Control Based Sensor Management for a Detection Scenario

Yong Xun and Mieczyslaw M. Kokar
Department of Electrical and Computer Engineering
Northeastern University, Boston, Massachusetts
`yxun@ece.neu.edu` and `kokar@coe.neu.edu`
Kenneth Baclawski
College of Computer Science
Northeastern University, Boston, Massachusetts
`kenb@ccs.neu.edu`

Abstract

In this paper we apply control theory techniques to a dynamic sensor resource management problem. The sensors monitor sources (emitters) of electronic radiation in the environment (such as radars), emitting in different frequencies and patterns. The goal is to tune the sensor to the appropriate frequency bands, so as to maximize the probability of detection of the illuminations. We show that our control-theory based sensor manager outperforms a commonly used scheduler.

Keywords: real-time scheduling, real-time resource allocation, sensor scheduling, sensor management

1 Introduction

Modern avionics systems carry a number of sensors for collecting data about the environment. Normally the demands for sensing are much higher than the physical capabilities that the sensors can provide. As a consequence, the sensors must be used selectively, balancing the conflicting sensing requests from the point of view of the goal of the mission of the avionic system. The research domain that deals with controlling sensors is called *Sensor*

Management. According to [1], the goal of a sensor management system is to "select the right sensor to perform the right service on the right object at the right time".

In this paper we consider the problem of controlling a sensor that monitors (detects) sources of electronic radiation in the environment (such as radars). The sources emit radiation in different frequencies and with various illumination patterns and periodicities. For a sensor to detect an illumination by an emitter, the sensor must be tuned to a frequency band that contains the frequency being emitted, and the sensor must be sensing at the time when the illumination occurs. Because of the large number of actual and potential emitters in an environment, it is important to minimize the time spent attempting to observe each emitter, while also maximizing the probability of detecting an illumination by an emitter. It is also important to keep frequency bands available for other uses, such as jamming. This results in additional pressure to reduce the time scheduled for the sensor to perform observations in each frequency band.

A generally agreed upon model for sensor management (cf. [1]) is to view it as a loop consisting of three major functions: *generate options, prioritize options* and *schedule tasks*. The first function generates request for sensor tasks, the second assigns priorities to the tasks, and the third then assigns the time of execution to each task. In this paper we consider all three functions. Since in the subject domain the solution to this sensor management problem is termed as "scheduling", in the rest of this paper we use this term to mean inclusively "option generation, prioritization and scheduling".

The environment that we deal with is highly dynamic. Emitters are rotating and a sensor is located on a moving platform. In the general case, emitters can move, and illumination patterns can vary over time. Furthermore, events can occur over a very large range of time scales. Individual radiation pulses may be as brief as a nanosecond, while illumination periodicities can be as long as several seconds. The background of sensors is developed in more detail in Section 2. Additionally, since the scheduler is decoupled from the task generation and prioritization functions, the scheduler may introduce delays in scheduling the sensor (cf. [2]).

To compensate for the dynamics of the environment and of the system, we build and use a control based resource manager; we call it Dynamic Scan Scheduler (DSS). We formalize the dynamics of this resource allocation problem in Section 4.2. In order to map the problem onto the control architecture, we must also have a well defined quality of service (QoS) metric that specifies the required level of service. In addition to providing a metric for comparing the quality of various algorithms, the QoS is used as part of the control architecture. Various choices for the QoS are discussed in Section 4.3.

The next step is to choose a suitable control architecture. Our methodology is based on a general architecture for self-controlling software [3] which is specialized to this particular resource allocation problem. In this case, we chose a two-level feedback control architecture. We discuss our rationale for this choice in Section 4.1.

Finally, we evaluate our solution and compare it with the other commonly used approaches used to solve this particular problem. The results of our simulations and comparisons are presented in Section 5. The most commonly used scheduler uses a schedule determined in advance of a mission. This type of scheduler is called a fixed scan scheduler (FSS), introduced in Section 3. An algorithm for constructing fixed scan schedules is presented, and some examples are given. Although this scheduling technique is the most commonly used one, it does not have very good performance, and the reasons for this are discussed in some detail.

