed lines show the performance of the two prescreeners. The black dashed line shows the performance of a set of alarms detected by a standard GPR prescreener on GPR data. The dashed red line shows the performance of a set of alarms detected by a standard FLIR prescreener on the same dataset using FLIR data. The 4 solid lines show the performance of an SVM classifier trained on EHD features extracted at the FLIR alarm locations. Each line corresponds to the performance achieved using some subset of the available EHD features at that location. As more EHD features are used, the performance increases as expected. However, there is a large disparity between the best performance achieved by the EHD features extracted from the FLIR alarms and the performance of the GPR prescreener.
Figure 5.6. This figure compares the performance of all detection algorithms. The performance metric (y-axis) is pAUC, which is computed between a FAR of 0 and a FAR $0.02 FA/m^2$. The blue line corresponds to the performance of the GPR EHD features as standoff distance is increased. The left-most blue point corresponds to using all 7 sets of EHD features, while the right-most blue point corresponds to using just a single EHD feature set. Standoff distance increases as fewer EHD features are used, at the cost of lower detection performance (as measured by pAUC). However, the performance loss incurred from removing each EHD set is very low for the first several sets, making it feasible to increases in standoff with the system. The black dashed line is the performance of the FLIR prescreener as a stand-alone detector. A major limiting factor of the FLIR-GPR system overall is the performance disparity between existing GPR prescreeners and full FLIR detection systems. The red dashed line is the performance of a standard GPR prescreener as a stand-alone detector. Note that the FLIR prescreener has a very large added standoff distance (off the plot on the right side), while the GPR prescreener has 0 cm added standoff.

Although the FLIR-GPR system has the potential to achieve improvements in standoff distance, its practicality is fundamentally limited by the performance of the FLIR prescreener. This system is only realizable with a large standoff prescreener such as the FLIR detector considered here. It cannot work with existing short standoff GPR
prescreeners. Unfortunately, the FLIR detection system used here performs far worse than the GPR prescreener. This limits the performance of the subsequent feature processing, and ultimately limits the performance of the “fused” FLIR-GPR system. The FLIR prescreener considered here operates at a detection rate of 90% and still has a much higher false alarm rate than the basic GPR prescreener. This disparity is illustrated below in Figure 5.7. Although better performance might be achievable with more sophisticated existing FLIR processing, the FLIR detector used here is representative of performance disparity that exists between FLIR and GPR prescreening. The FLIR-GPR fusion system proposed here will ultimately require the development of FLIR processing that is comparable in performance to existing GPR prescreeners.

Figure 5.7. This figure shows the alarm locations for the GPR and FLIR prescreener plotted on the testing lane, along with target locations. Black circles
indicate alarms from the GPR prescreener, red circles indicate FLIR detector alarms, and blue unfilled circles represent target locations. This illustrates how many more false alarms the FLIR prescreener incurs as compared to the GPR prescreener. This is despite the fact that the FLIR prescreener is operating at a 90% probability of detection while the GPR prescreener is operating at near 100%. This performance gap must ultimately be bridged in order for a large standoff FLIR-GPR system, like the one proposed here, to be a practically feasible detection system.

5.5 Conclusions

In this chapter, the goal was to develop a detection system with performance similar to that of a downward-looking GPR but with greater standoff. This goal was consistent with previous chapter but the strategy was different. Rather than improving the performance of a large standoff detector, this chapter proposed a system for fusing two different sensors with complimentary detection characteristics. Specifically, a system was proposed for combining a large-standoff FLIR detector and a high detection performance (i.e., large partial area under the ROC curve, pAUC) GPR detector. The FLIR detector was used to mark locations on the ground where GPR data was then collected for further processing. Edge histogram descriptor (EHD) features were extracted from the GPR data and then classified with an SVM. Using this framework, it was possible to increase the detection standoff distance by reducing the amount of GPR data collected around each FLIR alarm location. However, if less data is collected it resulted in fewer EHD features available for classification, and this tended to result in lower detection performance (i.e., worse pAUC). Experiments were conducted to investigate the tradeoff between added standoff distance and detection performance that
is achievable using this system. The experimental results suggest the following conclusions:

- The proposed FLIR-GPR system offers a potential method of obtaining increased detection standoff distance.
- The proposed FLIR-GPR system can increase standoff distance substantially without much loss in detection performance (as measured by pAUC).
- There is a large detection performance gap (as measured by pAUC) between existing FLIR detectors and the existing GPR prescreeners that must be bridged for the proposed FLIR-GPR system to be practical.

As the detection performance (as measured by pAUC) of FLIR-based systems improves, the FLIR-GPR system may eventually become a practical method for increasing standoff. However, it is unclear when it, or any large standoff modality, will achieve performance comparable to that of a GPR prescreener. One of the primary limitations of the fusion approach here is that, much like many BTD fusion methods, it requires that all system sensors be operating simultaneously. This limits the overall system standoff (or proportionally its velocity) to be no greater than that of the sensor with the shortest standoff. In the next chapter, a similar FLIR-GPR BTD system is studied, but is operated so that the GPR is only activated under certain conditions to
improve detection performance, and otherwise remains inactive to increase system standoff and ROA.
6. A new multi-sensor management strategy for buried target detection systems

In the last chapter, a multi-sensor BTD detection system was proposed that combined a FLIR with a (downward-looking) GPR system in order to effectively increase the standoff of the GPR system, and thereby improve its ROA. One of the primary limitations of this approach was that both the GPR and FLIR system were required to be operating at all times. This requirement limits the system standoff to be no greater than the standoff of the GPR. In this chapter, a new approach is taken where the sensor activation is allowed to vary over time. For example, the GPR may temporarily deactivate under certain conditions are encountered, until which time the standoff and ROA of the system are effectively increased. The GPR can likewise reactivate when needed so that the system can exploit the detection performance (e.g., high $P_d$, given a fixed FAR) of the GPR.

