Project Report
MD-51

Sensor Placement Analysis for Defense Against Uncertain Raids of Ballistic Threats

Z.T. Chance
S.R. Vogl
L.D. Layne

6 March 2018

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS

This material is based upon work supported by the Missile Defense Agency under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001.

This report is the result of studies performed at Lincoln Laboratory, a federally funded research and development center operated by Massachusetts Institute of Technology. This material is based upon work supported by the Missile Defense Agency under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Missile Defense Agency.

Massachusetts Institute of Technology
Lincoln Laboratory

Sensor Placement Analysis for Defense Against Uncertain Raids of Ballistic Threats

Z.T. Chance
L.D. Layne
Group 36

S.R. Vogl
Group 38

Project Report MD-51

6 March 2018

Lexington Massachusetts

This page intentionally left blank.

ABSTRACT

Sensor management lies at the crux of engagement support, and effective deployment of sensors is an important component of a battle management system. Sensors need to be properly placed to provide sufficient coverage for detection, tracking, and threat identification capabilities. This becomes particularly challenging in scenarios where the available sensor resources are strained by a large number of concurrent threats. However, many such difficult scenarios can be dealt with if sensors can cooperate and information can be effectively combined. Thus, selection and placement of sensor resources for complex scenarios should be done at a system level (i.e., where information from all sensors is available) to account for potential performance gains enabled by the fusion of data from multiple sensors. To this end, this work constructs a mathematical framework for the analysis of effective deployment of multiple sensors with the objective of supporting weapons systems. In addition, it is assumed that the defense is posed against a large number of concurrent threats characterized by uncertain a priori information.

In the development of a sensor placement strategy, there exist two main steps: analysis and optimization. Analysis is the construction of meaningful metrics of performance, i.e., objective functions, to quantify the desirability of a given sensor layout. Ideally, the metrics chosen should accurately reflect the relative success or failure to meet design objectives and also impart intuition on the sensitivity of the performance to deviations from what is expected. Optimization is the efficient search over sensor parameters to maximize/minimize the analysis objective functions, and, thus, cannot be performed until the analysis step has been executed. As the specific problem of optimizing sensor placement for defense against a raid of ballistic threats has not been addressed previously, an analysis framework needs to be developed before optimization takes place—that is the main contribution of this work. Specifically, an analysis methodology is built using two novel mechanisms:

- Sensor placement objective functions based on the quality system-level tracking,
- Probabilistic model for a raid of ballistic threats that allows the calculation of any chosen statistic of system performance.

Sensor placement grounded in system-level tracking enables straightforward evaluation of the capability to meet track-quality-based engagement constraints. Further, it encapsulates many joint sensor effects such as geometric diversity when employing track fusion. A probabilistic threat model creates the capability to inject confidence into the threat model, i.e., higher probability mass conveys higher confidence. In some cases, founding the threat model in probability can also allow for sensitivity analysis without the need to run Monte Carlo trials that may require significant computation time. Also, describing the enemy courses of action in a Bayesian sense facilitates the definition of uncertainty in threat aspects, i.e., high variance is equivalent to low certainty. Simulations are provided to show operating area analysis using a system-tracking-based objective function and probabilistic threat definition.

This page intentionally left blank.

TABLE OF CONTENTS

This page intentionally left blank.

LIST OF FIGURES

LIST OF FIGURES
(Continued)

LIST OF TABLES

This page intentionally left blank.

1. INTRODUCTION

The intelligent emplacement of sensor resources is a key component of engagement support, and the deployment of sensors is a critical function of a battle management system. To provide sufficient coverage for detection, tracking, and threat identification capabilities, sensors need to be properly placed. This becomes particularly challenging in scenarios where the available sensor resources are stressed by a large number of concurrent threats. However, many difficult scenarios can be addressed if sensors cooperate and information is effectively combined. Thus, preplanning the allocation of sensor resources for complex scenarios should be done at a *system level* (i.e., where information from all sensors is available) to exploit the potential performance gains enabled by the fusion of data from multiple sensors. This problem is important as it arises in many different mission areas with complex scenarios; for example, the preplanning of sensors is an important component used in many facets of air and missile defense.

In recent years, the mission objectives of air and missile defense have evolved to incorporate the need to handle large numbers of simultaneous threats. However, much of the currently-used preplanning theory and software was developed under the assumption that few threats will be present concurrently. Thus, there lies an inherent theoretical gap between existing planning capabilities and the requirements of air and missile defense. This gap presents the main issue at hand: a planning system is needed to effectively manage sensor resources in the presence of many concurrent threats. The goal of this report is to develop a framework to address this issue.

In this report, the examples and results focus on the missile defense mission, i.e., a strong emphasis on the deployment of sensor systems used by the Ballistic Missile Defense System (BMDS) including maritime sensors, airborne sensors, land-based sensors, radio frequency (RF) sensors, and electro-optical/infrared (EO/IR) sensors. Algorithms used for planning the emplacement of these sensors are developed as part of the Command, Control, Battle Management, and Communications (C2BMC) program and the Ground-based Midcourse Defense (GMD) program.

1.1 SENSOR PREPLANNING ANALYSIS

Due to computational constraints, the management of sensor resources is generally divided into different subproblems (see Figure 1). This division arises due to the inherent trade-off between the time horizon for analysis and how often control decisions are made. For example, if one was practically limited to the analysis of 100 control decisions, then this could be used either to investigate the performance of a system with control decisions every 1 second for 100 seconds or a system with control decisions every 0.01 seconds for a second. As is shown in Figure 1, for a fixed computational limit, the time horizon must vary inversely with control frequency. Thus, two separate sensor management regimes are introduced, preplanning and dynamic sensor management. In the preplanning component, slow-time variables, i.e., variables that require many hours to days to change, such as sensor location and orientation, are analyzed and subsequently optimized (typically by an operator) for expected performance. Due to the limited frequency of control in this stage, performance can be calculated over long time horizons or entire scenario timelines. In contrast, dynamic sensor management deals with real-time control of the sensing systems, e.g., radar beam

scheduling and optical sensor slewing, which can make changes on the order of seconds. Due to the number of control decisions possible, dynamic sensor management can only be analyzed and optimized for a limited time horizon. In this document, the focus will be on preplanning and the deployment of sensors for supporting weapons systems. Note that this objective falls under a class of problems known as *sensor placement* or *sensor deployment* problems.

Additionally, it is important to note the difference between performance analysis and optimization for sensor resource management. The key function of analysis is the development of meaningful measures of effectiveness to objectively quantify the performance of a system of sensors. Further, it should be able to impart intuition on the driving factors for success or failure of a particular sensor management doctrine. Optimization is the efficient search over possible sensor configurations to maximize (or minimize) the measures of effectiveness constructed in performance analysis. As there are many new aspects to the problem under consideration in this report, analysis must be approached first, and, accordingly, will be the chosen focus. Therefore, the major product of this work will be the construction of measures of effectiveness for the placement of sensors in support of weapons systems.

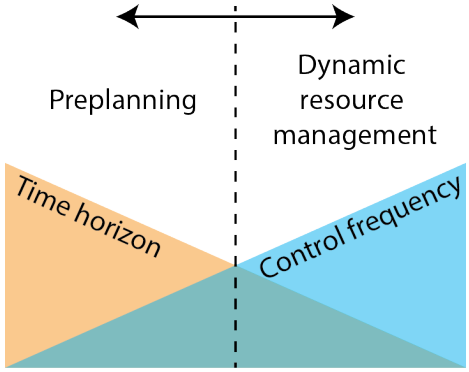


Figure 1. Relationship between preplanning and dynamic sensor management.

The preplanning of sensor placement has been addressed for many different applications. Much of the sensor preplanning literature studies the deployment of sensors in a network for monitoring applications [1, 2]. There is most often an emphasis on finding a sensor deployment strategy that maximizes sensor coverage [3–5] or the probability of detecting some event or object, such as a chemical attack or a foreign ship headed toward local ports [6, 7]. Some work [8–12] has also been focused on the deployment and control of sensor networks based on track quality metrics; however, the targets modeled in these works do not lend themselves easily to the application of missile defense. Additionally, in the breadth of research for sensor networks, the sensors tend to be inexpensive, low-precision, readily available, and easily dispensable—assumptions that do not lend themselves to planning for military applications like defense against ballistic missiles. This division is highlighted by the fact that sensor placement for a typical wireless sensor network usually deals with networks of 50 to 100 relatively inexpensive sensors whereas missile defense has fewer than 10 extremely sensitive, high-precision, very expensive sensors to place. Furthermore, the objective

function for missile defense preplanning is much more sophisticated than the typical objective functions used for general sensor networks; this stems from the need to model and analyze multiple sensor objectives supporting the missile defense mission as opposed to just single event detection.

1.2 OPTIMIZING THE KILL CHAIN

A successful engagement of a threat in a battle scenario relies on the careful execution of four main steps called the *kill chain* (see Figure 2). The role of the sensor is predominant in the first three tasks and can thus be divided into three main objectives: (i) detect, (ii) track, and (iii) discriminate. To accomplish these goals, sensors take measurements during a time-limited duration referred to as *sensor access*; the sensor access period is limited due to the measurement restrictions of the sensor, e.g., sensitivity and range from target. In Figure 3, the sensor access region has been denoted in relation to the timeline of a ballistic missile launch.

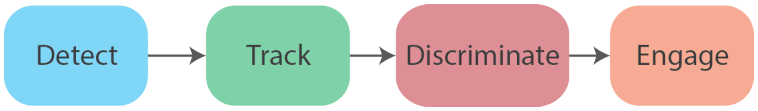


Figure 2. Order of system operations for successful engagement of a threat.

Also in Figure 3, one can see a duration called *weapon access*; this is the period in which it is feasible for the weapon to commit on the intended threat, i.e., to command to launch the intercepting missile. Therefore, the sensor must carry out its detection and tracking objectives before or during the weapon access window. Note that the discrimination task is not confined to the denoted weapon access window as that information can be communicated to the interceptor after interceptor launch.

Because the sensor and weapon systems are often managed independently, there exists a common handshake method to verify that the sensor is effectively supporting the needs of the weapon. This handshake method is based on a metric known as *quality-of-service* that provides a quantitative measure of the sensor data's capability to satisfy the tracking and discrimination requirements of the weapons system. For example, a commonly used metric for tracking quality-of-service is a measure of the track error volume of the sensing system. For the purposes of this report, the focus will be on tracking quality-of-service, although there can also be quality-of-service measures for discrimination.

Thus, from the perspective of the sensor, ensuring engagement support to the weapons is equivalent to providing data that meet a given quality-of-service. Therefore, to interface with the sensor system, the weapon system decides a suitable tracking quality-of-service requirement that needs to be achieved during the weapon access window. With this sensor-weapon architecture, one can observe that an appropriate measure of performance for the sensor is how well it achieves the desired tracking quality-of-service within the weapon access period. In fact, that is the exact objective of this report—to describe a method to properly place and orient a set of sensors such

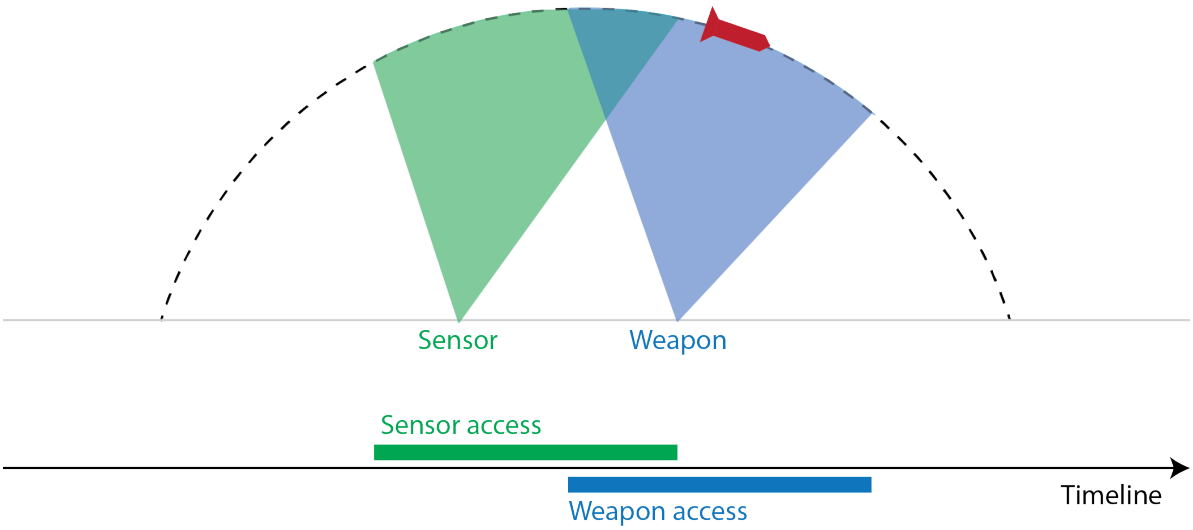


Figure 3. Interaction between sensor and weapons systems in ballistic missile defense.

that the system-level tracking quality-of-service for a raid of threats meets a given set of thresholds within a desired set of time windows.

In previous literature addressing missile defense, a main focus for sensor placement has been to maximize coverage and probability of detection. This would correspond to designing a sensor layout that best optimizes for just one of the kill chain objectives—detection. This is sufficient for many applications, but one can see that this can prove nearsighted in terms of improving performance of the entire kill chain. In fact, optimizing probability of detection or coverage does not reflect many joint sensor effects like geometric diversity and past measurement history. System-level track quality does, however, encapsulate these joint effects through mature techniques like Cramér-Rao lower bound analysis. Note that one could go a step further and optimize based on discrimination, but this report will not address that topic due to the complexity and variety of discrimination algorithms.

1.3 PLANNING UNDER UNCERTAINTY

An inevitability encountered in missile defense is that a priori information about threat characteristics is uncertain and can often be given with varying levels of confidence. This presents the problem of planning the allocation of sensor and weapon resources against a random threat. In particular, one faces the challenge of defining a meaningful measure of effectiveness against a non-deterministic threat. A common approach is to analyze the possible parametric space of the threat and use worst-case performance to measure the effectiveness of a specific layout of sensors; this approach guarantees a bottom-line performance against all of the possible enemy courses of action but is pessimistic and restrictive to the model of the threat. Another approach that many current planning systems employ is to perform Monte Carlo analysis over many different initial scenario parameters, e.g., launch location, intended impact point, and generate estimated probabilities of

events of interest. This method is less constraining than worst-case but only partially illustrates the sensitivities of the defense design to the threat parameters.