We also consider another scheduling technique that tries to use prior knowledge about the emitters to achieve optimum performance. This technique is introduced in Section 3.3. This alternative scheduling technique is dynamic but not adaptive. We show that the control-theory based scheduler outperforms both of the other schedulers. Furthermore, our solution uses only about 10% of the observation time as the fixed scan schedule, and our solution does not depend as much on a priori knowledge about the emitters as the second scheduling technique. The main reason for the performance advantages of our solution is that neither of the other two schedulers can adapt to dynamic changes in the environment. We end the paper in Section 6 with our conclusions and an outline of future research directions.

Control based approaches to sensor management have been used, e.g., [2, 4]. The main goal of the control theoretic approaches is to address problems with a optimization approach proposed in [5] and then extended by many others. One of the directions was to use information theory based performance metrics [6]. Although similarly as in [2] in our scenario we also have an architecture in which the scheduling function is separated from the option generation function, and thus we have a similar problem with the delays due to this architecture, we could not use these results to our scenario mainly because these results are related to tracking, rather than detection. One of the distinguishing features of our scenario is that our system has discrete input (detection or non-detection).

Feedback and control based algorithms have also been used in the area of real-time systems, where there is a need for scheduling CPU for computational tasks (cf. [7, 8, 9, 10]). This work, however, addresses only one of the problems of sensor management, i.e., the scheduling part.

2 Basic Background of Emitters and Sensors

The sensors being considered in this paper are receivers located on a moving platform such as an aircraft. When aircraft are operating in hostile territory, attempts will be made to detect and to track their movements. This is most often done by using ground or missile based radars. Such radars emit electromagnetic radiation that illuminates the aircraft. Radiation reflected from the surfaces of the aircraft is used by the radar for detection, identification and tracking purposes. We refer to these radars as *emitters.*

Because of the tactical importance of these radars, it is important for aircraft to detect that they are being illuminated. For this reason, aircraft are equipped with receivers that attempt to sense when the aircraft is being illuminated by an emitter. The aircraft receivers will be called *sensors.* A sensor can detect an emitter if the sensor is within the beam width from the central beam line of the emitter's antenna and within the detection range.

It is normally assumed that the kinds of emitter that might be encountered during a mission are known in advance. In particular, the system has basic data on the frequencies used by the emitters and the times between successive illuminations when the emitters are operating. It is not normally assumed that one will know when a given kind of emitter will begin operating or how long it will operate.

In Figure 1 we show a typical emitter illumination pattern. The radiation is produced in short pulses. These, in turn, form batches called illuminations. The pulses are shown here as square pulses, but the actual pulses are high frequency waveforms. The line over each illumination group is meant to represent the envelope of the illumination. It is not another signal.

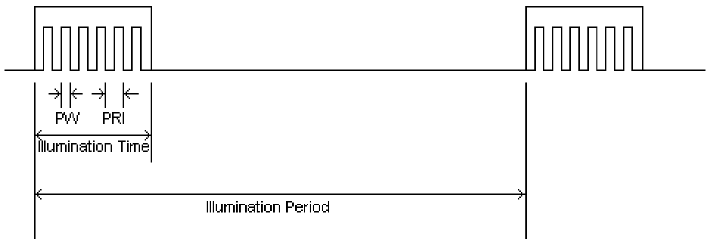

Figure 1: Signal

While the pulses are produced electronically, the illuminations are usually the result of the mechanical motion of the radar antenna. This has some important consequences. The time between pulses can be very short and can be very regular. On the other hand, the time between successive illuminations is typically much longer and need not be as regular.

The most important emitter parameters are the following: f_e - the frequency of the emitter's signal (waveform); A_e - signal strength (power level) of the emitter; T_{pri} - Pulse Repetition Interval (PRI) of the emitter (time between successive pulses within a single illumination); T_{pw} - pulse width (PW) of the emitter; T_{eip} - the illumination period (time between successive illuminations, determined by the rotation period of the radar); B_e - beam width of the emitter (angle scope within which the emitter is radiating at a given time, determines the angle within which a target can be detected); T_{eit} - illumination time (duration of a single illumination); τ_e - the minimum time required for a detection of the emitter against background noise.