The key idea behind the proposed strategy is to split the detection system operation into multiple states. Consider a FLIR-GPR system with two states of operation. Operation begins in the first state where only the FLIR operates and the system maintains a high ROA. When the FLIR detector identifies a suspicious subsurface location the system slows down and activates the GPR for improved detection performance. Once GPR processing is applied to the suspicious location, the GPR deactivates and the system again increases its ROA. In theory, this approach allows the system to travel with a greater average ROA than the GPR system, while still
benefiting from the GPR detection performance advantages. This is illustrated in Figure 6.1 in the right-most graphic.

Figure 6.1. The panel on the left illustrates a 2-sensor BTD system that combines a large standoff FLIR detector with a short standoff GPR. The FLIR operation benefits from large standoff and high ROA, while the GPR operation mode has much better detection performance (e.g., a high $P_d$ given a fixed FAR) and a lower ROA. The panel on the right illustrates a new sensor management strategy where the system ROA is high with the FLIR only (GPR deactivated) until a suspicious location is identified by the FLIR BTD system, at which point the vehicle slows down and activates the GPR system to achieve better detection performance.

Although the proposed strategy offers potential benefits, it is more difficult to analyze theoretically. The most basic problem is that the system velocity varies over time, making it difficult to compute its ROA a priori. Another problem is that the ROA is dependent upon the system FAR, which in turn depends upon the overall system probability of detection, $P_d$. As $P_d$ increases then so does the number of false alarms, and therefore the system stops more often and the ROA becomes lower.

Another problem is that, given a desired system $P_d$, it is unclear how to choose the two individual sensor detection probabilities ($P_{d1}$ and $P_{d2}$ shown in Figure 6.2) to achieve the best ROA (see Figure 6.2). For example, given the proposed sensor management strategy, it is desirable to keep the FLIR FAR ($FAR_1$ in Figure 6.2) lower
because the system ROA is higher in this state of operation. However, doing this will lower $P_{d1}$, thereby imposing that $P_{d2}$ is set higher to achieve the same system $P_d$, because for the entire system $P_d = P_{d1}P_{d2}$. This increase in $P_{d2}$ will then increase the GPR false alarm probability ($P_f$ in Figure 6.2), and force the system to stop more frequently, thus lowering ROA. In order to operate the proposed system effectively, it is necessary to understand (i) how to compute ROA based on a particular choice of $P_{d1}$ and $P_{d2}$ (and then choose the best combination for a desired $P_d$) and (ii) to use this capability to measure the tradeoffs between system $P_d$ and ROA. Achieving these two goals then allows a system designer to choose a $P_d$ and ROA intelligently depending on the relative importance of the two metrics, and to ensure that the very best possible $ROA$ has been achieved given a choice of $P_d$ (by optimizing over $P_{d1}$, and $P_{d2}$), or visa versa.

In this work a probabilistic model is proposed to analyze the proposed BTD sensor management strategy described above in Chapter 6. The system is modeled as a queue (a waiting line) and then results from queuing theory are used to achieve the analytical goals stated above: primarily to compute the tradeoffs between ROA and system $P_d$, and to optimize over choices of $P_{d1}$ and $P_{d2}$. Although the model is applied to a multi-sensor FLIR-GPR system, in principle the approach is applicable for a wide variety of multi-sensor systems.
Figure 6.2. This figure illustrates the performance dynamics of the proposed sensor management strategy, in this case operating with a forward looking infrared (FLIR) camera and a downward-looking ground penetrating radar (GPR) system. FLIR and GPR data are fed to the system (left) and each detection system is operated at a certain operating point on their respective operating curves. The FLIR system operates at probability and detection and false alarm of $P_{d1}$ and $P_{f1}$ respectively (shown in red). The alarms from the FLIR are fed to the GPR (operating point shown in blue). The overall system metrics are shown on the far right. Overall probability of detection, $P_d$, and probability of false alarm, $P_f$, are computed on the right. The system false alarm rate is computed by multiplying the $P_f$ by the FLIR alarm rate, $R_{FA}$, which is the number of alarms detected per unit of distance driven. The other metric of interest here is the rate of advance, ROA, but it is unclear how to compute it because system velocity changes over time.

### 6.1 A queuing model for a 2-sensor detection system

This section presents a derivation of a queuing model (and corresponding CTMC model) for a 2-sensor BTD system. The derivation is applicable to BTD systems like the one described in the introduction of this chapter (Chapter 6), or in the introduction of this document (Chapter 1). The relationship of this BTD model to a queue is described first, and then in Section 6.1.1, a derivation is provided for the corresponding CTMC. Mathematical symbols that are not defined in the text below are consistent with
A 2-sensor BTD system, like the one considered here, can be thought of as a queue where alarms act as jobs, which arrive to the station, which is represented by the GPR (i.e., or other short standoff system) for processing. The alarms arrive from the FLIR (or some large standoff BTD system) as they are detected, and then each alarm is processed by the GPR. Once GPR processing is complete for a particular prescreener alarm, there is some probability it will be discarded by the GPR algorithm. There is also some probability that the alarm will cause the BTD system operator to stop the vehicle. These two outcomes and their probabilities correspond to the parameters $\alpha$, and $\bar{\alpha}$, in the left-most graphic in Figure 2.10.