The sensitivity of a system-level measure of effectiveness is better encapsulated in a complete histogram or probability distribution, e.g., probability mass function, probability density function (see Figure 4). As is evident from the figure, event probabilities and worst-case performance can be derived from the single probability distribution. Thus, having access to this function allows for complete flexibility in terms of objective function definition, i.e., average, worst-case, best-case, variance. It is therefore a design objective that the planning algorithm constructed in this report generates a probability distribution of a chosen measure of effectiveness.

Defining the uncertainty in the threat is an initial, key step in analyzing the sensitivity of performance to variations in enemy behavior. As mentioned above, this is typically done by defining a possible threat parameter space and testing performance over this space or samples of this space. However, without further sophistication, there are two main limitations: (i) one cannot reflect the relative confidence of a parameter value, and (ii) one cannot define or measure the uncertainty of a parameter or a set of parameters. To this end, a probabilistic model of a raid of ballistic threats is proposed in this work. First, by founding the threat models in probability, relative confidence in threat characteristics can be represented through probability mass, i.e., higher probability reflects higher confidence. The distributions of each threat parameter also reflect how strictly each threat aspect is known, e.g., low variance distributions reflect high certainty in parameter values. In addition, if it is possible to directly map the threat probability distributions to the performance of the system, one can completely sidestep the need for Monte Carlo analysis.

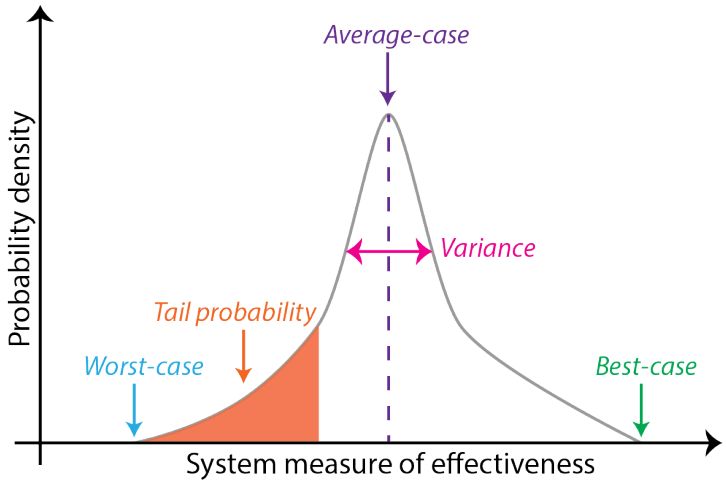


Figure 4. Example probability density function (pdf) of a chosen system measure of effectiveness with various sensitivity metrics.

This report is structured as follows. A methodology for analyzing a layout of sensors against a random threat is defined in Section 2. The proposed analysis requires the construction of a probabilistic model for a raid of ballistic threats and the definition of a metric for system-level

performance. In Section 3, a parametric model for a raid of ballistic threats is proposed with a subsequent definition in probability space. Following this, Section 4 defines the preliminaries for a track-quality-based performance metric. Section 5 selects a system performance metric, describes the sensor placement problem, and provides results for sensor placement analysis.

2. PLANNING FOR UNCERTAIN ENEMY COURSES OF ACTION

The main objective of this work is to analyze the effectiveness of a placement of a set of sensors to handle a set of incoming threats. It is unrealistic to assume that a priori knowledge of the threat characteristics is perfect. Therefore, threat behavior is perceived to be random to the defense, although some aspects of the threat may be more uncertain than others. Thus, not only is absolute system performance relevant when constructing a blue force laydown, but also robustness of the sensor layout to deviations in the perceived enemy course of action.

In this work, the performance of a network of sensors against an uncertain raid of threats is quantified using probabilistic measures of performance. To describe this process, let the flyout of threats be completely described by a set of parameters, Θ, e.g., launch times, launch velocities, number of threats, etc. The set of parameters lies within a known parameter space, $\mathcal{S}_\Theta$, and, due to the uncertainty in the enemy course of action, is assumed random. More specifically, the threat parameter set Θ is drawn according to a given probability density function (pdf), $p(\theta)$, where θ is a realization of Θ. In addition, it is assumed that the joint placement of the network of sensors is uniquely defined by a control vector, $\mathbf{u}$, lying within a control space $\mathcal{S}_u$; this vector can be used to describe sensor aspects such as location, orientation, and sensing mode.

Now, it can be observed that a scenario is completely described by both a realization of the threat parameters, θ, and the defense design information, $\mathbf{u}$, as the enemy course of action and appropriation of defenses has been defined. With the scenario given, one can calculate a measure of performance, $\text{Perf}(\theta, \mathbf{u})$, using an appropriate performance measurement function, $\text{Perf}(\cdot, \cdot)$, such that

$$\text{Perf} : (\mathcal{S}_\theta \times \mathcal{S}_u) \to \mathbb{R}.$$

Note that the performance measurement function can take on many different forms, e.g., the probability of engagement success, number of successful engagements, number of targets detected, and can be quite complex as it will typically rely on analyzing significant portions, if not all, of the kill chain.

An important aspect implicit to the definition of the performance function is that the performance of the sensor network is assumed deterministic given the specific realization of the threat. Realistically, this assumption is untrue as sensor performance depends on many random quantities such as measurement noise and sensor scheduling. However, the main difference between the uncertainty in the threat and the uncertainty in the sensor system is that the defense has the ability to test and accurately characterize the performance of the network of sensors. Further, one can create conservative bounds on the performance of each sensor such that an assumed performance is obtained with high confidence and, thusly, model the efficacy of the sensor as deterministic. For example, one can functionally model the tracking accuracy of a sensor as the accuracy achieved with 97 percent confidence over random measurement noise and sensor scheduling.

Now, it can be seen that the probabilistic nature of the threat parameters induces random behavior in the system performance. This can be observed by defining a new random variable $M \in \mathbb{R}$ representing the performance of the system such that

$$M = \text{Perf}(\Theta, \mathbf{u}).$$

To be able to characterize the statistics of M, it is necessary to calculate its corresponding probability distribution. In general, both the control space, $\mathcal{S}_u$, and the threat parameter space, $\mathcal{S}_\Theta$, can be of mixed type, i.e., consisting of both continuous and discrete variables. This can be seen by noting a sensor may depend on a continuous parameter such as location, but also a discrete parameter like sensing mode. Due to the mixed nature of the threat parameters and/or control variables, the probability distribution of the performance measure may also be of mixed type and difficult to calculate. However, in many practical applications (and in this work), the parameter space, $\mathcal{S}_\Theta$, and control space, $\mathcal{S}_u$, are assumed to be discrete. This can be done by appropriately sampling the continuous threat and sensor variables, e.g., using a grid of possible locations instead of allowing it to be continuous. So, noting that the product space $\mathcal{S}_\theta \times \mathcal{S}_u$ (along with its image) is now also discrete, the performance measure, M, lies in a discrete space. Thus, the behavior of M can be described using a probability mass function (pmf) defined as

$$\Pr[M = m; \mathbf{u}] = \sum_{\theta \in \mathcal{R}(m,\mathbf{u})} \Pr[\Theta = \theta], \tag{1}$$

where the region of summation $\mathcal{R}(m, \mathbf{u})$ is

$$\mathcal{R}(m, \mathbf{u}) = \{\theta \in \mathcal{S}_\Theta : \mathrm{Perf}(\theta, \mathbf{u}) = m\}. \tag{2}$$

This formulation can be derived from the fact that the probability of an event E for a discrete random variable X (with corresponding realization x) can be written as

$$\Pr[E] = \sum_{x \in E} \Pr[X = x].$$

The distribution of M in (1) or a portion of it is commonly approximated by drawing threat parameter samples θ_k for $k = 1, 2, \ldots, n_{\text{trials}}$ from $p(\theta)$ and using Monte Carlo analysis to obtain

$$\Pr[M = m; \mathbf{u}] \approx \frac{1}{n_{\text{trials}}} \sum_{k=1}^{n_{\text{trials}}} \mathrm{I}\{\mathrm{Perf}(\theta_k, \mathbf{u}) = m\},$$

where $\mathrm{I}\{x\}$ is an indicator function such that

$$\mathrm{I}\{x\} = \begin{cases} 1, & x \text{ is true,} \\ 0, & x \text{ is false.} \end{cases}$$

However, this method of approximating $\Pr[M = m; \mathbf{u}]$ can require many trials to get an accurate estimate of the distribution. This can be costly in terms of computation time, especially if the calculation of the performance measure $\mathrm{Perf}(\Theta, \mathbf{u})$ requires significant time. In subsequent sections, it is shown that under certain assumptions Monte Carlo analysis can be sidestepped in favor of directly calculating the probability mass function, although, in some cases, the system and performance measure may be too complicated for this technique; in those cases, Monte Carlo analysis must be used.

Intuitively, if the threats are completely known, the system performance as defined would be deterministic. Thus, the performance of the sensor layout (i.e., the control vector, $\mathbf{u}$) could be analyzed directly. However, since both the threat and system performance are random, one cannot simply obtain a single value but, instead, analyze the sensor layout based on a statistic of the measure of effectiveness, e.g., worst-case, best-case, average. With this, the objective function for the placement of sensors, $\mathcal{O}(\mathbf{u})$, can be generally stated as

$$\mathcal{O}(\mathbf{u}) = E\left[g(M); \mathbf{u}\right], \tag{3}$$

where the expected value is written as $E[\,\cdot\,; \mathbf{u}]$ to emphasize dependence on the control vector, $\mathbf{u}$, and $g(\cdot)$ is a function chosen to represent a desired statistic of the performance—this is described further below. In the case of a discrete performance metric space, the objective function can be expanded to give

$$\mathcal{O}(\mathbf{u}) = \sum_{m} g(m) \Pr[M = m; \mathbf{u}].$$

Note that this form of objective function can be used to employ many common statistics, e.g.,

- Tail probability, $g(M) = \mathrm{I}\{M \leq m_0\}$ or $g(M) = \mathrm{I}\{M \geq m_0\}$ for some threshold m_0,
- Average, $g(M) = M$,
- Variance, $g(M) = (M - E[M])^2$.

This page intentionally left blank.

3. PARAMETRIC MODEL OF A RAID OF BALLISTIC THREATS

It is important to remark that, for simplicity, it is assumed throughout this report that threat trajectories can be approximated by ballistic trajectories, i.e., the threat has reached a burnout velocity at the launch point and travels in a vacuum with constant gravity. However, the methodology presented in this report is not limited to this assumption. Due to the invocation of pure ballistic threat dynamics, the set of threat parameters, Θ, will be constructed to represent the degrees of freedom in a raid of ballistic threats. To start, one can intuitively look at what it takes to define a single ballistic trajectory.

3.1 SINGLE BALLISTIC THREAT

In this section, it will be shown that the trajectory and timing of a ballistic threat can be uniquely determined using the following:

- Launch time
- Time of flight
- Launch location
- Impact location

This parameterization will prove convenient when describing the threat probabilistically. However, to show that these aspects uniquely describe a ballistic trajectory, one must look at the traditional state space definition of ballistic motion.

Let the time-varying, real-valued dynamic state of the single threat be denoted as $\mathbf{x}(t)$. As is commonly done, the threat state vector consists of position and velocity components such that

$$\mathbf{x}(t) = \begin{bmatrix} p(t) \\ v(t) \end{bmatrix},$$

where $p(t)$ is the position of the threat and $v(t)$ is the velocity. The key defining aspect of a ballistic trajectory's shape and impact location is the kinematic state at launch, $\mathbf{x}(\mathrm{T})$, where T is the launch time. Therefore, a more detailed model is constructed for this aspect. More specifically, the launch state can be parameterized by the launch location, the impact location, and the time of flight. Note that the launch velocity is not listed as a parameter due to the constraints of ballistic flight. Once the launch location, impact location, and time of flight have been chosen, the launch velocity is uniquely determined [13] (known as orbit determination or Gauss' problem). This particular parameterization of the launch state is driven by the application of ballistic missile defense and will also prove convenient when investigating different enemy launch doctrines or intended targeting. To formally introduce these parameters, let the time of flight for a ballistic threat be denoted as Γ; then, the launch location and impact locations are simply $p(\mathrm{T})$ and $p(\mathrm{T}+\Gamma)$, respectively.

It is now possible to state the set of parameters necessary for describing the flight of a single ballistic trajectory: the launch time, T; the launch location, $p(\mathrm{T})$; the time of flight, Γ; and the

impact location, $p(\mathrm{T}+\Gamma)$. Thus, the shape and timing of each threat's flight in a raid can be summarized with these four degrees of freedom—this is illustrated further in Figure 5. To this end, it will be convenient to group these degrees of freedom as a single threat parameter set, Λ, such that

$$\Lambda = \{p(\mathrm{T}), p(\mathrm{T}+\Gamma), \mathrm{T}, \Gamma\}. \tag{4}$$

Therefore, a ballistic threat can be completely described by a corresponding set Λ. The total set of parameters for a single ballistic trajectory are summarized in Table 1.

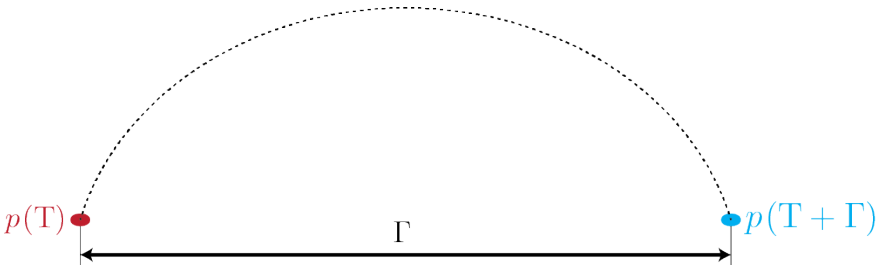


Figure 5. Example ballistic trajectory with launch location $p(\mathrm{T})$, time of flight, Γ, and impact location $p(\mathrm{T}+\Gamma)$.

TABLE 1

Summary of Parameters for a Single Ballistic Trajectory

Parameter	Description
T	Launch time
Γ	Time of flight
$p(\mathrm{T})$	Launch location
$p(\mathrm{T}+\Gamma)$	Impact location
Λ	Set of all parameters for a single threat

3.2 RAID OF BALLISTIC THREATS

Now, consider describing the joint flight of a set of ballistic threats, which instills the need to characterize launch times and launch states for each threat. This task can be somewhat simplified by introducing structure. Initially, one can describe a group of threats launching from a common region. For instance, let a set of threats be launched from a given launch area $\mathcal{L}_\ell$ where $\ell \in \{1, 2, \ldots, L\}$ and L is the total number of launch areas. Due to the assumption of a discrete threat parameter space, each launch area is uniquely defined as a finite set of geographic points; more specifically, let launch area $\mathcal{L}_\ell$ be defined as

$$\mathcal{L}_\ell = \{\omega_{\ell,j} : j = 1, 2, \ldots, \Omega_\ell\},$$

where $\omega_{\ell,j}$ is the j^{th} geographic point in the launch area, and Ω_ℓ is the total number of geographic points describing the launch area.