2.1 Sensor Control and Scheduling

A sensor is controlled by giving it a command called a *Control Description Word* (CDW). A CDW specifies when and for how long the sensor is to be tuned to a particular frequency band. Each CDW commands a single dwell of the sensor. Mathematically, a CDW is a triple:

$$CDW = (t_{ds}, \tau_d, \omega) \tag{1}$$

where t_{ds} is the time when the dwell begins and τ_d is the length of time that the receiver devotes to this dwell. A *scan schedule* (SS) is a sequence of CDWs, one per dwell:

$$SS = \{CDW_1, CDW_2, ..., CDW_n\} \tag{2}$$

A scan schedule is simply the sequence of commands given to a receiver. Within a scan schedule, one can define the notion of the *revisit time*, τ_{rv}, the time from the beginning of the last dwell until the beginning of the next dwell on a particular frequency band. From the beginning time of the last dwell and the revisit time, one can compute the start time of the next dwell.

The most basic constraint that any sensor must satisfy is called the *capacity constraint*. This constraint simply states in mathematical form the fact that a sensor can only be tuned to one frequency band at a time. For a scan schedule for which the number of frequency bands is a fixed number N and for which the dwell time and revisit time for each frequency band i is fixed, the capacity constraint is:

$$W = \sum_{i=1}^{N} \frac{\tau_d(i)}{\tau_{rv}(i)} \leq 1 \tag{3}$$

The left hand side of this equation is the *workload*. Another way to view the capacity constraint is to introduce the notion of the *duty cycle*; it is the fraction of the time allocated to this emitter. More precisely, it is the ratio of the dwell time to the revisit time for this emitter. The capacity constraint states that the total of all duty cycles can be no larger than 1.

While the capacity constraint is a necessary condition, it is not sufficient to ensure that a scan schedule exists. One limitation is that sensors require a small amount of time to switch from one frequency band to another. Another limitation that is more subtle is that the revisit times of different frequency bands may cause scheduling conflicts: two or more frequency bands may require the sensor at the same time. For this reason scheduling is an important part of the control of a sensor.

When an emitter has been detected, an attempt is made to identify the kind of emitter from a database of potential emitters. If this is successful, then the entry in the database is transferred to a table called the *Active Emitter Table* (AET). The AET is the primary output of the sensor, and it can also be used for scheduling the sensor.

If the emitter cannot be identified as a known kind of emitter, then the system creates a new AET entry as well as a new database entry. This usually requires that the emitter be detected more than once in order to obtain an estimate for the illumination period. Needless to say, such an unanticipated kind of emitter has important tactical and strategic consequences.

3 Fixed Scan Scheduling

The most commonly used sensor scheduling technique is the *Fixed Scan Scheduling* (FSS). A fixed scan schedule consists of a fixed sequence of CDWs for the entire mission of the platform. In this sequence, the CDWs for a specific emitter differ only in the start time t_{ds}, i.e., all the dwell times τ_d and the revisit times τ_{rv} for that emitter are the same.

Aside from the fact that the FSS is the most commonly used scheduling technique, it is important because it is usually the initial schedule for any other kind of scan scheduling. In the absence of any new information (i.e., in the absence of any emitter detections), an FSS has reasonably good performance, especially if the scenario is the same as was anticipated during the selection of a given schedule.

A FSS has the advantage that it is simple and thus easy to implement. However, it is based on a priori knowledge about the environment, so it is not applicable in unknown environments or environments that change. In particular, since it does not have a feedback mechanism, it cannot adapt to changes in the environment.

3.1 Constructing Fixed Scan Schedules

There are several ways to construct an FSS. A *rate monotonic fixed scan scheduler* (RM) [11, 12, 13, 14] is a general task scheduling technique that uses two pieces of data about tasks: task execution time and period. It is a priority-based scheduling technique which assigns the highest priority to the task with the shortest period. For sensors, the execution time is the dwell time and the period is the revisit time. If one assumes that the dwell time is equal to K pulse repetition intervals (where K is typically equal to 3), then the following equations describe the rate monotonic scan scheduler:

$$\tau_d = K \cdot T_{pri} \tag{4}$$

$$\tau_{rv} = T_{eit} - \tau_d \tag{5}$$

This assumes that the emitters emit pulses in a uniform pattern. It would not apply to emitters that use a non-uniform pattern, such as radars that employ a stagger pattern in which a non-periodic pattern of pulses is emitted repeatedly.