Consider the latter scenario, when an alarm causes the BTD system to stop. In real demining scenarios, the BTD system operator stops the advance of the system when a location on the ground is designated as an alarm by both the FLIR and the GPR systems. In such a case the operator manually inspects the sensor data, or excavates the earth at the suspicious location. In the BTD queuing model proposed here, the value $T_{\text{stop}}$ will refer to the average time that the BTD system advance is stopped: this includes inspection of the data and excavations. Further, it will be assumed that the stopping times are independently and exponentially distributed. This is an approximation to reality which is made to simplify the queuing model, and also because there are no
statistics currently available for stopping times upon which to create a more sophisticated model. The exponential assumption can be relaxed in the future if stopping time statistics become available.

Now consider the $\alpha$, and $\bar{\alpha}$, parameters in the generic queuing model in Figure 2.10. These values correspond to the probabilities of the queue halting operation, or not. This is analogous to the probabilities of the BTD system stopping operation upon encountering a high confidence alarm, or not. Stopping the system due to an alarm (as opposed to simply discarding the alarm) occurs when the confidence of an alarm being processed by the GPR BTD has a confidence that exceeds a (designer-chosen) threshold, which itself corresponds to an operating point on the GPR BTD ROC curve. Denote the confidence of an alarm and the threshold of the GPR BTD system as $t_a$, and $t_0$, respectively. Let $H1$ denote the condition that the alarm under consideration is a target, and let $H0$ denote the condition that the alarm under consideration is a false alarm. Then the probability stopping the BTD system is given by

$$Pr(stopping) = Pr(t_a > t_0)$$

$$= Pr(t_a \geq t_0|H1) Pr(H1) + Pr(t_a \geq t_0|H0) Pr(H0)$$

$$= P_{d2} Pr(H1) + P_{f2} Pr(H0).$$

In many demining scenarios there are far more false alarms encountered by the system than true targets. This means that $Pr(H0) \gg Pr(H1)$, which implies that $Pr(H0) \cong 1$ because the two probabilities must sum to one. From these observations, an approximation to the probability of stopping is used in this work, and given by
\[ \Pr(\text{stopping}) \equiv P_{f2}. \]  

The queuing model resulting from the aforementioned assumptions is illustrated in the left-most graphic in Figure 5.3. The queue in Figure 5.3 is analogous to the generic queue presented in Figure 2.10, except the queue parameters are now replaced with values corresponding to a real BTD system.

One additional modeling assumption is also made about the behavior of the system operator during conditions when the system is stopped for an alarm. It is assumed that, if the BTD system is stopped for an alarm, then the operator automatically inspects any remaining alarms in the queue (i.e., alarms that have not finished processing at the GPR BTD system). This effectively allows the system to discard any alarms in the queue whenever the system is stopped. This assumption will become important when the CTMC model for the BTD queue is discussed in the next section. Similar to other assumptions made thus far, it greatly simplifies the model, and the assumption can be relaxed in the future if needed. This concludes the description of the correspondence between the FLIR-GPR BTD system and a queue. The BTD system queuing model is illustrated below in the left-most illustration in Figure 5.3.

### 6.1.1 Derivation of the state transition rates

In order to find a corresponding CTMC for the BTD system queue, it must be shown, or assumed, that the arrival and departures of alarms from the GPR detector form Poisson processes. Further, the rate parameter for each of the processes must be
derived because they represent the transition rates between the states of the CTMC (as described in Section 2.6). The CTMC is fully characterized once expressions for these parameters are determined, and the desired system characteristics (e.g., ROA) can be computed.

The arrival rate can be derived by first assuming that the FLIR alarms do indeed form a Poisson process. This assumption is shown to be reasonable based on experimentation using real BTD sensor and alarm data in Section 6.2.4. Given that the alarms do form a Poisson process, it is necessary to derive an expression for the rate parameter in terms of real system parameters. Let the rate parameter of the process be denoted by \( \lambda \). Then the probability that \( k \) alarms are encountered in time, \( T \) is given by the following probability mass function:

\[
f_k(k; \lambda, T) = \frac{e^{-\lambda T} (\lambda T)^k}{k!}
\]  

(37)

Note that the expectation of \( k \) is the average number of alarms received in \( T \) seconds. Now, let \( FAR_1 \) be the false alarm rate (in alarms per meter driven) of the FLIR system, given by

\[
FAR_1 = P_{f1} R_{FA},
\]  

(38)

where \( P_{f1} \) is the probability of false alarm of the FLIR detector at a given operating point (see Figure 6.2 for an illustration), and \( R_{FA} \) is the rate of alarms the prescreener generates per unit of distance going down a path (usually given in alarms per meter). Note that \( FARs \) are routinely estimated and reported for detection
algorithms in the BTD literature [5], [30], [67], and therefore are typically readily available. Now, let $D$ be the distance travelled in $T$ seconds, and $v$ be the system velocity. Then we have that

$$E[k] = \lambda T = FAR_1 D = FAR_1 (vT) = (FAR_1 v)T$$

$$\rightarrow \lambda = FAR_1 v,$$  \hspace{1cm} (39)

where $\lambda$ is the rate parameter for alarm arrivals for the current state, which can be called $i$, and is therefore equivalent to the transition rate for that state, $q_{i(i+1)}$. If we allow the velocity to vary by state we have that

$$q_{i(i+1)} = FAR_1 \cdot v_i.$$  \hspace{1cm} (40)

This states that the alarm arrival rate depends on the $FAR_1$ (e.g., alarms per meter driven) of the FLIR detector and the velocity of the system. $FAR_1$ can be estimated from real data and can be chosen, either directly or by setting $P_{d1}$. The service rate is simpler to derive. It is the rate at which the queue processes jobs. Let the time required for the GPR detection algorithm to process a job be independent and exponentially distributed with average time, $T_p = 1/\mu_p$. This assumption is routinely applied to computer programs in queuing theory applications and is therefore used here [47]. This assumption implies that the processing times form a Poisson process and therefore the corresponding transition rates are given by
\[
\mu_i = \frac{1}{T_p} = \mu_p.
\] (41)

Here the rate is constant for all states because it is assumed that the processing time does not depend on the state. The CTMC model corresponding to this queue is shown in the right-most panel below in Figure 6.3. This completes all the transition rate specifications, and the model can now be used to derive state probabilities, \( \pi \), in terms of real system parameters, using equation (11) in Section 2.6.