Each threat now has a unique subscript index such that $\mathbf{x}_{\ell,n}(t)$ is the kinematic state of the n^{th} threat from the ℓ^{th} launch area. It also follows that $n \in \{1, 2, \ldots, N_\ell\}$ where N_ℓ is the number of threats originating from the ℓ^{th} launch area (although there can also be zero threats in a launch area—this is discussed later). Likewise, each threat also has its own unique launch time $\mathrm{T}_{\ell,n}$, time of flight $\Gamma_{\ell,n}$, and threat parameter set $\Lambda_{\ell,n}$. The main constraint imposed by a launch area is that the launch location of each threat within launch area $\mathcal{L}_\ell$ is constrained such that $p_{\ell,n}(\mathrm{T}) \in \mathcal{L}_\ell$ for all n. An example launch area is given in Figure 6.

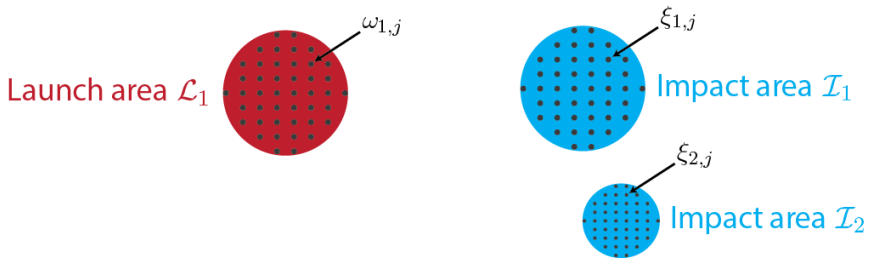


Figure 6. Example scenario with one launch area, $\mathcal{L}_1$, and two impact areas, $\mathcal{I}_1$ and $\mathcal{I}_2$. Also illustrated are the geographic points making up each area. For instance, $\omega_{1,j}$ is a point in the launch area $\mathcal{L}_1$, and $\xi_{2,j}$ is a point in the impact area $\mathcal{I}_2$.

The launch area forms a fundamental building block for the definition of a possible raid. After constructing a model for a launch area, one can build a raid using multiple instances of the launch area model and also define relationships on a launch area level instead of an individual threat level—this will be the path taken in this work. The set of parameters defining the threats of a launch area, Θ_ℓ, can be written as

$$\Theta_\ell = \left\{ N_\ell, \{\Lambda_{\ell,n}\}_{n=0}^{N_\ell} \right\}, \tag{5}$$

which includes the number of threats originating from the launch area and their respective launch locations, impact locations, launch times, and times of flight. To allow for the special case of $N_\ell = 0$ threats, it will be necessary to define $\Lambda_{\ell,0} \triangleq \varnothing$. Note also that the geographic region $\mathcal{L}_\ell$ is not included in the launch area parameter set as it is assumed to be deterministic, i.e., the geographic points in the launch area are not random.

In addition to the concept of a launch area, it is also essential to define the idea of an impact area. Intuitively, these are the regions in which the impact locations for threats may be allocated. Similar to the definition of a launch area, let an impact area $\mathcal{I}_i$ be described by a corresponding finite set of geographic points defined by

$$\mathcal{I}_i = \{\xi_{i,j} : j = 1, 2, \ldots, \Xi_i\},$$

where $\xi_{i,j}$ is the j^{th} geographic point in the impact area, and Ξ_i is the total number of geographic points describing the impact area. Example impact areas are shown in Figure 6.

The notion of the impact area is introduced to constrain the space in which impact locations, $p_{\ell,n}(\mathrm{T}_{\ell,n} + \Gamma_{\ell,n})$, can be placed. Thus, for each threat, it is assumed that its impact location

lies within at least one of the impact areas (possibly multiple if impact areas overlap), i.e., $\exists i \in \{1, 2, \ldots, I\}$ such that $p_{\ell,n}(\mathrm{T}_{\ell,n} + \Gamma_{\ell,n}) \in \mathcal{I}_i$. Constraining the impact space will prove convenient when describing possible targeting doctrines for a given launch area. All parameters for the threats, launch areas, and impact areas are summarized in Table 2. Now, viewing a complete raid as the total set of threat parameters, Θ can then be constructed as

$$\Theta = \bigcup_{\ell=1}^{L} \Theta_\ell. \tag{6}$$

It is also important to remark that as each threat belongs to one and only one launch area [due to the (ℓ, n) indexing convention of the threats], the following is true:

$$\Theta_\ell \cap \Theta_{\ell'} = \varnothing, \text{for } \ell \neq \ell'. \tag{7}$$

TABLE 2

Summary of Launch Area, Impact Area, and Threat Parameters

Parameter	Description
$\mathcal{L}_\ell$	Set of geographic points describing the ℓ^{th} launch area
$\omega_{\ell,j}$	A point in the launch area $\mathcal{L}_\ell$
$\mathcal{I}_i$	Set of geographic points describing the i^{th} impact area
$\xi_{i,j}$	A point in the impact area $\mathcal{I}_i$
$\mathrm{T}_{\ell,n}$	Launch time for the n^{th} threat from launch area ℓ
$\Gamma_{\ell,n}$	Time of flight for the n^{th} threat from launch area ℓ
$p_{\ell,n}(\mathrm{T}_{\ell,n})$	Launch location for the n^{th} threat from launch area ℓ
$p_{\ell,n}(\mathrm{T}_{\ell,n} + \Gamma_{\ell,n})$	Impact location for the n^{th} threat from launch area ℓ
$\Lambda_{\ell,n}$	Set of threat parameters for the n^{th} threat from launch area ℓ
N_ℓ	Number of threats originating from launch area ℓ
Θ_ℓ	Set of all threat parameters for threats launching from launch area ℓ

The disjoint nature of the launch area parameters in (7) will be helpful in describing the joint probabilistic behavior of the launch areas as it provides a straightforward method of describing independence and/or correlation at the launch area level.

3.3 PROBABILISTIC DESCRIPTION OF THREATS

Having defined the threat parameter set Θ, one can now define the corresponding probability distribution over Θ. For this report, it will be assumed that the threat parameter space, $\mathcal{S}_\Theta$, is discrete, and, thus, the distribution of interest is the probability mass function (pmf) $\Pr[\Theta = \theta]$. Note that this pmf can be expanded in terms of the launch area parameter sets, Θ_ℓ, by using (6) as

$$\Pr[\Theta = \theta] = \Pr[\Theta_1 = \theta_1, \Theta_2 = \theta_2, \ldots, \Theta_L = \theta_L], \tag{8}$$

where, as in (6), it follows that $\theta = \bigcup_{\ell=1}^{L} \Theta_\ell$. At this point, it will be convenient to invoke a *launch area independence* assumption that

$$\Pr[\Theta = \theta] = \prod_{\ell=1}^{L} \Pr[\Theta_\ell = \theta_\ell]. \tag{9}$$

Equation (9) implies that there is no assumed coordination of launch processes from one launch area to another. Further, the launch area independence assumption (9) not only simplifies the form of the pmf (8) but also can be seen as a pessimistic measure by assuming ignorance of coordinated launches from one launch area. Something also to notice in the form of (8) and (9) is the absence of the variable L within the probability distribution as the number of launch areas is assumed to be known deterministically. It is straightforward to consider the case in which the number of launch areas is also random, but that will not be considered in this work.

Looking more closely at the launch area pmf $\Pr[\Theta_\ell = \theta_\ell]$, one could further expand the pmf using (5) and the definition of conditional probability as

$$\Pr[\Theta_\ell = \theta_\ell] = \Pr[\Lambda_{\ell,1} = \lambda_{\ell,1}, \Lambda_{\ell,2} = \lambda_{\ell,2}, \ldots, \Lambda_{\ell,n_\ell} = \lambda_{\ell,n_\ell} \,|\, N_\ell = n_\ell] \Pr[N_\ell = n_\ell]. \tag{10}$$

If it can be assumed that threat characteristics within a launch area are drawn independently or, in other words, no threat level coordination is to be modeled, then (10) can be simplified as

$$\Pr[\Theta_\ell = \theta_\ell] = \Pr[N_\ell = n_\ell] \prod_{n=0}^{n_\ell} \Pr[\Lambda_{\ell,n} = \lambda_{\ell,n} \,|\, N_\ell = n_\ell]. \tag{11}$$

With the form of the pmf in (11), one can see that the probability distribution of the launch area depends on the distribution of the number of threats and the individual threat parameter sets, $\Lambda_{\ell,n}$, which describe how launch times, times of flight, launch locations, and impact locations are drawn.

Briefly narrowing focus and studying the individual threat parameter distribution, $\Pr[\Lambda = \lambda]$, more carefully, one can rewrite it using (4) as

$$\Pr[\Lambda = \lambda] = \Pr[\mathrm{T} = \tau, \Gamma = \gamma, p(\tau) = \mu, p(\tau + \gamma) = \eta], \tag{12}$$

where μ and η are realizations of the launch and impact locations, respectively. For this work, it will be assumed that the launch times and times of flight are drawn independently of each other and the launch/impact locations. This implies that (12) can be further decomposed as

$$\Pr[\Lambda = \lambda] = \Pr[\mathrm{T} = \tau] \Pr[\Gamma = \gamma] \Pr[p(\tau) = \mu, p(\tau + \gamma) = \eta]. \tag{13}$$

The probability distribution $\Pr[p(\tau) = \mu, p(\tau + \gamma) = \eta]$ is worth studying in further detail. To do this, consider a single threat launching from the $(\ell^*)^{th}$ launch area, $\mathcal{L}_{\ell^*}$ [and briefly dropping the (ℓ, n) subscript]. The launch and impact locations will be drawn according to the following process:

1. The launch location is drawn from within the launch area $\mathcal{L}_{\ell^*}$ dictated by $\Pr[p(\tau) = \mu]$.

2. The intended impact area is chosen using the probability $\Pr[i = i^*; \ell = \ell^*]$, i.e., the probability that the impact area is $\mathcal{I}_{i^*}$ when the launch area is $\mathcal{L}_{\ell^*}$. The notation $\Pr[i = i^*; \ell = \ell^*]$ is used to emphasize that ℓ is a deterministic parameter and not a random quantity. An example set of distributions can be seen in Figure 7.

3. The impact location is randomly selected within the impact area $\mathcal{I}_{i^*}$ according to $\Pr[p(\tau+\gamma) = \eta \,|\, i = i^*]$.

Thus, the total probability $\Pr[p(\tau) = \mu, p(\tau + \gamma) = \eta]$ for a threat launching from launch area $\mathcal{L}_{\ell^*}$ can be written as

$$\Pr[p(\tau) = \mu, p(\tau + \gamma) = \eta] = \Pr[p(\tau) = \mu] \sum_{i^*=1}^{I} \Pr[p(\tau + \gamma) = \eta \,|\, i = i^*] \Pr[i = i^*; \ell = \ell^*]. \tag{14}$$

An illustration of the full targeting process is given in Figure 8.

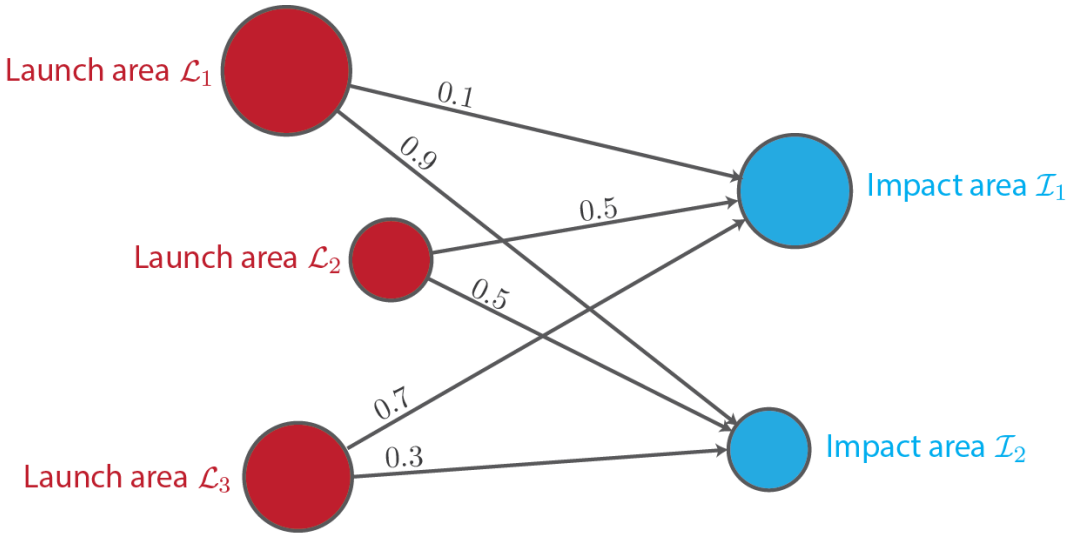
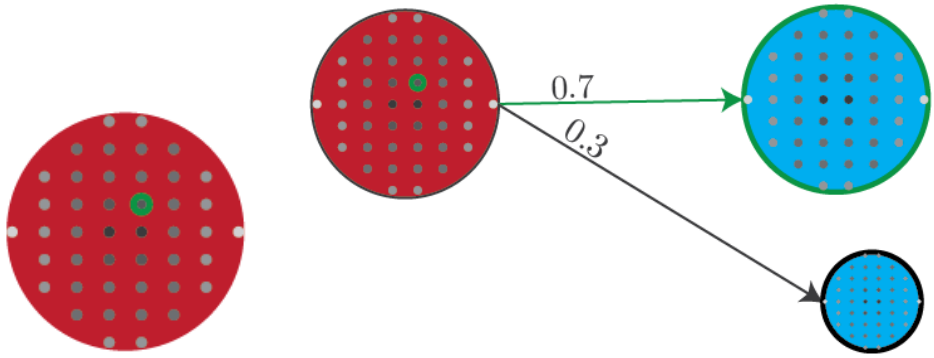


Figure 7. Example targeting probabilities. Note that geographic points within each area are not depicted at this scale.