Figure 2 is an example of a rate-monotonic schedule. This schedule includes four emitters (E1, E2, E3, E4). Their characteristics (dwell time, illumination time) are given by: E1 : (0.5, 4), E2 : (0.3, 6), E3 : (0.8, 8) and E4 : (0.6, 12), respectively. Since RM scheduling is applied here, the priorities for these four emitters are 4, 3, 2 and 1.

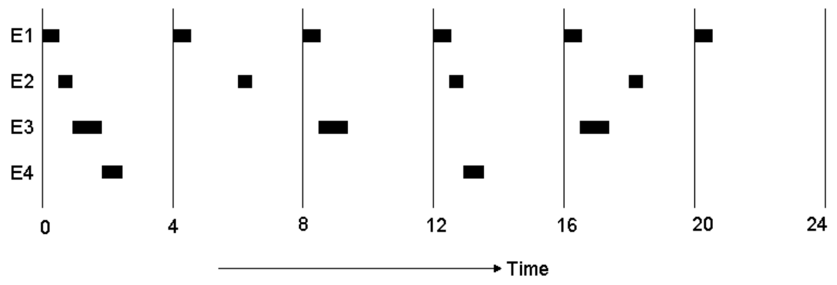

Figure 2: Example: A Fixed Scan Schedule

The computation of a rate-monotonic FSS starts by determining the least common multiple of the revisit times of the emitters. We call this the *scheduling cycle*. In the example of Figure 2, the revisit times are 4, 6, 8 and 12, so that the scheduling cycle is 24. In the next step, the CDW for the emitter having the highest priority is specified. This process is then repeated for the other emitters in order of priority until the scan schedule has been determined for the entire scheduling cycle. If the capacity constraint (Equation 3) is violated, then some revisit times must be increased using some rule, such as increasing the revisit time of lower priority emitters, until a scan schedule satisfies the capacity constraint and a schedule can be found. The schedule developed for one scheduling cycle is then repeated for all consecutive cycles.

3.2 Disadvantages of Fixed Scan Schedules

We now discuss the problems that arise when sensors are controlled using only an FSS. Consider, for example, a rate monotonic FSS and a simple scenario consisting of a single emitter and a single sensor having the following parameters:

$$T_{pri} = 2.0 \cdot 10^{-3} \text{ sec} \tag{6}$$

$$T_{pw} = 1.5 \cdot 10^{-5} \text{ sec} \tag{7}$$

$$T_{eip} = 4.0 \text{ sec} \tag{8}$$

$$B_e = 2^o \tag{9}$$

From these equations one can calculate the illumination time to be:

$$T_{eit} = 4.0 \text{ sec} \cdot (2^o/360^o) = 0.02 \text{ sec} \tag{10}$$

The dwell and revisit times for a rate monotonic fixed scan schedule will then be:

$$\tau_d = 3 \cdot T_{pri} = 6.0 \cdot 10^{-3} \text{ sec} \tag{11}$$

$$\tau_{rv} = T_{eit} - \tau_d = 0.0162 \text{ sec} \tag{12}$$

In a rate-monotonic FSS for this scenario, there are 246 CDWs for this emitter in each scheduling cycle. This corresponds to a duty cycle of

$$d = \tau_d/\tau_{rv} = 0.37 \tag{13}$$

for just this one emitter. The 246 receiver dwells corresponding to these CDWs will detect at most two illuminations of the emitter. Thus only two dwells will be effective; the other 244 will be wasted. When there are tens or even hundreds of emitters in the environment, this kind of scheduling will not be efficient. The goal is to reduce the number of CDWs required to detect each emitter, or, equivalently, to reduce the emitter duty cycles. Note, however, that we cannot lower the duty cycles by enlarging revisit times, since this might result in non-detections of emitter illuminations.

3.3 Introducing Dynamics to a Scheduler

One way to lower the number of dwells without missing (not detecting) illuminations is to synchronize the dwells with the illumination period of a given emitter. This is possible only after first detection of that emitter. However, in the dynamic environment we still are risking missing illumination due to dynamic effects. To compensate for the dynamics of the environment, a *dynamic scan schedule* (DSS) is used. Rather than being a fixed scheduling pattern, a dynamic scan schedule is an algorithm that dynamically modifies the scan schedule in response to detection events.