### 6.1.2 Computation of ROA

Now that the CTMC model parameters are all specified, it is straightforward to apply the Markov reward model (see (16) in Section 2.6.1). Using this model it is possible to compute the average system ROA by assigning the rewards for each state as follows,

\[
r_j = v_j
\]

\[
\rightarrow E_X[v] = \text{expected ROA}
\]

This represents a method of computing the system ROA. With these equations it is possible to make predictions about the ROA, which is one of the major goals of this work. Further, this establishes a relationship between the ROA and \( P_d \). This is because the steady-state probabilities given by equation (14) in Section 2.6 depend directly on \( FAR_1 \) and \( P_{f2} \), and these values corresponds to particular choices of \( P_d = P_{d1}P_{d2} \). Therefore, it is possible to quickly compute the tradeoffs between ROA and \( P_d \), and/or to optimize over \( P_{d1} \) and \( P_{d2} \).
Figure 6.3. The left-most illustration shows the FLIR-GPR system with several important processing quantities labeled. The right-most illustration depicts the corresponding birth-death model for the FLIR-GPR system, along with the arrival and processing parameters that were derived in Section 6.1.1. In this model, alarms from the FLIR detector represent jobs arriving to the queue, which is the GPR feature processing stage. The rate at which alarms arrive to the GPR from the FLIR detector depends on the false alarm rate of the FLIR ($FAR_1$, in false alarms per meter driven) and the velocity at which the system is moving down track in each state, $v_i$. The job processing rate is based on how quickly the GPR algorithms require to process each alarm, $T_{GPR}$.

### 6.2 Validating the sensor management strategy and queuing model

This section presents the results of two experiments: one experiment to prove the applicability and/or accuracy of the proposed model on a BTD system, and one to show that the proposed sensor management strategy achieves the goal of leveraging the relative benefits of the FLIR and GPR sensors in the system. The first experiment examines whether BTD data meet the assumptions of the proposed model, and whether its predictions match those obtained with simulations using real field-collected BTD system data. The second experiment demonstrates that a FLIR-GPR system operating with the proposed sensor management strategy achieves greater ROA than a stand-
alone GPR system, but higher performance than a stand-alone FLIR system. The next three sections present an overview of the experimental data, the algorithms, and the model parameter settings respectively. The two sections that follow present the results of the two experiments.

### 6.2.1 Dataset description

The experiments in this work utilize two sets of data: one GPR and one FLIR dataset. The FLIR data was collected using a vehicle-mounted forward looking infrared camera with 640x480 pixel resolution and 16-bit pixel intensity resolution. The training and testing data were collected using several runs over two different lanes at an arid western U.S. location. One lane was used as a training dataset for inferring parameters of the FLIR detector. The second lane was used as a testing dataset. This data has been used in many previous FLIR BTD investigations [60], [66].

All GPR data was collected from a NIITEK 51-channel array, with 6 cm and 5 cm crosstrack and down-track sampling respectively. The GPR training data was collected over several runs on a several lanes at an arid western U.S. test site on which alarm locations that were marked in by the F1V4 prescreener [64], [65]. The resulting training data contained several thousand target and non-target alarms. Edge histogram descriptor (EHD) features were extracted at each alarm and a support vector machine (SVM) was trained on the features. The testing GPR data was collected over the same
lane as the FLIR testing data, and EHD features were extracted at alarm locations indicated by the FLIR detector.

### 6.2.2 Detection algorithms

The FLIR detection algorithm used in this work was developed in two recent FLIR landmine detection studies [19], [39] and subsequently used in numerous others studies (e.g., [23], [34]). The FLIR detector operates by filtering each FLIR image with an ensemble of size-contrast filters (anomaly detectors) and then performing several clustering steps. The final output of the algorithm is a list of alarms in UTM coordinates along with a decision statistic indicating the likelihood that a target is at that location. All algorithm parameters were learned on a separate training dataset. The GPR detector consists of first extracting Edge histogram descriptor (EHD) features [30] at all the alarm locations marked by the FLIR prescreener described above. Once the features are extracted from each alarm location (a GPR image patch), an SVM with a radial basis function kernel is used [42] to produce a decision statistic for that patch. The SVM was trained on a separate training dataset.

### 6.2.3 Detection system experimental settings

There were a number of system parameters specified in the model development that correspond to real-world quantities that depend on the FLIR-GPR system design. In order to run simulations and/or make model predictions, values must be specified for these BTD system parameters. Table 3 below summarizes these parameter choices for
the experiments that are presented next. The values were chosen to be similar to values that might be encountered in practice. Several new parameters are also introduced. First, $V_{GPR}$ and $V_{FLIR}$ that correspond to the velocities of the system when the GPR is active and the GPR inactive, respectively. This implies that the state velocities are set $v_i = V_{GPR} \forall i \geq 1$, where $i$ indicates the state. Then $v_0 = V_{FLIR}$. The other parameter is $D_{GPR}$ which is the assumed standoff of the GPR system.

Table 3 A table of the values of physical system parameters that are used in the experiments presented in this chapter. Values were chosen to be representative of what might reasonably be encountered in practice.