Now, it is possible to describe the complete threat pmf $\Pr[\Theta = \theta]$ using lower-level probability distributions. This can be observed by noting that (9) can be rewritten using (11), (13), and (14) as

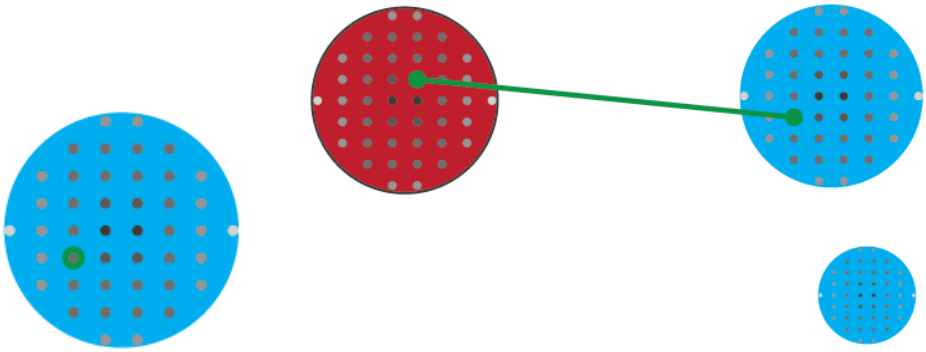
$$\Pr[\Theta = \theta] = \prod_{\ell=1}^{L} \Pr[N_\ell = n_\ell] \prod_{n=0}^{n_\ell} \Pr[\mathrm{T}_{\ell,n} = \tau_{\ell,n}] \Pr[\Gamma_{\ell,n} = \gamma_{\ell,n}] \Pr[p_{\ell,n}(\tau_{\ell,n}) = \mu_{\ell,n}] \cdot \left(\sum_{i^*=1}^{I} \Pr[p_{\ell,n}(\tau_{\ell,n} + \gamma_{\ell,n}) = \eta_{\ell,n} \,|\, i = i^*] \Pr[i = i^*; \ell = \ell^*] \right). \tag{15}$$

The necessary low-level distributions are summarized in Table 3.



(a) Selection of launch location.

(b) Random draw of impact area.



(c) Selection of impact location within selected impact area.

(d) Total realization of launch and impact locations.

Figure 8. Example random targeting realization; all parameter realizations are shown in green. The location probability mass functions are illustrated in grayscale where darker represents higher probability mass and lighter conveys low probability mass.

TABLE 3

Summary of Threat Low-Level Probability Distributions Used to Define $\Pr[\Theta = \theta]$

Distribution	Description
$\Pr[N_\ell = n_\ell]$	Number of threats launched from the ℓ^{th} launch area
$\Pr[\mathrm{T}_{\ell,n} = \tau_{\ell,n}]$	Launch time for the n^{th} threat from the ℓ^{th} launch area
$\Pr[\Gamma_{\ell,n} = \gamma_{\ell,n}]$	Time of flight for the n^{th} threat from the ℓ^{th} launch area
$\Pr[p_{\ell,n}(\tau) = \mu_{\ell,n}]$	Launch location of the n^{th} threat within the ℓ^{th} launch area
$\Pr[i = i^*; \ell = \ell^*]$	Probability that a threat originating from the $(\ell^*)^{th}$ launch area will land in the $(i^*)^{th}$ impact area
$\Pr[p_{\ell,n}(\tau + \gamma) = \eta_{\ell,n} \mid i = i^*]$	Impact location given the threat is landing in the $(i^*)^{th}$ impact area

This page intentionally left blank.

4. SENSORS AND TRACKING

Observing the impending threats is a set of sensors that are controllable in terms of placement and orientation. It is assumed that there are a total of S sensors, each with a time-invariant state vector $\mathbf{u}_s$ for $s = 1, 2, \ldots, S$. The sensor state vector, $\mathbf{u}_s$, contains sensor location and other parameters such as orientation, boresight information, and sensing modes. More specifically, all of the parameters in the sensor state vector are assumed to not vary over the course of the scenario. The total control vector (as mentioned in Section 2) can be constructed by concatenating the sensor state vectors such that

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_S \end{bmatrix}.$$

Each sensor has the capability to take measurements of the kinematic states of visible targets at discrete time instances. For instance, sensor s can take measurements at times $t_{s,k}$ for $k = 1, 2, \ldots, K_s$ where K_s is the total number of possible measurements for sensor s. The number K_s can be derived based on sensor and threat specifications like sensitivity and possible measurement periods. It will be assumed that the measurements for each sensor have been *cued*, meaning that another entity has given prior spatial information of the threat to the sensor to avoid having to search for the threat before tracking. Although it is possible to also include a search operation into the model (i.e., reduction in measurement update rates due to resource usage), it will not be explored in this work.

At each measurement time, $t_{s,k}$, sensor s takes measurements on one or more apparent threats depending on the capabilities of the sensor. However, it is assumed that the sensor has a known, fixed limit on the number of targets on which it can collect simultaneous measurements. Thus, at each discrete measurement opportunity the sensor has to select which visible targets it will measure. This process invokes a *scheduling* mapping from measurement times to threats. More specifically, let the mapping

$$\beta_s : \underbrace{\{1, 2, \ldots, K_s\}}_{\text{measurement time indices}} \rightarrow \underbrace{\{(\ell, n) : \ell = 1, 2, \ldots, L, n = 1, 2, \ldots, N_\ell\}}_{\text{threat indices}},$$

be the scheduling function for sensor s. An example illustration of the scheduling function is given in Figure 9. The $\beta_s(\cdot)$ function directly maps each measurement time to the observed threat or threats—the specific method with which each sensor schedules threat observations will not be optimized in this work but should be considered in preplanning.

Note that in real-time operation this mapping would not be known as the true threat indices are not available to the sensors. However, it will be assumed that, for the purposes of this work, data association is perfect, i.e., the mapping from measurement to threat, $\beta_s(\cdot)$, is known. The assumption of perfect data association is made for multiple reasons. First, simulation of the data association process, e.g., using a multitarget tracking algorithm on multiple random draws of trajectories, process noise, and/or measurement noise, can be computationally expensive and

impractical for the amount of possible trajectories considered in preplanning. Secondly, the tracking performance under the perfect data association assumption can be used as an accurate upper bound on real-time performance.

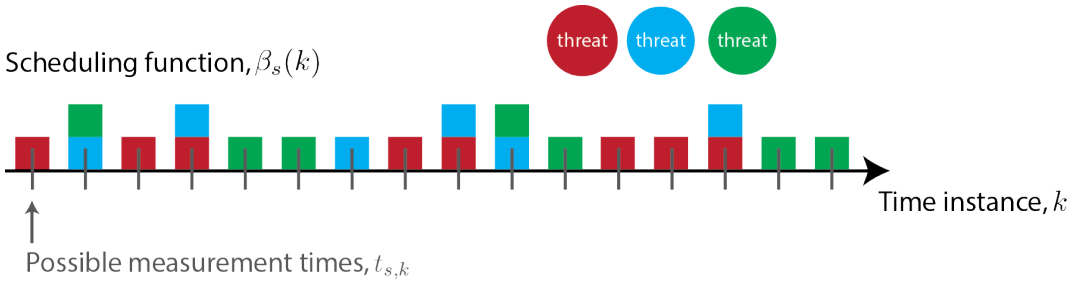


Figure 9. Example scheduling function for a sensor with a maximum of two concurrent measurements and three threats present.

As mentioned in the introduction (see Section 1), this work will focus on defining a measure of performance for the system based on a system-level track quality. At the system level, it is assumed that all of the collected measurements from the sensors are available with negligible delay. With these measurements, the system generates state estimates for each threat, $\hat{\mathbf{x}}_{\ell,n}(t)$, with corresponding state covariance matrices, $\mathbf{P}_{\ell,n}(t)$. The quality of a track is often defined based on its uncertainty; thus, the quality of the track on n^{th} threat from the ℓ^{th} launch area at time t will be defined as

$$q_{\ell,n}(t) = \mathcal{Q}(\mathbf{P}_{\ell,n}(t)), \tag{16}$$

where $q_{\ell,n}(t) \geq 0$ and $\mathcal{Q}(\cdot)$ is a chosen function that maps covariances matrices to nonnegative real numbers, e.g., $\mathrm{tr}((\cdot)^{-1})$, $\log\det(\mathbf{I} + (\cdot)^{-1})$. Usually, it is assumed that track quality, $q_{\ell,n}(t)$, increases with decreasing track uncertainty. It will now be useful to describe how one models the state covariance evolution over the scenario.

4.1 TRACKING A SINGLE THREAT

As it is assumed that the threats undergo the deterministic dynamics of ballistic motion, the uncertainty in the state estimates can be tightly bounded by the Cramér-Rao lower bound as derived in [14]. This will be the primary method used for predicting tracking accuracy in this work. To describe this, one can focus on the multisensor tracking of a single ballistic threat with dynamic state $\mathbf{x}(t)$. Now, by definition, the dynamics of a ballistic target are deterministic, meaning that there is no process noise or unknown state input. In this case, the state of the threat evolves according to the following differential equation:

$$\dot{\mathbf{x}}(t) = \mathrm{f}(\mathbf{x}(t)), \tag{17}$$

where $\mathrm{f}(\cdot)$ is a known, nonlinear function. Assuming that the state of the threat is fixed in position before a launch time T, the trajectory at any point in time can be found using

$$\mathbf{x}(t) = \begin{cases} \mathbf{x}_0, & t < \mathrm{T}, \\ \mathbf{x}(\mathrm{T}) + \int_{\mathrm{T}}^{t} \mathrm{f}(\mathbf{x}(\zeta))d\zeta, & t \geq \mathrm{T}, \end{cases}$$

where $\mathbf{x}_0$ is some constant state of the target before launch. Thus, as the dynamics function $\mathrm{f}(\cdot)$ is known, the trajectory $\mathbf{x}(t)$ after launch is completely determined by the threat state at launch $\mathbf{x}(\mathrm{T})$ (as was observed in Section 3).

Now, for example, assume that all S sensors observe a threat and the measurements are aggregated at a system fusion node. Each measurement can be written in the following form:

$$\mathbf{z}_{s,k} = \mathrm{h}_s(\mathbf{x}(t_{s,k})) + \mathbf{w}_{s,k}, \tag{18}$$

where $\mathbf{z}_{s,k}$ is the real-valued measurement vector obtained at sensor s at time $t_{s,k}$, $\mathrm{h}_s(\cdot)$ is a known (and possibly nonlinear) measurement function, and $\mathbf{w}_{s,k}$ is a sample of white Gaussian measurement noise distributed as $\mathbf{w}_{s,k} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_{s,k})$. The measurement noise is assumed independent from sensor to sensor and time to time. Note also that the measurement covariance model encapsulates many sensor aspects like sensitivity and resolution.

The state covariance, $\mathbf{P}(t)$, can be lower bounded as

$$\mathbf{P}(t) \succeq \mathbf{J}^{-1}(t),$$

where $\mathbf{A} \succeq \mathbf{B}$ if $\mathbf{A} - \mathbf{B}$ is a positive semidefinite matrix and $\mathbf{J}(t)$ is the Fisher information matrix associated with the state estimation problem [15]. In fact, for this report, it will be assumed that $\mathbf{P}(t) \triangleq \mathbf{J}^{-1}(t)$ (which is true for linear Gaussian problems). Now, it was shown in [14] that the matrix $\mathbf{J}(t)$ can be built using the following quantities:

- The Jacobian of the measurement function in (18) evaluated at the true state at the time of measurement,

$$\mathbf{H}_{s,k} = \frac{\partial}{\partial \mathbf{x}} \mathrm{h}_s(\mathbf{x}) \bigg|_{\mathbf{x}=\mathbf{x}(t_{s,k})},$$

- The Jacobian of the dynamics function in (17), $\mathrm{f}(\cdot)$,

$$\mathbf{F}(\mathbf{x}) = \frac{\partial}{\partial \mathbf{x}'} \mathrm{f}(\mathbf{x}'),$$

- The state transition matrix of the threat from time t' to t, $\mathbf{\Phi}(t, t')$, can be obtained by integrating the differential equation

$$\frac{d}{dt}\mathbf{\Phi}(t, t') = \mathbf{F}(\mathbf{x}(t))\mathbf{\Phi}(t, t'),$$

from time t' to t with the initial condition that $\mathbf{\Phi}(t', t') = \mathbf{I}$.

With these definitions, the Fisher information matrix $\mathbf{J}(t)$ can be written as

$$\mathbf{J}(t) = \underbrace{\boldsymbol{\Phi}^{-T}(t,\mathrm{T})\mathbf{P}^{-1}(\mathrm{T})\boldsymbol{\Phi}^{-1}(t,\mathrm{T})}_{\text{propagated initial information}} + \underbrace{\phi(\Delta_1(t),\Delta_2(t),\ldots,\Delta_S(t))}_{\text{accumulated measurement information}}, \tag{19}$$

where $K_s(t)$ is the total number of measurement times at sensor s until time t, $\phi(\cdot)$ is a desired track fusion function, and $\Delta_s(t)$ is the gain in information due to the measurements from sensor s up until time t defined as

$$\Delta_s(t) = \sum_{k=1}^{K_s(t)} \boldsymbol{\Phi}^{-T}(t,t_{s,k})\mathbf{H}_{s,k}^T\mathbf{R}_{s,k}^{-1}\mathbf{H}_{s,k}\boldsymbol{\Phi}^{-1}(t,t_{s,k}). \tag{20}$$

The fusion function $\phi(\cdot)$ allows for different methods of integrating tracks from different sensors. Some special cases would include [16]:

- Ideal track-to-track fusion,

$$\phi(\Delta_1(t),\Delta_2(t),\ldots,\Delta_S(t)) = \sum_{s=1}^{S}\Delta_s(t), \tag{21}$$

- Track-quality-based selection,

$$\phi(\Delta_1(t),\Delta_2(t),\ldots,\Delta_S(t)) = \max_s \mathcal{Q}(\Delta_s^{-1}(t)). \tag{22}$$

4.2 TRACKING MULTIPLE THREATS

Fortunately, extending the calculation of the Cramér-Rao lower bound in (19) to the case of tracking multiple threats is relatively straightforward. In fact, the general form is very similar except that a (ℓ,n) index is required to reference the threat. This can be observed in the following equation for the Fisher information matrix for the n^{th} threat from launch area ℓ:

$$\mathbf{J}_{\ell,n}(t) = \boldsymbol{\Phi}^{-T}(t,\mathrm{T}_{\ell,n})\mathbf{P}_{\ell,n}^{-1}(\mathrm{T}_{\ell,n})\boldsymbol{\Phi}^{-1}(t,\mathrm{T}_{\ell,n}) + \phi\left(\Delta_1(t;\ell,n),\Delta_2(t;\ell,n),\ldots,\Delta_S(t;\ell,n)\right), \tag{23}$$

The main difference lies in the definition of $\Delta_s(t;\ell,n)$,

$$\Delta_s(t;\ell,n) = \sum_{k=1}^{K_s(t)} \boldsymbol{\Phi}^{-T}(t,t_{s,k})\mathbf{H}_{s,k}^T\mathbf{R}_{s,k}^{-1}\mathbf{H}_{s,k}\boldsymbol{\Phi}^{-1}(t,t_{s,k})\mathrm{I}\{\beta_s(k)=(\ell,n)\}, \tag{24}$$

which now requires an indicator function to sum over only the sensor measurements scheduled for that specific threat. It is worth restating that data association is assumed perfect not only at the measurement-to-track level, as in (19), but also at the track-to-track level in (23). Thus, these bounds are achievable yet practical data association complexities are not represented in these expressions.