<table>
<thead>
<tr>
<th>System parameter</th>
<th>Experiment 1</th>
<th>Experiment 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_d$</td>
<td>[0.9, 0.3]</td>
<td>[0.9, 0.3]</td>
</tr>
<tr>
<td>$P_{d_1}$</td>
<td>[0.9, 0.3]</td>
<td>[0.9, 0.3]</td>
</tr>
<tr>
<td>$P_{d_2}$</td>
<td>$= \frac{P_d}{P_{d_1}}$</td>
<td>$= \frac{P_d}{P_{d_1}}$</td>
</tr>
<tr>
<td>$V_{FLIR}$</td>
<td>[2.8,5.6] m/s, [10-20] KPH</td>
<td>4.2 m/s (15 KPH)</td>
</tr>
<tr>
<td>$V_{GPR}$</td>
<td>[0.5, 2.5] m/s, [2.5-7.5] KPH</td>
<td>1.4 m/s (5 KPH)</td>
</tr>
<tr>
<td>$D_{GPR}$</td>
<td>0.75 m</td>
<td>0.75 m</td>
</tr>
<tr>
<td>$T_p$</td>
<td>$= \frac{D_{GPR}}{V_{GPR}}$ seconds</td>
<td>$D_{GPR}/V_{GPR} = 0.54$ seconds</td>
</tr>
<tr>
<td>$T_s$</td>
<td>[5,15] seconds</td>
<td>10 seconds</td>
</tr>
</tbody>
</table>

6.2.4 Experiment 1: validity of the queuing model

This section presents experiments that examine the effectiveness of the proposed queuing model at representing the proposed BTD system. The first experiment examines the validity of the assumption that alarm arrivals form a Poisson process. If this is true, then it can be shown [47] that the alarm inter-arrival times are exponentially distributed, and mutually independent. This assumption is examined on the alarms from the FLIR dataset described in Section 6.2.1. For this experiment, the inter-arrival times of alarms were computed assuming a FLIR velocity of $V_{FLIR} = 4.2 m/s$. A
histogram was created of the inter-arrival times of the alarms (by second) and an exponential distribution was fit (via the maximum likelihood method) to the data. These results are shown below in left-most panel of Figure 6.4. The maximum-likelihood inter-arrival frequencies are plotted in red over the observed values. If we denote the inter-arrival times by $A_t$, and let $A_{t-1}$ denote the inter-arrival time of all alarms immediately preceding the alarm times given by $A_t$. A 2-D histogram was constructed for these two values to examine their independence. These results are shown in the right-most panel in Figure 6.4. Observations greater than 3 standard deviations from the mean have been removed. The results show that an exponential distribution is, visually, a reasonable approximation for the alarm inter-arrival times. The maximum likelihood fit well approximates the true values. The 2-D histogram of inter-arrival times indicates that a relatively small (linear) correlation exists (Pearson correlation coefficient, $\rho = 0.245$ with p-value=0.000). This is a modest departure from independence that potentially explains some of the mismatch in the histogram.
Figure 6.4. In the left-most panel is a histogram of the FLIR alarm inter-arrival times, assuming a velocity of 4.2m/s. An assumption of the proposed BTD queuing model is that the distribution of the alarm inter-arrival times is exponential. A qualitative assessment of this assumption is made by fitting (via maximum likelihood estimate) an exponential distribution to the histogram (red). This shows that, while the exponential assumption is not perfect, it appears to be reasonable. Another assumption of the birth-death queuing model is that any two alarm inter-arrival times are independent of one another. On the right-most panel is a 2-D histogram showing the joint distribution between alarm inter-arrival given by $A_t$, and the inter-arrival times immediately preceding $A_t$, given by $A_{t-1}$. This figure shows that any two consecutive inter-arrival times (i.e., $A_t$ and $A_{t-1}$) are correlated, however this correlation is not very strong. This is indicated by the Pearson correlation coefficient shown in the title, along with its p-value.
Figure 6.5. These figures provide scatter plots of predictions made by the FLIR-GPR queuing model and the corresponding values estimated by simulation of the FLIR-GPR system using real data. The left-most plot is the most important. It illustrates the agreement between the model predictions of the system ROA and those obtained from simulations. This is quantified by the Pearson correlation coefficient and corresponding p-value in the title. The four smaller plots on the right illustrate some disagreement of state probability predictions made by the model, and those from simulation. As the state index increases, the queuing model consistently underestimates the amount of time (by proportion) that the system spends in that state. This is likely due to the linear correlation between alarm inter-arrival times. Alarms tend to come in groups and cause the queuing model to more frequently occupy larger-index states than would be predicted by the queuing model because it assumes independent inter-arrival times.

The second experiment presented here investigates the ability of the FLIR-GPR queuing model to make accurate system performance predictions. This is accomplished by running a simulation of the FLIR-GPR system using the real field-collected FLIR and GPR data described in Section 6.2.1. Based on the simulated vehicle movement, the ROA and state probabilities of the FLIR-GPR system are computed and compared with
those predicted by the queuing model. This procedure was performed 1000 times using
uniformly randomly distributed system parameters from value ranges given in Table 3.
The results of this simulation process are shown below in Figure 6.5. The results on the
left are the most important, showing the ROA computed from simulations vs. those predicted using the model. As can be seen, the correspondence is nearly perfect (Pearson correlation of $\rho = 0.998$ with $p$-value=$0.000$). Similar to ROA, the state probability estimates are also shown on the left for the first 4 states. In these plots there is a clear negative bias in the model predictions as compared to the simulated estimates, specifically for higher states. This is likely a consequence of the correlation between inter-arrival times that was described in the previous section. These correlations mean alarms tend to be encountered in clusters, causing the queue to move towards higher states than might be caused if the alarms were independently distributed. The model assumes no correlations and therefore tends to underestimate the occupancy of the lower states. These alarm arrivals can be handled with a more sophisticated queuing model, such as Markov Modulated Poisson Processes [47]. This is left for future work.

6.2.5 Experiment 2: validity of sensor management strategy

This section investigates the ability of the proposed multi-sensor BTD system to leverage the relative benefits of its constituent sensors: in this case the FLIR and GPR. In order for the system to achieve this goal it is important that the system parameters are set to effective values. This is specifically true for the choice of $P_{d1}$ and $P_{d2}$, because
these values determine how much the system target detection rate, \( P_d \), relies on the GPR and FLIR respectively, and similarly (in a complicated way) how much time is spent with the GPR active (in which case the system velocity is low). The impact of this choice is shown in Figure 6.6 in the left-most panel. The ROA of the system is predicted (by the queuing model) as a function of \( P_{d1} \) and \( P_{d2} \) (and \( P_d \) fixed at 0.65). The results show that the ROA can vary dramatically depending on how these system parameters are chosen. Further, it shows that, at least in this situation, relying on either the GPR or the FLIR completely for detection is a poor choice. The red circle indicates the best overall choice and the blue circle indicates the ROA if detection relies completely on the FLIR. There is no circle for the GPR only (i.e., if \( P_{d2} = 0.65 \)) because this would require that \( P_{d1} = 1.0 \), and the FLIR does not find all the targets, so this is not possible. Nonetheless, it is also not effective overall to rely too heavily on the GPR BTD system either: the best choice is a balance, which can be computed precisely using the model and used in real operation.

The overall performance comparison of the FLIR-GPR system (red line) is shown in the right-most panel in Figure 6.6, and it is compared with the performance of a stand-alone FLIR system (blue line), and a stand-alone GPR system (black line). For each choice of overall system detection sensitivity (x-axis), the queuing model was used to find the combination of \( P_{d1} \) and \( P_{d2} \) for the FLIR-GPR system that achieves the greatest ROA. The results indicate that, for any choice of \( P_d \), the FLIR-GPR system
always achieves a greater ROA than the FLIR system. This is an expected result because the FLIR-GPR system can always perform exactly the same as the FLIR by relying completely on the FLIR detector for detection (by always setting \( P_{d2} = 1 \)), and therefore its ROA is lower bounded by the FLIR system. The FLIR-GPR system can sometimes obtain better ROA, as compared to the FLIR system by using the GPR when it is beneficial. The GPR can be employed by setting \( P_{d2} < 1 \). This has the tendency to place more detection responsibility on the GPR, and therefore the system spends more time with the GPR operating. This slows down the system but it can also lower the amount of false alarms that result in stops, and therefore it improves the system ROA overall. So therefore, the FLIR-GPR system allows a designer to tradeoff velocity from the FLIR with lower false alarm rates with the GPR to achieve an overall ROA that is better than either system could obtain individually. Therefore, the FLIR-GPR system has the advantage (over the FLIR) of relying upon the GPR only when it is beneficial for the overall FLIR-GPR system ROA, and therefore the FLIR-GPR system always obtains an ROA that is greater than, or equal to, the standalone FLIR detector.

The FLIR-GPR system also achieves better ROA than the stand-alone GPR system for most ranges of detection sensitivity, but not all. It cannot always achieve a greater ROA than the GPR because the FLIR sensor fundamentally limits the performance of the FLIR-GPR system. The GPR can only operate on alarms that come from the FLIR, whereas in a stand-alone GPR detector, all of the data is operated on
using the GPR detection system. The GPR detection performance (e.g., higher $P_d$, given a fixed FAR) is substantially better than that of the FLIR, and therefore it has very few (almost no) false alarms, which is why the GPR curve is nearly flat. However, the GPR system is fundamentally limited to travel at 1.4 m/s and therefore can never have a greater ROA than this, even if it rarely stops for false alarms.

In practice, detection sensitivities need to be very high and, while the FLIR-GPR system often offers the best overall performance (i.e., the best combination of ROA & detection sensitivity), the stand-alone GPR system offers the best performance at high sensitivities. This is primarily because the FLIR detection system (specifically the algorithm) does not perform well enough to yield benefits at high detection sensitivities. However, in principle, if the FLIR detection system (or any large standoff detection system) performs well enough, then the proposed system will offer better performance than the stand-alone GPR system, even at high detection sensitivities, and that is the purpose of the experiments presented here. As the large standoff system detection performance (as measured by ROC analysis) becomes better, both the red and blue curves will shift to the right and the region over which the FLIR-GPR system yields improvements will increase.
Figure 6.6. The left-most panel of this figure illustrates the impact of $P_{d1}$ and $P_{d2}$ on the overall system ROA, for the specific case when the system detection sensitivity, $P_d = 0.65$. A much better ROA (red circle) can be obtained by properly choosing $P_{d1} \& P_{d2}$. The right-most panel shows the relationship between overall system detection sensitivity and ROA for three different systems: the FLIR-GPR system (red curve), a stand-alone FLIR system (blue curve), and a stand-alone GPR system (black curve). The results show that the FLIR-GPR system offers better ROA than the stand-alone FLIR for all detection sensitivities, but that the GPR offers better ROA at high sensitivities. This is because the FLIR-GPR system is fundamentally limited by the performance of its FLIR detector, but in principle, the FLIR-GPR can achieve better ROA even at high sensitivities if the FLIR detector (or any other large standoff detector) used in the system achieves better detection performance.

The next several figures illustrate more details about the relationships between various system parameters and the behavior of the BTD systems. Note that the standalone GPR system is not included in these figures because, in contrast to the other two systems, the ROA of the standalone GPR system is nearly constant across all combinations of $P_d$ and FAR.