5. SYSTEM PERFORMANCE AND ANALYSIS

Having now defined the threat parameter set Θ and the control vector $\mathbf{u}$ in Sections 3 and 4, respectively, one can now define the performance measure function $\text{Perf}(\Theta, \mathbf{u})$. In this case, the performance measure will be constructed to reflect the sensor network's capability to achieve a given tracking performance on each threat during some desired time window. To describe this, let the desired track quality for the n^{th} threat from the ℓ^{th} launch area be denoted as $Q_{\ell,n}$. In addition, define a window $w_{\ell,n} \subset [0, \Gamma_{\ell,n}]$ in which it is required that the track quality of the threat surpasses the threshold $Q_{\ell,n}$; this is illustrated in Figure 10. An important implicit assumption of the window $w_{\ell,n}$ is that it is defined *relative to the launch time* as it is defined from zero to the threat time of flight. This makes intuitive sense for if the launch time were shifted by some delay, you would expect the window for the track quality constraint to also move by the same delay. Explicitly, the windowed track quality constraint can be written as

$$q_{\ell,n}(t) \geq Q_{\ell,n} \text{ for some } t \in w_{\ell,n}.$$

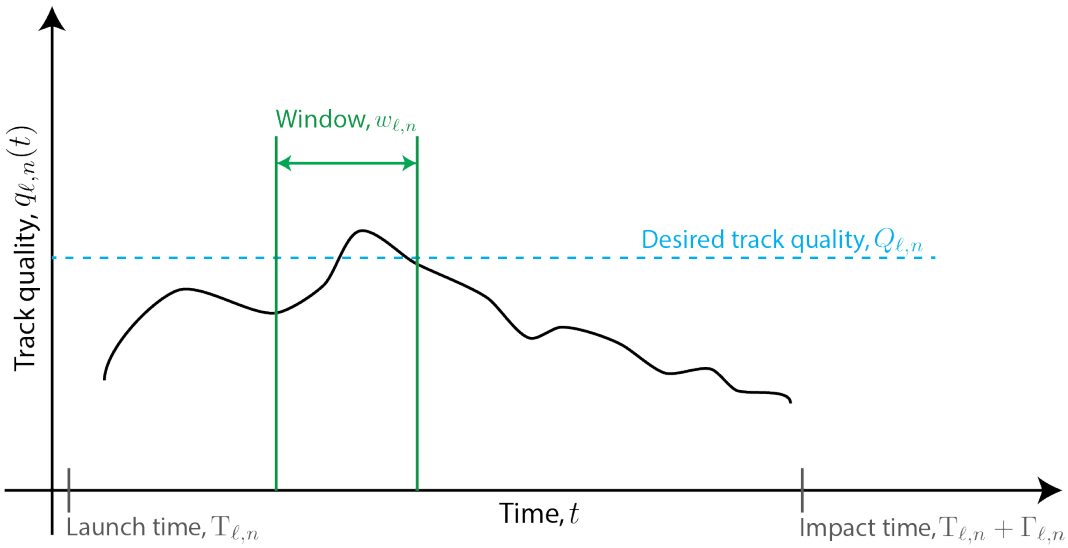


Figure 10. Example track quality plot with window, $w_{\ell,n}$, and desired track quality $Q_{\ell,n}$. It can be seen in this example that the track quality threshold is met during the window.

With the type of track quality constraint depicted in Fig. 10, one can define the following performance metric, $\text{Perf}(\Theta, \mathbf{u})$:

$$\text{Perf}(\Theta, \mathbf{u}) = \sum_{\ell=1}^{L} \sum_{n=1}^{N_\ell} \text{I}\{q_{\ell,n}(t) \geq Q_{\ell,n}, t \in w_{\ell,n}\}, \tag{25}$$

Note that both N_ℓ and $q_{\ell,n}(t)$ implicitly depend on both the threat parameters, Θ, and the sensor control vector $\mathbf{u}$. The performance metric in (25) can be perceived as a total number of threats

that meet the track quality constraint. This will be the primary performance measure used in the simulations presented later in this report.

Now that the performance metric $M = \text{Perf}(\Theta, \mathbf{u})$ has been defined in (25), one can calculate the objective function value for a given sensor control vector (as in Section 2):

$$\mathcal{O}(\mathbf{u}) = \sum_{m} g(m) \Pr[M = m; \mathbf{u}], \tag{26}$$

$$= \sum_{m} g(m) \sum_{\theta \in \mathcal{R}(m,\mathbf{u})} \Pr[\Theta = \theta], \tag{27}$$

where $\mathcal{R}(m, \mathbf{u})$ is

$$\mathcal{R}(m, \mathbf{u}) = \{\theta \in \mathcal{S}_\Theta : \text{Perf}(\theta, \mathbf{u}) = m\}.$$

For the purposes of this report, it will be assumed that the statistic of interest is the expected value of the performance, i.e., $g(M) = M$. Therefore, the final version of the sensor placement objective function can be stated as

$$\mathcal{O}(\mathbf{u}) = \sum_{m} m \sum_{\theta \in \mathcal{R}(m,\mathbf{u})} \Pr[\Theta = \theta]. \tag{28}$$

In general, the probability $\sum_{\theta \in \mathcal{R}(m,\mathbf{u})} \Pr[\Theta = \theta]$ can be difficult to calculate. In the next section, a simplification is introduced so that this probability can be reasonably generated.

5.1 SIMPLIFICATION USING WORST-CASE SENSOR SCHEDULING

For the necessary calculation of the performance metric probability mass function in (25), one has to be able to calculate

$$\sum_{\theta \in \mathcal{R}(m,\mathbf{u})} \Pr[\Theta = \theta] = \sum_{\theta \in \mathcal{S}_\Theta} \Pr[\Theta = \theta] \mathrm{I}\{\text{Perf}(\theta, \mathbf{u}) = m\}, \tag{29}$$

$$= \Pr\left[\text{Perf}(\Theta, \mathbf{u}) = m\right], \tag{30}$$

$$= \Pr\left[\sum_{\ell=1}^{L} \sum_{n=1}^{N_\ell} \mathrm{I}\{q_{\ell,n}(t) \geq Q_{\ell,n}, t \in w_{\ell,n}\} = m\right], \tag{31}$$

where (29) is possible due to the definition of $\mathcal{R}(m, \mathbf{u})$ in (2). Equation (30) is possible due to the definition of the indicator function, $\mathrm{I}\{\cdot\}$, and (31) can be obtained using the definition of the performance metric in (25).

Something important to notice is that, in general, the track quality constraint events, $\{q_{\ell,n}(t) \geq Q_{\ell,n}, t \in w_{\ell,n}\}$, are not necessarily independent. This is due to the fact that sensor schedules can be expected to change depending on how many targets are currently visible. However, this dependence can be mitigated if a specific scheduling doctrine is assumed at each sensor. More specifically, if it is known ahead of time the maximum number of targets that could be seen simultaneously, the sensor schedule can be constructed employing a worst-case measurement frequency to guarantee that all objects are measured in the given measurement interval. For example, a sensor may have an ideal measurement frequency of 3 Hz (i.e., one measurement opportunity every 0.33 seconds)

with a maximum of one concurrent measurement. But if it is known that a maximum of 5 threats can be seen, the sensor scheduling can be approximately modeled as having an update rate of $\frac{3}{5}$ Hz with a maximum of 5 concurrent measurements. An illustration of this scheduling method is given in Figure 11.

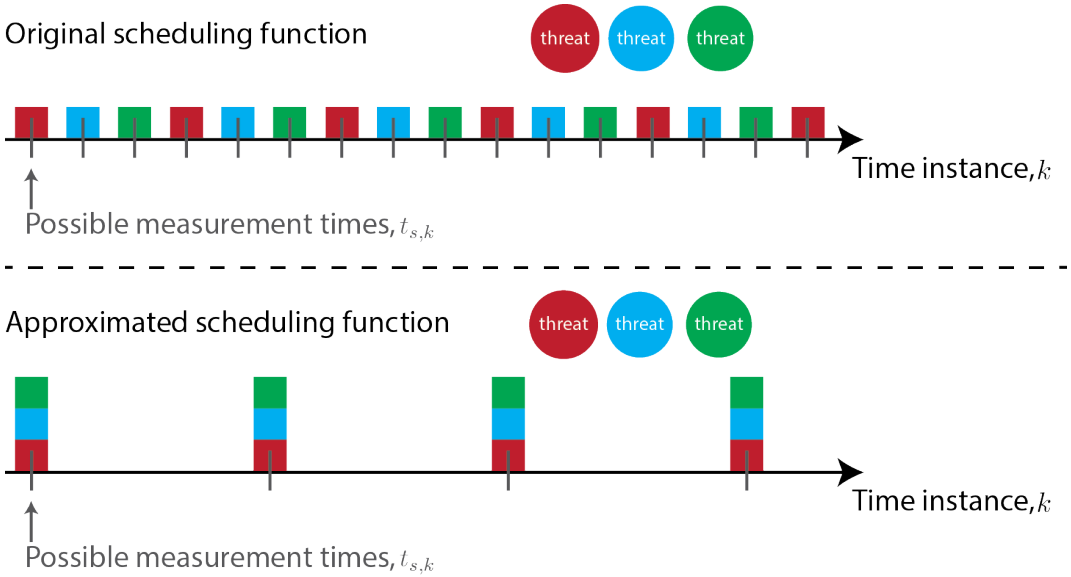


Figure 11. Example scheduling function with its corresponding approximated worst-case scheduling function.

The scheduling approximation allows for the decoupling of track quality constraint events as the sensor schedules are constant and no longer depend on how many targets are present at each measurement time. Thus, for a given set of launch times, it can be observed that no matter how many threats are launched, each sensor will schedule the same set of measurements for each threat.

With the decoupling of track qualities afforded by approximated, worst-case scheduling, the pmf (31) can be rewritten as

$$\Pr\left[\sum_{\ell=1}^{L}\sum_{n=1}^{N_\ell} \mathrm{I}\{q_{\ell,n}(t) \geq Q_{\ell,n}, t \in w_{\ell,n}\} = m\right] = \Pr\left[\sum_{\ell=1}^{L} A_\ell = m\right], \tag{32}$$

where $A_\ell = \sum_{n=1}^{N_\ell} \mathrm{I}\{q_{\ell,n}(t) \geq Q_{\ell,n}, t \in w_{\ell,n}\}$ are independent random variables. This follows from the fact that functions of sets of independent random variables are also independent, i.e., if random variables X and Y are independent, then $u(X)$ and $v(Y)$ are also independent for some functions $u(\cdot)$ and $v(\cdot)$. Now, using the form of the pmf of a sum of independent random variables [17], Equation (32) can be rewritten as

$$\Pr\left[\sum_{\ell=1}^{L} A_\ell = m\right] = \sum_{\substack{a_1,a_2,\ldots,a_L: \\ \sum_{\ell=1}^{L} a_\ell = m}} \prod_{\ell=1}^{L} \Pr[A_\ell = a_\ell]. \tag{33}$$

It can be shown that the sum over a_ℓ in (33) is equivalent to the L-fold convolution of the pmfs $\Pr[A_\ell = a_\ell]$.

Looking more closely at the pmf $\Pr[A_\ell = a_\ell]$, one can expand this probability as

$$\Pr[A_\ell = a_\ell] = \sum_{\alpha=0}^{\infty} \Pr\left[\sum_{n=1}^{\alpha} \mathrm{I}\{q_{\ell,n}(t) \geq Q_{\ell,n}, t \in w_{\ell,n}\} = a_\ell \middle| N_\ell = \alpha\right] \Pr[N_\ell = \alpha], \tag{34}$$

$$= \sum_{\alpha=0}^{\infty} \binom{\alpha}{a_\ell} \left[\nu_\ell^{a_\ell}(1-\nu_\ell)^{\alpha-a_\ell}\right] \Pr[N_\ell = \alpha], \tag{35}$$

where (34) is obtained by using the definition of A_ℓ, the equality (35) can be produced by noting that A_ℓ is a sum of independent indicator functions that can be modeled as a binomial random variable, and ν_ℓ is the probability that a single threat from launch area ℓ will meet the track quality constraint. This probability can be written as

$$\nu_\ell = \sum_{\theta_\ell} \Pr[\Theta_\ell = \theta_\ell] \mathrm{I}\{q_{\ell,1}(t) \geq Q_{\ell,1}, t \in w_{\ell,1}\},$$

where the $(\ell, 1)$ subscript is used to emphasize that the probability is being defined for a single isolated threat from launch area ℓ. By the earlier observation that the ability to meet the track quality constraint is invariant to number of targets, the probability ν_ℓ can be simplified as

$$\nu_\ell = \sum_{\theta_\ell} \Pr[\mathrm{T}_{\ell,1} = \tau_{\ell,1}] \Pr[\Gamma_{\ell,1} = \gamma_{\ell,1}] \Pr[x_{\ell,1}(\tau_{\ell,1}) = \mu_{\ell,1}, x_{\ell,1}(\tau_{\ell,1} + \gamma_{\ell,1}) = \eta_{\ell,1}] \cdot \tag{36}$$
$$\mathrm{I}\{q_{\ell,1}(t) \geq Q_{\ell,1}, t \in w_{\ell,1}\}.$$

5.2 RESULTS

In this section, example scenarios are used to illustrate the defined performance measure and how it varies with sensor layout. This is performed by evaluating the objective function (28) over many different locations and orientations of a controllable sensor. This procedure can also be thought of as an illustration of optimization by exhaustive search. To begin, a notional sensor model is introduced that is subsequently used in the examples.

5.2.1 Sensor Model

For the purposes of this work, it is assumed that the sensor is radar-like. The control parameters for the radar include its geographic position along with its orientation in the azimuth and elevation dimensions. The orientation of the sensor needs to be described as it is assumed its visibility is limited in both in azimuth and elevation (see Figure 12). Thus, the control vector $\mathbf{u}_s$ for sensor s can be written

$$\mathbf{u}_s = [p_s, \phi_s, \psi_s]^T,$$

where p_s is the position of the sensor, ϕ_s is the azimuth boresight angle, and ψ_s is the elevation boresight angle of the sensor.

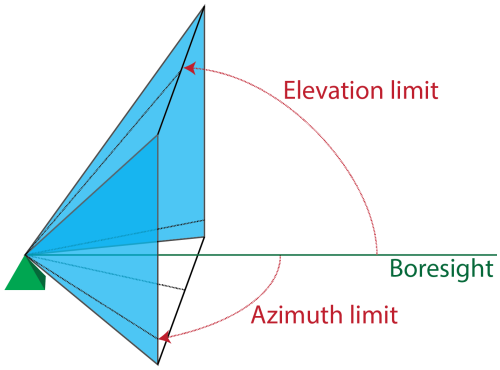


Figure 12. Radar-like sensor visibility model, i.e., azimuth-limited and elevation-limited.

Another required definition for the model of the sensor is the measurement covariance, $\mathbf{R}_{s,k}$. In this case, it is assumed that measurements are made in range, azimuth, and elevation. Thus, the measurement covariance is of the form

$$\mathbf{R}_{s,k} = \begin{bmatrix} \sigma_r^2 & 0 & 0 \\ 0 & \sigma_\phi^2 & 0 \\ 0 & 0 & \sigma_\psi^2 \end{bmatrix},$$

where σ_r, σ_ϕ, σ_ψ are the standard deviations of range, azimuth, and elevation, respectively. For this notional sensor model, these values are chosen to be $\sigma_r = 5$ m, $\sigma_\phi = 1$ milliradian, and $\sigma_\psi = 1$ milliradian. The measurements are limited to be within 60° of azimuth boresight, above 2° elevation, and within a maximum range of 1000 km from the sensor. Additionally, the nominal measurement rate of the sensor is 10 Hz.

5.2.2 Track Quality

For the following scenarios, the measure of track quality, $\mathcal{Q}(\cdot)$, is split into two components: position track quality $\mathcal{Q}_p(\cdot)$, and velocity track quality, $\mathcal{Q}_v(\cdot)$. These are defined for a track covariance matrix $\mathbf{P}$ as

$$\mathcal{Q}_p(\mathbf{P}) = 3 \cdot \max_{j=1,2,3} \sqrt{P_{jj}},$$

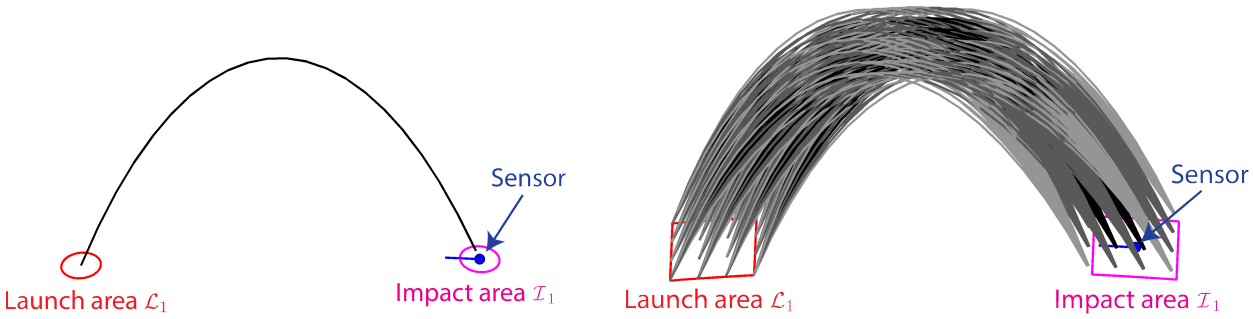
$$\mathcal{Q}_v(\mathbf{P}) = 3 \cdot \max_{j=4,5,6} \sqrt{P_{jj}},$$

where P_{jj} is the $(j,j)^{th}$ entry of $\mathbf{P}$. These track quality measures equate to the worst 3-σ dimension of the covariance in position and velocity, respectively. For the notional examples, the track quality constraints will be defined as $\mathcal{Q}_p(\mathbf{P}) \leq 2$ km and $\mathcal{Q}_v(\mathbf{P}) \leq 12$ m/s for a window of $[300, 370]$ seconds after launch.

5.2.3 Example Scenarios

In the first set of scenarios, the effect of increasing uncertainty in your threat parameter space is studied. Initially, the scenario in Figure 13a is assumed. In this case, every aspect of a single

threat launch is completely determined with probability one, i.e., the launch location, the impact location, the number of threats (one). For this point case, the distribution of the number of threats serviced is given in Figure 14a. It can be seen that the track quality constraint is met for the single threat with probability one. Now, it is assumed that the launch and impact locations are uncertain. Due to this uncertainty, the launch and impact areas have grown; this can be seen in Figure 13b. Due to the variation in trajectory shape from launch to impact, there is now a nonzero probability of not meeting the track quality constraint on the threat. This is shown in the pmf in Figure 14b. Finally, the number of threats is now allowed to be uncertain. In Figure 14c, one can see the effect of now assuming that the number of threats launched in the scenario in Figure 13b is a uniform random variable from one to seven.



(a) Scenario 1a

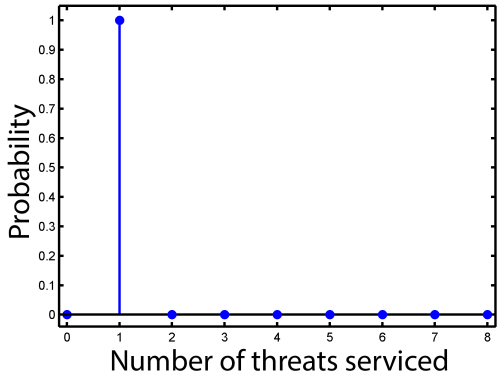
(b) Scenario 1b and 1c

Figure 13. Scenarios 1a, 1b, and 1c.

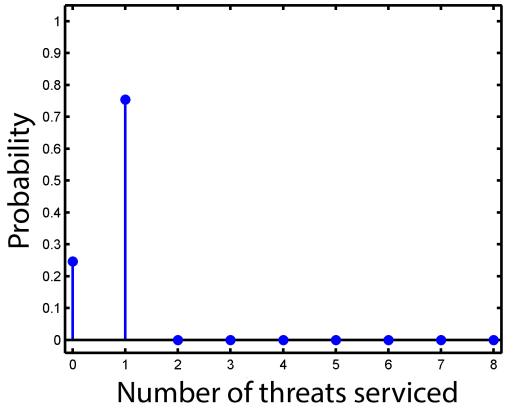
In the second scenario, there are two launch areas and two impact areas as depicted in Figure 15. Each of the two launch areas consist of $\Omega_1 = \Omega_2 = 16$ geographic points. The two impact areas are both made of $\Xi_1 = \Xi_2 = 4$ points each. Launch area $\mathcal{L}_1$ draws the number of threats N_1 as a uniform random variable from 1 to 7. Launch area $\mathcal{L}_2$ draws its number of threats, N_2, as a uniform random variable from 1 to 10. Launch area locations and impact locations are all drawn uniformly, e.g., the launch location of a threat from launch area $\mathcal{L}_2$ is equally likely to be from one of its $\Omega_2 = 16$ geographic points. To simplify analysis, worst-case scheduling will be employed; this implies that the effective rate of the sensor will be $10/17$ Hz ≈ 0.6 Hz as 17 is the maximum number of concurrent threats. In this scenario, it is chosen that targeting probabilities are symmetric. To see this, the target probabilities can be written compactly as a matrix

$$\Pr[i = i^*; \ell = \ell^*] = \begin{bmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{bmatrix},$$

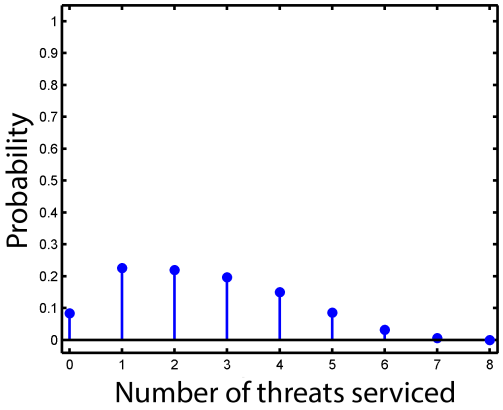
where the rows correspond to launch area index ℓ^* and the columns correspond to impact area index i^*. It can be seen that written this way the target probabilities exhibit symmetry; the probability of targeting the further impact area is equal for each launch area. This imposes a spatial symmetry as seen in Figure 15.



(a) Scenario 1a



(b) Scenario 1b



(c) Scenario 1c

Figure 14. Probability mass functions for Scenarios 1a, 1b, and 1c.

The position and orientation of a sensor is varied over a given operating area (i.e., geographic region where the sensor can be placed) in between the two impact areas. A small circle is shown for every possible sensor position. At each position, an arrow shows the orientation with the best objective function value, and the color of the arrow shows the objective function value attained at that position and direction. Red dots indicate sensor positions with identically zero performance over all possible directions. It can be seen in this example that placing the sensor closer to the launch areas and the center of the two impact areas tends to give the largest objective function values with a maximum of 6.3547 serviced on average.

In the third example scenario shown in Figure 16, the target probabilities are no longer symmetric. More specifically, the targeting probability matrix is now defined as

$$\Pr[i = i^*; \ell = \ell^*] = \begin{bmatrix} 0.1 & 0.9 \\ 0.1 & 0.9 \end{bmatrix}.$$

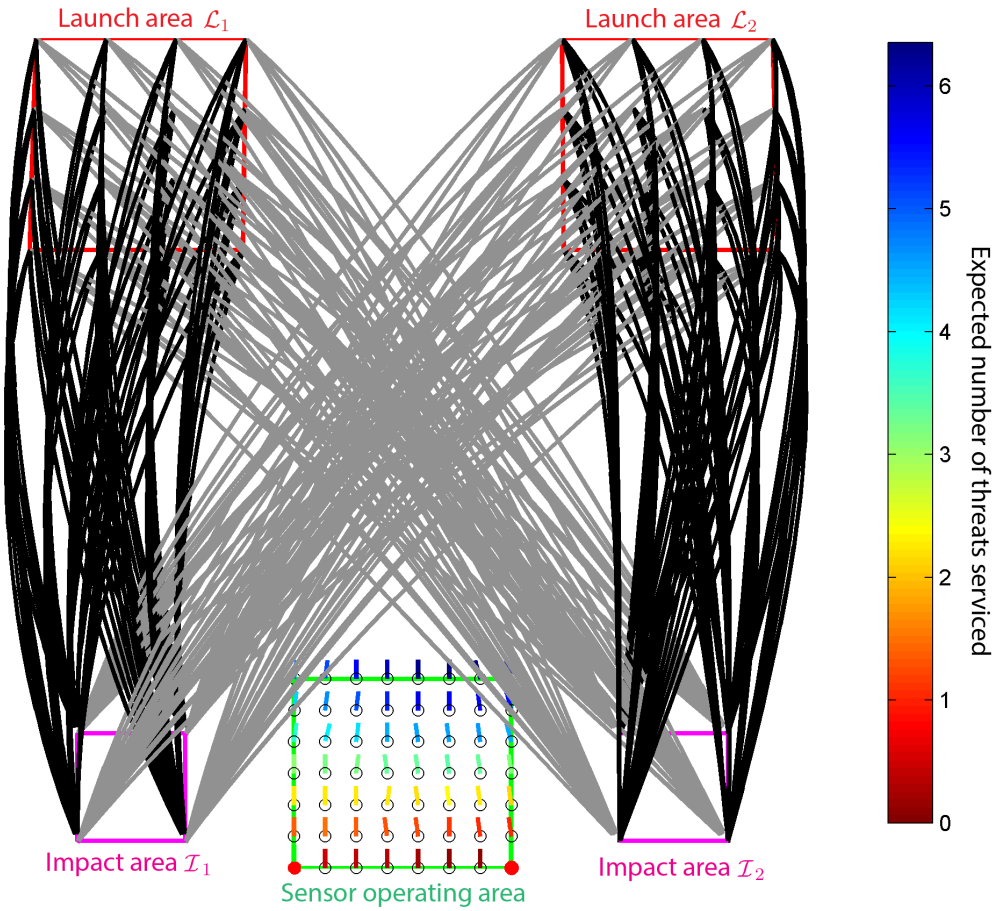


Figure 15. Example scenario with two launch areas, two impact areas, and one sensor. The targeting probabilities are assumed symmetric. The colorbar shows the expected number of threats serviced. The probability of each trajectory is shown in grayscale—the darker the trajectory, the higher the probability mass. Maximum object value is attained by placing the variable sensor at the top-middle of the operating area with an objective value of 6.3547 number of threats serviced on average.

Now, both launch areas are much more likely to send threats toward impact area $\mathcal{I}_2$. This changes the sensor placement results from the previous scenario in two ways: (i) the positions of highest objective value now favor the trajectories toward impact area $\mathcal{I}_2$ as can be seen by the rightward shift of the optimal sensor placements, (ii) the maximum attained objective value increases to 6.7234. The increase in highest objective value is due to the fact that the threats tend to be closer in space now that they favor an impact area. This allows the sensor to narrow its focus and visibility to possibly see threats earlier and generate better quality tracks.

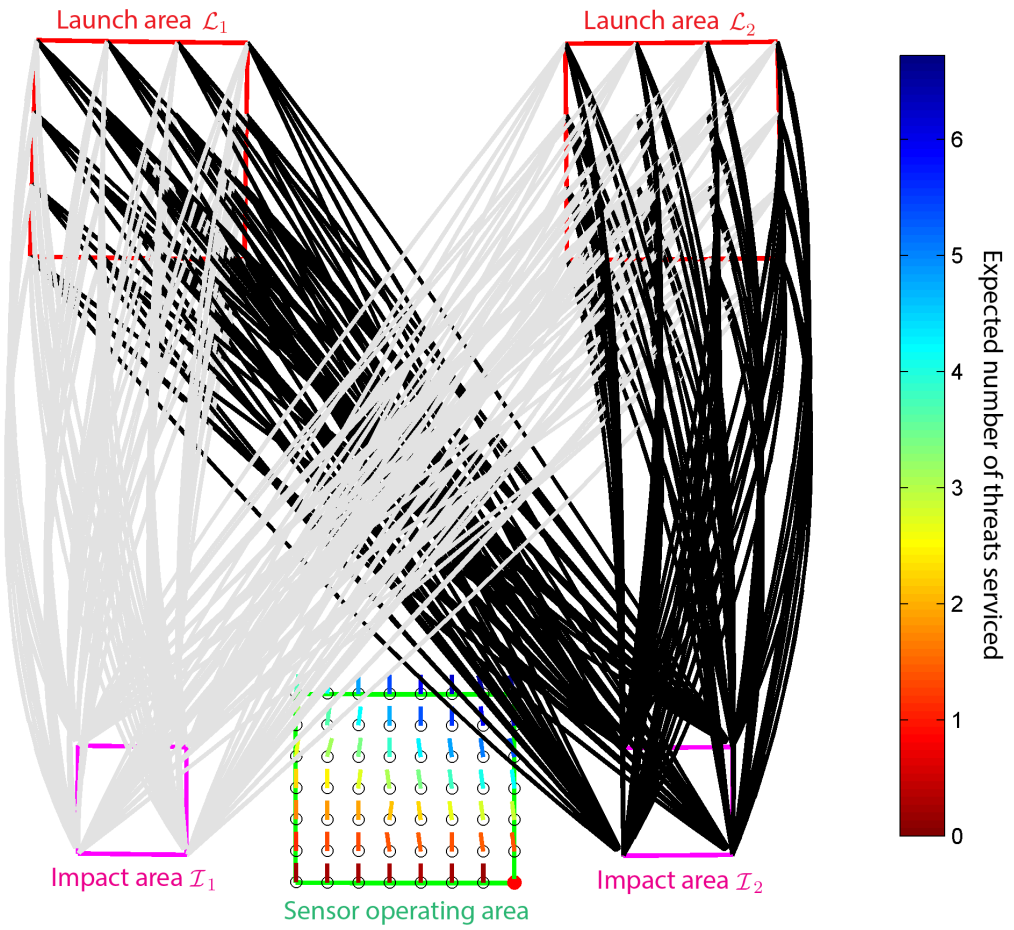


Figure 16. Example scenario with two launch areas, two impact areas, and one sensor. The targeting probabilities are assumed asymmetric. The colorbar shows the expected number of threats serviced. The probability of each trajectory is shown in grayscale—the darker the trajectory, the higher the probability mass. Maximum object value is attained by placing the variable sensor at the top-right of the operating area with an objective value of 6.7234 number of threats serviced on average.