Figure 6.7 expands on Figure 6.6 by showing ROA and system $P_{d_1}$ as well as the system false alarm rate (FAR), for the FLIR system and the SMS system (i.e., the FLIR-
GPR system). This figure shows that, as the system $P_d$ is increased, the system FAR is increased, and this results in a lower system ROA due to increased slowdowns (to activate the GPR) and vehicle stops.

Figure 6.7. This figure expands on Figure 6.6, and shows the relationship between ROA and System $P_d$, as well as the system false alarm rate (given in false alarms per meter driven) for the FLIR system and the SMS System (i.e., FLIR-GPR). As the system $P_d$ increases the false alarm rate increases, and consequentially lowers the system ROA by increasing the amount of time spent driving slowly, or stopped.

Figure 6.8 illustrates the relationship between the false alarm rate and the probability of detection of the SMS system and the standalone FLIR system. Further, each point on each ROC curve corresponds to a specific system ROA (in meters per second), which is written in black next to each plot marker. Recall that the SMS system
was optimized to achieve the best ROA for a given $P_d$. Therefore the SMS system (for these testing conditions) always achieves a greater ROA than the standalone FLIR system, given a fixed choice of system $P_d$.

**Figure 6.8.** This figure shows the relationship between the false alarm rate and the probability of detection of the SMS system (i.e., FLIR-GPR) and the standalone FLIR system. Each point, on each ROC curve, corresponds to a specific system ROA (in meters per second), which is written in black. Recall that the SMS system was optimized to achieve the best ROA for a given $P_d$. Therefore the SMS system (for these experimental conditions) always achieves a greater ROA than the standalone FLIR system for any choice of system $P_d$.

Figure 6.9 illustrates the dependence of the system ROA on the operating point chosen for the FLIR and GPR, respectively. The leftmost panel shows the relationship
between the FLIR $P_d$ (i.e., $P_{d1}$) and the GPR $P_d$ (i.e., $P_{d2}$). Each contiguous blue/green contour has a constant system $P_d$. The red line shows the values of $P_{d1}$ and $P_{d2}$ that optimize the system ROA for each choice of system $P_d$. The best balance between $P_{d1}$ and $P_{d2}$ changes depending on the desired overall system $P_d$. The right-most panel shows the system ROA as a function of the GPR false alarm rate (FAR), and the FLIR FAR. Note that any choice of $P_{d1}$ and $P_{d2}$ (as in the left-most panel) corresponds to a choice of GPR FAR and FLIR FAR, respectively. This plot shows specifically how the system performs as ROA increases. Increases in FLIR FAR correspond to more frequent system slowdowns, while increases in GPR FAR correspond to more frequent stops.

Figure 6.9. The leftmost panel shows the relationship between the FLIR $P_d$ (i.e., $P_{d1}$) and the GPR $P_d$ (i.e., $P_{d2}$). Each contiguous blue/green contour has a constant system $P_d$. The red line shows the values of $P_{d1}$ and $P_{d2}$ that optimize the system ROA for each choice of system $P_d$. The best balance between $P_{d1}$ and $P_{d2}$ changes depending on the desired overall system $P_d$. The right-most panel shows the system ROA as a function of the GPR false alarm rate (FAR), and the FLIR FAR. Note that any choice of $P_{d1}$ and $P_{d2}$ (as in the left-most panel) corresponds to a choice of GPR FAR and FLIR FAR, respectively. This figure shows specifically how the system performs as ROA increases. Increases in FLIR FAR correspond to more frequent system slowdowns, while increases in GPR FAR correspond to more frequent stops.
6.3 Conclusions

In this chapter, a new multi-sensor management strategy was proposed for vehicle-mounted BTD systems. The proposed system combines a high detection performance (e.g., high $P_d$, given a fixed FAR), but low ROA, GPR system with a FLIR system with complementary characteristics: a high ROA and lower detection performance. The potential benefit of the proposed BTD system is that it can achieve combinations of $P_d$, FAR, and ROA that are not achievable by either of the two constituent sensors individually. However, despite this potential advantage, the proposed system ROA and PD/FAR is difficult to analyze because realistic con-ops require changing system velocities and sensor activities. This makes the design of such a system difficult because the designer must choose an operating detection sensitivity (i.e., operating point on the detector ROC curve) and it is unclear how this will impact the system ROA. To solve this problem, a new probabilistic model is also derived for the proposed BTD system based on modeling the system with a queue and using queuing theory to derive a relationship between the systems detection sensitivity and its ROA. It also facilitates optimization of some other system parameters as well.

Results for two experiments were presented in this work. The first experiment used the derived queuing model to make predictions about the BTD system behavior and these predictions were compared to predictions derived from simulations of the real system using real field collected BTD system data. The second experiment measured the
ROA and detection sensitivity of a FLIR-GPR system against the performance of a stand-alone FLIR system and a stand-alone GPR system. Based on the experimental results, the following conclusions can be drawn:

- The derived queuing model for the proposed multi-sensor BTD system is capable of accurately predicting the ROA, and other parameters, for the example FLIR-GPR system utilized in this work.

- The proposed multi-sensor BTD system is often capable of yielding better overall performance, in terms of ROA, $P_d$, and FAR than its constituent sensors, as illustrated on the test FLIR-GPR system.