Finally, in the fourth scenario (Figure 17), the effect of an additional sensor is investigated. As in the first scenario, the target probabilities are symmetric, i.e.,

$$\Pr[i = i^*; \ell = \ell^*] = \begin{bmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{bmatrix}.$$

However, now there is an additional sensor to the west of the impact area $\mathcal{I}_1$. It is assumed to be identical to the variable sensor in terms of measurement rate, visibility, etc. Further, the tracks from both sensors are combined together using ideal fusion as in (21). As with changing the spatial symmetry of the threats in the second example, the incorporation of another sensor induces some changes to the objective function values attained for the variable sensor. First, it largely moves the

highest objective value toward the threats not seen by the fixed sensor. This illustrates the effects of cooperation between sensors as the best layouts in this case spread the two sensors to better accommodate the spatial extent of the raid. Second, it further increases the maximum objective value obtained by the sensor network to 6.8688. This is intuitive as it is expected that the more sensors one has available, the better the ability to provide engagement support to the weapons systems.

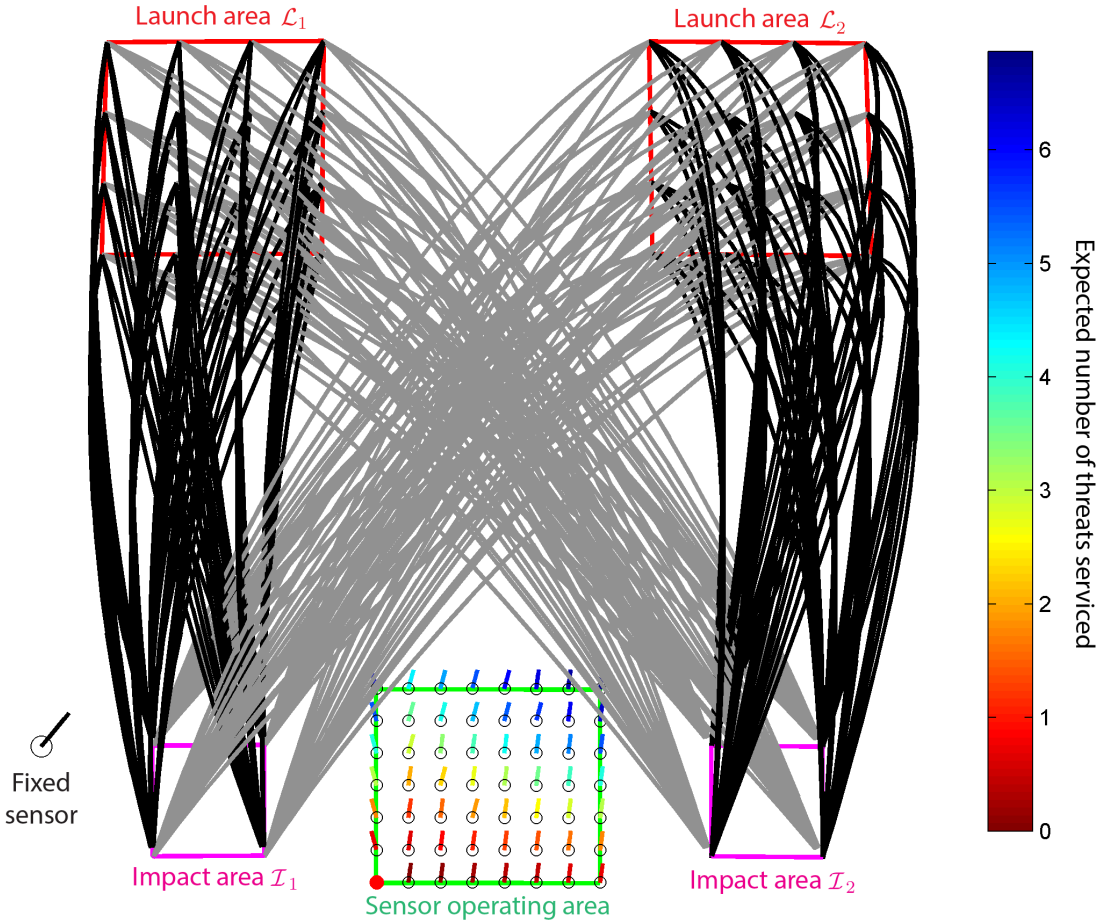


Figure 17. Example scenario with two launch areas, two impact areas, and two sensors (one fixed). The targeting probabilities are assumed symmetric. The colorbar shows the expected number of threats serviced. The probability of each trajectory is shown in grayscale—the darker the trajectory, the higher the probability mass. Maximum object value is attained by placing the variable sensor at the top-right of the operating area with an objective value of 6.8688 number of threats serviced on average.

6. CONCLUSIONS

This report investigates the analysis of sensor layout for the purpose of providing engagement support when defending against large raids of threats. In addition, it is assumed that the a priori characterization of the threats is described with varying levels of certainty. To perform this task, an objective function was developed to quantify the best sensor layout against a set of threats that is random in nature. This was followed by the proposal of a parametric model for a raid of threats that assume ballistic trajectories. The parametric model was then used to define the threat in probability space. To quantify the success of the network of sensors in supporting engagements, a performance metric grounded in system-level track quality is derived and evaluated using the state estimation Cramér-Rao lower bound for ballistic dynamics. With the probabilistic model and performance metric defined, sensor placement analysis is performed over a set of chosen scenarios to illustrate example output. It is also shown that if certain sensor scheduling assumptions are made, a closed-form probability mass function for the performance metric can be directly calculated.

Although the focus of this work is on the analysis of sensor placement, an equally important task is that of optimization. The main role of sensor placement analysis is to impart intuition to the warfighter on how system performance varies within allowable degrees of freedom. With this knowledge, he/she can enforce high-level system constraints and subsequently optimize on the defense design accordingly. However, optimization of the sensor placement can also be used to inform the warfighter as to bounds on performance and illustrate regions of optimality. The topic of efficient optimization of the system performance over sensor placement is left for future work.

This page intentionally left blank.

APPENDIX A: IMPLEMENTATION

The inherent complexity of scenarios grows very quickly in a number of various dimensions, e.g., number of launch locations, number of targeted defended assets, number of threats. As the scenarios increase in complexity, the larger the number of required tracking quality-of-service calculations. In addition, the tracking calculations further depend on the sensor layout; thus, as the number of possible sensor layouts increase, so does the number of tracking calculations. To stem the tide of combinatorics, there are a few useful tools and mathematical exploits that are used to make runtime for the planning process manageable. The first that will be discussed is a utilization of the fact that information is additive across sensors. The second technique relies on parallel processing to efficiently calculate tracking quality-of-service. The third is an intuitive pruning of the feasible solution space in terms of sensor layout.

A.1 INFORMATION ADDITIVITY

For a given combination of sensor locations and orientations, the measurement and measurement covariance matrices for these location/orientation pairs play a role in the track quality for the threat. It turns out that by switching the order of summation in the computation of the target information matrix, that the problem can be cast in a more simplified form. Rather than compute a target information matrix for each combination of sensor positions/orientations, we now compute a target information matrix for each individual sensor position and orientation combination, and then, find the positions/orientations whose matrices combine in an optimal fashion. The joint sensor information matrix can be decomposed into a sum of single sensor information matrices. This allows optimization of sensor positions and bearings to take place in two steps. First, single sensor information matrices are computed for each position and bearing. Then, if $S = 1$ sensor is used, we compute, for each target (ℓ, n), whether the objective function at a given sensor position and bearing exceeds the stated threshold. Then, after this is done for each target and sensor position/orientation, we choose the position or subset of positions that are able to service the greatest number of targets. If $S > 1$ sensors are used, the initial computation of single sensor information matrices is the same. Now, however, for each target (ℓ, n), and a given combination of S sensor positions and bearings, we add the single sensor information matrices and evaluate the objective function. Then, as before, we pick the joint sensor position/bearing that services the most targets.

We begin this section with some analysis on the computation needed to compute the different matrices required for analysis and optimization. Because the information gain,

$$\phi\left(\Delta_1(t;\ell,n), \Delta_2(t;\ell,n), \ldots, \Delta_S(t;\ell,n)\right),$$

in (23), includes information from all S sensors, in order to compute the above quality-of-service metric for all possible combinations of sensor locations, the measurement matrices, $\mathbf{H}_{s,k}$, and the measurement covariances, $\mathbf{R}_{s,k}$, must be computed for the entire timeline (up to and including the deadline for each target). This means that for a given sensor, the number of times that these matrices are computed is

$$|\mathcal{L}_s||\varphi_s|K_s,$$

where $\mathcal{L}_s$ is the discrete of possible locations and φ_s is the discrete set of possible boresight angles for sensor s. Note that if there are S sensors and each sensor s can take one of $|\mathcal{L}_s|$ locations and φ_s orientations, then there are $\prod_s |\mathcal{L}_s||\varphi_s|$ possible combinations of sensor locations and orientations. In addition to computation of the measurement and measurement covariance matrices, computing the track quality on a given target requires computation of the target's dynamics transition matrices. Also, there is the complexity required in computing a log-determinant of a matrix.

In this section it is shown that the optimization over sensor location and boresight bearing can be reduced greatly. In fact, the quality-of-service on a given trajectory can be decomposed as a sum of individual information matrices from the individual sensor locations. In the absence of process noise, the tracking quality-of-service at the deadline can be analyzed in closed form due to (23). For a general quality-of-service metric $\mathcal{Q}(\cdot)$, the quality-of-service for target (ℓ, n) at time t can be rewritten as

$$q_{\ell,n}(t) = \mathcal{Q}\left(\mathbf{\Phi}^{-T}(t, \mathrm{T}_{\ell,n})\mathbf{P}_{\ell,n}^{-1}(\mathrm{T}_{\ell,n})\mathbf{\Phi}^{-1}(t, \mathrm{T}_{\ell,n}) + \phi\left(\Delta_1(t;\ell,n), \Delta_2(t;\ell,n), \ldots, \Delta_S(t;\ell,n)\right)\right), \tag{A.1}$$

by using (23), and, for simplicity, it is assumed that the launch time for the target of interest, (ℓ, n), will be zero (i.e., $\mathrm{T}_{\ell,n} = 0$). Note that the only part of (A.1) that can be optimized by controlling the deployment of the sensors is

$$\phi\left(\Delta_1(t;\ell,n), \Delta_2(t;\ell,n), \ldots, \Delta_S(t;\ell,n)\right).$$

What this result allows is that instead of computing the joint information matrix for target (ℓ, n) for each combination of sensor locations and boresight orientations, it is now possible to compute the individual information matrices from each individual sensor location and orientation once. Then, for a given combination of sensor locations/boresights, it is possible to simply combine the single sensor information matrices. More specifically, when optimizing over different combinations of sensor locations and boresight angles, the single-sensor information matrices are computed for each location/boresight. Then, for each joint configuration, the corresponding matrices are added and the quality-of-service is computed. Therefore, it is still necessary to consider an exponential number of sensor configurations; however, the expensive computation of the coordinate conversions, $\mathbf{H}_{s,k}$, and the matrix multiplications in (A.1) are done a linear rather than an exponential number of times, and it is here where the savings are gained.

Going further, if the trace-inverse quality-of-service is employed such that $\mathcal{Q}(\cdot) = \mathrm{tr}\left((\cdot)^{-1}\right)$, then

$$q_{\ell,n}(t) = \mathrm{tr}\left(\mathbf{\Phi}^{-T}(t, \mathrm{T}_{\ell,n})\mathbf{P}_{\ell,n}^{-1}(\mathrm{T}_{\ell,n})\mathbf{\Phi}^{-1}(t, \mathrm{T}_{\ell,n}) + \phi\left(\Delta_1(t;\ell,n), \Delta_2(t;\ell,n), \ldots, \Delta_S(t;\ell,n)\right)\right), \tag{A.2}$$

$$= \mathrm{tr}\left(\mathbf{\Phi}^{-T}(t, \mathrm{T}_{\ell,n})\mathbf{P}_{\ell,n}^{-1}(\mathrm{T}_{\ell,n})\mathbf{\Phi}^{-1}(t, \mathrm{T}_{\ell,n})\right) + \mathrm{tr}\left(\phi\left(\Delta_1(t;\ell,n), \Delta_2(t;\ell,n), \ldots, \Delta_S(t;\ell,n)\right)\right). \tag{A.3}$$

Recall from Section 4, if performing ideal track-to-track fusion, the fusion function becomes

$$\phi(\Delta_1(t), \Delta_2(t), \ldots, \Delta_S(t)) = \sum_{s=1}^{S} \Delta_s(t).$$

With the ideal form of fusion, the second term of (A.3) can be expanded as

$$\begin{aligned}\phi\left(\Delta_1(t;\ell,n),\Delta_2(t;\ell,n),\ldots,\Delta_S(t;\ell,n)\right) &= \sum_{s=1}^{S}\operatorname{tr}\left(\Delta_s(t;\ell,n)\right), \\ &= \sum_{s=1}^{S}\operatorname{tr}\left(\sum_{k=1}^{K_s(t)}\boldsymbol{\Phi}^{-T}(t,t_{s,k})\mathbf{H}_{s,k}^{T}\mathbf{R}_{s,k}^{-1}\mathbf{H}_{s,k}\boldsymbol{\Phi}^{-1}(t,t_{s,k})\mathrm{I}\{\beta_s(k)=(\ell,n)\}\right), \qquad \text{(A.4)} \\ &= \sum_{s=1}^{S}\sum_{k=1}^{K_s(t)}\operatorname{tr}\left(\boldsymbol{\Phi}^{-T}(t,t_{s,k})\mathbf{H}_{s,k}^{T}\mathbf{R}_{s,k}^{-1}\mathbf{H}_{s,k}\boldsymbol{\Phi}^{-1}(t,t_{s,k})\mathrm{I}\{\beta_s(k)=(\ell,n)\}\right). \qquad \text{(A.5)}\end{aligned}$$

Similar to above, an indicator function allows the sums to be interchanged:

$$\sum_{k=1}^{K_s(t)}\sum_{s=1}^{S}\operatorname{tr}\left(\boldsymbol{\Phi}^{-T}(t,t_{s,k})\mathbf{H}_{s,k}^{T}\mathbf{R}_{s,k}^{-1}\mathbf{H}_{s,k}\boldsymbol{\Phi}^{-1}(t,t_{s,k})\right)\mathrm{I}\{\beta_s(k)=(\ell,n)\}.$$

Note that in the case of a trace-inverse quality-of-service, the interchanging of sums is even more useful as it can be done outside of the trace operation. Thus, only the quality-of-service values at the deadline for each sensor have to be stored and later added over sensors to get the overall quality-of-service.