Despite these encouraging results, there are still many challenges with the proposed system. The queuing model presented here represents a somewhat simplified version of a real system and more sophisticated queuing models can be employed to more accurately capture true BTD system dynamics, or more than two sensors on a single platform. Additionally, the model was shown to be effective for a single pair of detection systems (a FLIR and GPR system, respectively). The proposed model should be validated on larger and more diverse sets of data to further validate its effectiveness. The next chapter draws some broad conclusions about the pursuit of greater standoff/ROA and my work presented in this document.
7. Conclusions and future work

The main objective of the work presented throughout this dissertation was to develop novel statistical and algorithmic methods to improve the ROA of BTD systems. More specifically, the goal was to develop a detection system that obtains detection performance (e.g., $P_d$, given a fixed FAR) that is comparable to downward-looking GPR-based systems but with greater ROA. GPR was the primary focus because it provides state-of-the-art detection performance among BTD modalities, but it suffers from relatively low ROA due to its very short detection standoff distance.

Large standoff sensors can typically achieve much greater ROA and many such sensors have been investigated for BTD, but they do not currently offer detection performance (e.g., $P_d$, given a fixed FAR) comparable to GPR. Therefore a primary strategy in the first half of my dissertation work was to develop detection algorithms that improve the detection performance of large-standoff modalities. In Chapter 3 and Chapter 4, new detection algorithms were developed for two large-standoff remote sensing technologies: for seismo-acoustic sensors, and FLIR cameras, respectively. Real field-collected sensor data was used for each sensor to develop and evaluate the efficacy of the proposed new algorithms. Results showed detection performance improvements for each modality [58], [68].

Although algorithmic improvements brought seismo-acoustic and FLIR detection closer to the performance of downward looking GPR sensor performance,
there still existed a large detection performance disparity. In other words, the large standoff modalities still exhibited $P_a$s that are too low given the operating FARs examined. This difficulty led to an alternative proposed strategy whereby GPR was combined with a large standoff FLIR on a single detection system to yield an overall system with greater standoff (and therefore ROA) than a stand-alone GPR system. The FLIR detector was used as a prescreener to mark locations on the ground where GPR data was then collected for feature processing. Edge histogram descriptor (EHD) features were extracted from the GPR data and then classified with an SVM. Using this framework, it was possible to increase the detection standoff distance by reducing the amount of GPR data collected around each FLIR alarm location. Experiments were conducted to investigate the tradeoff between added standoff distance and detection performance (as measured by partial area under the ROC curve) that is achievable using this system. The experimental results showed that the proposed FLIR-GPR system can increase standoff distance substantially without much loss in detection performance.

Although effective, the proposed fused FLIR-GPR system is fundamentally limited because both the FLIR and the GPR need to operate at all times. This limits the overall system ROA to be no greater than that of the sensor with the smallest ROA (or shortest standoff): in this case the GPR. This limitation motivated the investigation of a second proposed approach where a similar FLIR-GPR system was employed but with a sensor management strategy that allows the GPR to deactivate over periods of time in
which it is not needed. In the new approach the GPR is only activated under certain conditions to improve detection performance (e.g., increase $P_d$, given a fixed FAR), and otherwise remains inactive to increase system standoff and ROA. Although potentially effective, this new sensor management strategy also introduced new analysis challenges. Most importantly, it is more difficult to measure ROA because the vehicle velocity changes over time. To solve this problem, a new probabilistic model, based around the theory of queues (waiting lines), was developed to analyze BTD systems operating with the proposed sensor management strategy. Experiments were conducted with real field collected data and showed that the probabilistic model could accurately model the FLIR-GPR system and its new sensor management strategy, and that the proposed management strategy was effective for improving ROA.

### 7.1 Summary of contributions and future work

This document presented several scientific contributions in signal processing and machine learning motivated by the goal of improving the ROA of BTD systems. The main contributions of the work are summarized as follows:

- The development of new algorithms for improving target detection, as measured by ROC analysis, in seismo-acoustic and FLIR BTD systems, respectively.
• The development and testing of a two-sensor BTD system that facilitates a tradeoff between detection performance (e.g., $P_d$, given a fixed FAR) and standoff distance (and thereby system ROA).

• The development of a new sensor management strategy for two-sensor BTD systems that permits time-varying sensor activity and system velocity.
  
  o In particular it was shown that when applied to a FLIR-GPR system the proposed strategy yields a system with better combinations of ROA, $P_d$, and FAR than either of the two systems when operated alone.
  
  o A probabilistic model for analyzing two-sensor BTD systems operating under the proposed sensor management system. The model facilitates estimation of system ROA, and tradeoffs between ROA and system detection performance.

In particular, the new sensor management strategy proposed in Chapter 6 represents a promising general method for combining short standoff and large standoff sensors in a potentially very synergistic manner. It was shown that the combination of a GPR and a FLIR operating with the proposed sensor management strategy could achieve greater ROA, $P_d$, and FAR than either system alone. The main limitation however, was that the range of target detection rates (measured by $P_d$ in Figure 6.2) over which this synergistic
behavior was achieved was at impractically low levels of $P_d$. GPR systems, for example, often report probabilities of detection in the ranges of 80-95% [5], [17], [69]. It is plausible however, that if the FLIR detection performance (FAR, given a fixed $P_d$ around 90%), or the detection performance of some other large standoff detector, were improved enough then this synergistic region of operation would be more practical. Then, such an operating strategy would offer an excellent approach for combining multiple BTD systems to exploit their respective benefits. Therefore, an important area of future work is to examine the proposed system with more increasingly advanced large standoff BTD systems that may already achieve the desired detection performance. Alternatively, more work can be done to improve signal processing, and thereby detection performance, for existing large standoff BTD systems.
Bibliography


Biography

Jordan Malof was born in Fort Thomas, Kentucky on June 1\textsuperscript{st} 1986, and grew up in Florence, Kentucky. He attended high school at Boone County High School and then University of Louisville where he obtained a Bachelor of Science in Electrical Engineering. Subsequently he attended Duke University in Durham, NC where he obtained a Master of Science degree in Electrical Engineering, with a focus on signal processing, statistical modeling, and machine learning.