A.2 PARALLEL COMPUTATION OF COORDINATE CONVERSIONS

Historically, the processing power of a computer was generally related to the computational speed of the central processing unit (CPU). The need for increased computational performance drove CPUs to operate at increasingly higher speeds, expanding the number of computations that could be performed in a limited amount of time. However, even with increasingly capable manufacturing techniques, physical limitations prevent single cores from growing their speed ad infinitum. Recently, additional computational power has been achieved by including multiple CPU processing cores onto a single chip within the computer, effectively scaling the number of computations it can perform. Each of the cores in this parallel CPU architecture possess significant speed and processing power; but as the number of these cores grows, so does the cost and power consumption of the computing hardware. Most recently, the growing popularity of three-dimensional high-definition video gaming has driven an increasing need for affordable and low-power computers capable of performing a large number of independent calculations in real time. To meet this challenge, video card manufacturers designed the graphics processing unit (GPU), and later extended the cards capabilities to allow for general purpose computing on graphics processing units (GPGPU).

The raw power of the GPU arises from the sheer number of independent calculations it can perform in parallel as the result of having hundreds of processing cores on a single card. Although each of these cores processes the data at a slower rate than a CPU core otherwise would, the computations can be done in parallel over hundreds of cores rather than serially on just a handful. A possible limiting factor, however, is the time that is required to transfer data between the computers memory and the GPU memory (and back). This transfer takes place over a PCIe interface that is much slower than the CPU-RAM interface and consumes a significant amount more data transfer

time. To maximize the power of the GPU and reduce total processing time, data transferred to the GPU should remain within the units memory until all computations on the data are completed. The more parallel computations performed on a single dataset and fewer data transfers between the CPU memory and the GPU memory, the more time will be gained from using the GPU.

For our application, the calculations that best lent themselves to GPU processing were the coordinate conversions from a global three-dimensional Cartesian coordinate system to a localized spherical coordinate system. These calculations are highly parallelizable and are comprised of simple arithmetic and trigonometric calculations. The provided MATLAB GPU computing toolbox employs an impressive range of overloaded functions to support many existing functions to be run on the GPU, so no special coding was required within the coordinate conversion toolbox to make it GPU-compatible.

As mentioned before, GPU computations only outperform powerful parallel CPUs when performing many parallel computations on large datasets. In our case, our planner design was designed and built for the purpose of operating on a large number of threat trajectories over large spans of space and with numerous sparsely dispersed sensors. This problem inherently possesses the qualities perfect for GPU computation. As seen in Figure A.1, as the number of trajectories grows and the sampling interval falls, the number of computations required increases rapidly. This problem is exacerbated when additional sensors, and thus additional sensor-threat geometries, are included in the analysis. The number of computations versus the number of trajectories is shown in Figure A.1.

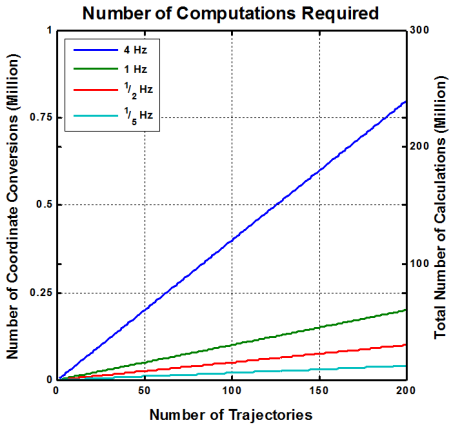


Figure A.1. Number of computations required versus number of trajectories.

Because of the large number of calculation operations required on a fixed set of data, and on such a highly parallelizable dataset (each coordinate conversion is independent from other coordinate conversions), we would expect a noticeable speed-up in the total time to perform this function. Comparing the time required to perform the same number of six-state coordinate conversions (1 million) with associated covariances, we see that the GPU performs the same operations, resulting

in the same answer as the CPU in a fraction of the time (1.6× speed-up). A comparison of GPU and CPU runtimes for coordinate conversions is illustrated in Figure A.2.

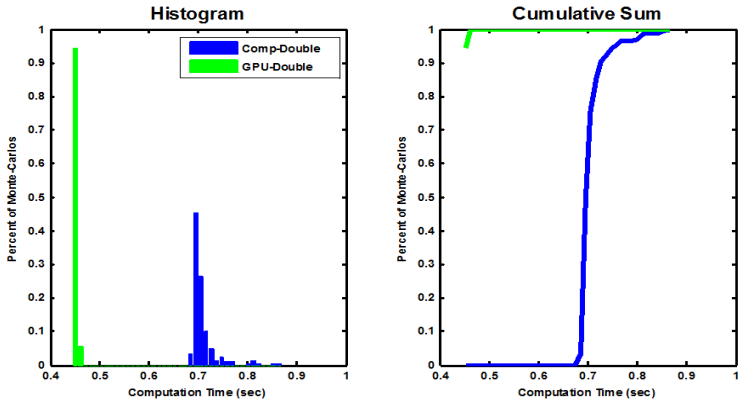
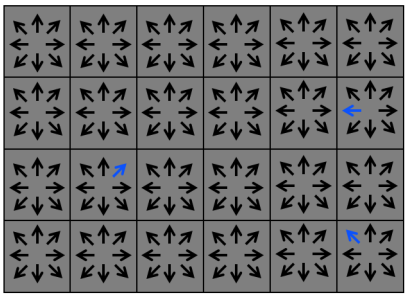


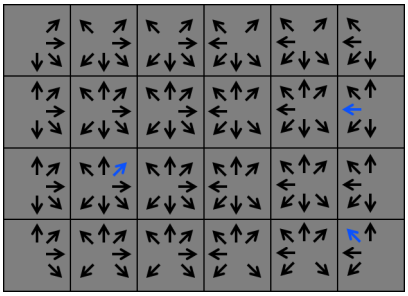
Figure A.2. Comparison of runtimes for GPU and CPU coordinate conversions with double precision.

A.3 SOLUTION SPACE PRUNING

The single sensor pruning function takes as input the initial set of azimuth values φ_s and the set of possible sensor locations $\mathcal{L}_s$ and determines for each location, the subset $\varphi_s^* \subset \varphi_s$ of valid azimuth angles for that sensor location. How a particular angle $\varphi \in \varphi_s$ is determined to be valid or not is based on whether the sensor would be able to observe the threat trajectory at any point during its flight. This is done by comparing the field of regard of the sensor with the angle subtended by the geodesic line connecting the launch and impact points with the sensor position. For each sensor position, the set φ_s^* of valid azimuth angles for location is constructed by testing each angle $\varphi \in \varphi_s$ at that position. Initially, we let $\varphi_s^* = \varphi_s$. Illustrations of the algorithm and its pseudocode can be seen in Figures A.3, A.4, and A.5.

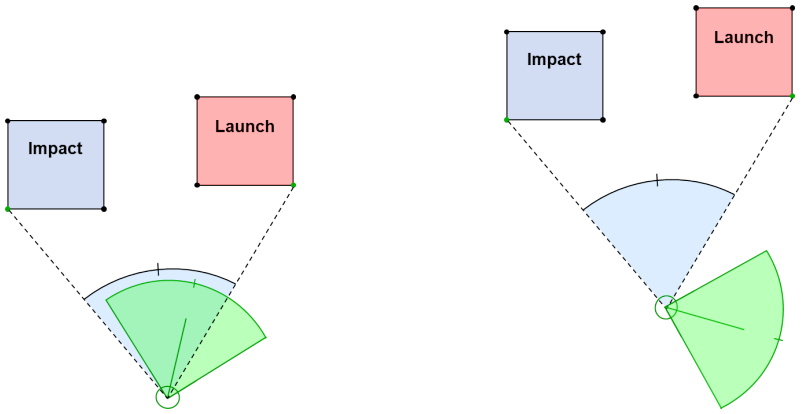


(a) Sensor location grid with azimuth angles



(b) Sensor location grid with azimuth angles after pruning

Figure A.3. Illustration of solution space pruning.



(a) Example accepted boresight angle (b) Example pruned boresight angle

Figure A.4. Illustration of pruning based on measurement feasibility.

Input: list of sensor locations $\mathcal{L}_s$ and default list of azimuth angles φ_s
Output: sets $\{\varphi_s^* : s \in \mathcal{S}_s\}$ of valid azimuth angles for each sensor location
for sensor position x_s **do**
compute geodesic angle (wrt to due North) from sensor position to launch site
compute geodesic angle from sensor position to impact site
let ψ be the angular extent spanned by these two geodesic lines
for boresight angle $\varphi \in \varphi_s$ **do**
Let F be the field of regard with boresight angle φ
if F intersects ψ **then**
include φ in φ_s^*
end if
end for
end for

Figure A.5. Algorithm for single-sensor pruning.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine* 40(8), 102–114 (2002).

[2] M. Tubaishat and S. Madria, "Sensor networks: an overview," *IEEE Potentials* 22(2), 20–23 (2003).

[3] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. Srivastava, "Coverage problems in wireless ad-hoc sensor networks," in *Proceedings of IEEE INFOCOM* (2001), vol. 3, pp. 1380–1387.

[4] S. Dhillon and K. Chakrabarty, "Sensor placement for effective coverage and surveillance in distributed sensor networks," in *Proceedings of IEEE Wireless Communications and Networking Conference* (2003), vol. 3, pp. 1609–1614.

[5] M. de Berg, O. Cheong, M. van Kreveld, and M. Oversmars, *Computational Geometry: Algorithms and Applications*, Springer, 3rd ed. (2008).

[6] J. Watson, H. Greenberg, and W. Hart, "A multiple-objective analysis of sensor placement optimization in water networks," in *Proceedings of Critical Transitions in Water and Environmental Resources Management* (2004), pp. 1–10.

[7] J. Berry, L. Fleischer, W. Hart, C. Phillips, and J. Watson, "Sensor placement in municipal water networks," *Journal of Water Resources Planning and Management* 131, 237–243 (2005).

[8] R. Rajagopalan, R. Niu, C. Mohan, P. Varshney, and A. Drozd, "Sensor placement algorithms for target localization in sensor networks," in *Proceedings of IEEE Radar Conference* (2008), pp. 1–6.

[9] O. Erdinc, J. Areta, P. Willett, and S. Coraluppi, "Multistatic sensor placement: a tracking perspective," in O.E. Drummond (ed.), *Signal and Data Processing of Small Targets* (2005), *Proceedings of SPIE*, vol. 5913.

[10] H. Wang, K. Yao, and D. Estrin, "Information-theoretic approaches for sensor selection and placement in sensor networks for target localization and tracking," *Journal of Communications and Networks* 7(4), 438–449 (2005).

[11] S. Martínez and F. Bullo, "Optimal sensor placement and motion coordination for target tracking," *Automatica* 42, 661–668 (2006).

[12] J. Spletzer and C. Taylor, "Dynamic sensor planning and control for optimally tracking targets," *The International Journal of Robotics Research* 22(1), 7–20 (2003).

[13] R. Bate, D. Mueller, and J. White, *Fundamentals of Astrodynamics*, Dover Publications, Inc. (1971).

[14] J. Taylor, "The Cramér-Rao estimation error lower bound computation for deterministic nonlinear systems," *IEEE Transactions on Automatic Control* 24(2), 343–344 (1979).

[15] S. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*, Prentice Hall (1993).

[16] Y. Bar-Shalom, P. Willett, and X. Tan, *Tracking and Data Fusion: A Handbook of Algorithms*, YBS Publishing (2011).

[17] A. Papoulis and S. Pillai, *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill (2002).

This page intentionally left blank.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
06-03-2018	Project Report	

4. TITLE AND SUBTITLE

Sensor Placement Analysis for Defense Against Uncertain Raids of Ballistic Threats

5a. CONTRACT NUMBER
FA8721-05-C-0002 and/or FA8702-15-D-0001

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Zachary T. Chance, Steven R. Vogl, and Lori D. Layne

5d. PROJECT NUMBER
1971

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

MIT Lincoln Laboratory
244 Wood Street
Lexington, MA 02421-6426

8. PERFORMING ORGANIZATION REPORT NUMBER

MD-51

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Missile Defense Agency
5700 18th Street, Bldg 245
Fort Belvoir, VA 22060-5573

10. SPONSOR/MONITOR'S ACRONYM(S)
MDA

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

13. SUPPLEMENTARY NOTES

13. ABSTRACT

Sensor management lies at the crux of engagement support, and effective deployment of sensors is an important component of a battle management system. Sensors need to be properly placed to provide suefficient coverage for detection, tracking, and threat identication capabilities. This becomes particularly challenging in scenarios where the available sensor resources are strained by a large number of concurrent threats. However, many such dicult scenarios can be dealt with if sensors can cooperate and information can be effectively combined. Thus, selection and placement of sensor resources for complex scenarios should be done at a system level (i.e., where information from all sensors is available) to account for potential performance gains enabled by the fusion of data from multiple sensors. To this end, this work constructs a mathematical framework for the analysis of effective deployment of multiple sensors with the objective of supporting weapons systems. In addition, it is assumed that the defense is posed against a large number of concurrent threats characterized by uncertain a priori information.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE		44	19b. TELEPHONE NUMBER *(include area code)*