There are several ways that one can introduce dynamics into a scan schedule. One possibility is to use a precomputed FSS until first detection, and then recompute an FSS so that the newly detected emitter is only observed at the predicted illumination times as specified by the entry in the AET. After recomputation the CDWs for the emitter differ only in the start time t_{ds}, i.e., all the dwell times τ_d for the frequency band are the same. We will call this technique *informed dynamic scan scheduling.* The only difference between this technique and FSS is that detected emitters will have a much longer revisit time.

While informed dynamic scan schedule seems like it solves the problem of resource allocation in this case, it has some serious disadvantages. The main problem is that the illuminations of the aircraft will vary over time for a variety of reasons. The radar antenna, being a mechanical device, could speed up or slow down. However, a more significant effect is due to the motion of the aircraft. As the aircraft moves relative to the emitter, the apparent time between illuminations changes. For instance, consider an emitter with a 6 sec illumination period (i.e., the radar antenna rotates by one full cycle in 6 sec). Now suppose that the aircraft is 60,000 feet away from the radar and is moving in a circle around the radar at 1000 feet/sec, in a direction opposite to the rotation of the radar. In such a case while the radar rotates by 360 degrees, the aircraft moves by 6,000 feet, which is equivalent to about 6 degrees. Consequently, the apparent time between illuminations will be shorter by approximately 0.1 sec. This might seem like an insignificant shift, but since the shift is much higher than the illumination time, this shift is actually quite important. In fact, the illumination time is about 20% of the 0.1 sec shift, which is more than sufficient to cause a detection failure.

To address the problem of dynamic changes in the environment, we propose a feedback-control based DSS.

4 Mapping to a Control Architecture

We now show how the sensor scheduling problem can be mapped to a control based architecture. We first review some background in the control theory metaphor of software development. A specific mapping to a control theory architecture is then proposed.

4.1 Control Theory Metaphor for Software Engineering

In [3] we proposed a control theory metaphor for software development. The reason for proposing such an approach was to address the problem of changes in software requirements. In the case of embedded software, changes in software requirements come from changes in the environment with which the software interacts. As we discussed in Section 3.2, fixed scan schedules do not perform well when the environment changes dynamically or the workload is heavy. This is due to the high duty cycle required for each potential emitter and the relatively low probability of detecting each emitter. The control theory paradigm seems natural for this kind of application.

The main idea behind the control theory metaphor is to treat the basic software functionality as a Plant, to treat changes in the environment as disturbances, and to use feedback from the environment to adjust system behavior. In order to compensate for disturbances, a Controller must be added. From the point of view of software requirements, the Controller is a redundancy that is added to the system. The controller is not part of the original software requirements. Other kinds of (redundant) modules need to be added to implement a control loop, as described below.

In the control approach a system consists of two basic elements: a Plant (the system being controlled) and a Controller. To map a problem to a control architecture, it is sometimes necessary to add a Quality-of-Service (QoS) module that assesses the Plant's output in terms of a Quality-of-Service measure. (Note that in the control literature, QoS is not usually regarded as a separate module; it is simply a part of the controller.)

4.2 Plant

The whole system (the plant) for this problem includes a receiver (sensor), a sensor management module and an environment (a number of potential radar emitters). The receiver is traveling through the environment on a platform. The platform is usually an airplane.

The sensor management module is viewed, similarly as in [1], as consisting of three functions: *generate options*, *prioritize* and *schedule*. In our mapping, the first two functions are included in the module called *CDW Generator*. In the following, we specify all these modules.

4.2.1 Emitters and Receiver

The state of the platform and its receiver at a time t is determined by the position of the platform and by the frequency band to which the receiver is tuned. The position of the platform is determined by its cartesian coordinates $(x(t), y(t))$ over the terrain. The altitude of the platform is not significant for this problem domain. The trajectory of the platform can be arbitrary but it is fixed in a particular mission so it is modeled as the input process of the dynamic system. The position as well as the velocity $(x'(t), y'(t))$ of the platform can therefore be regarded as known.

The platform has a receiver that can be tuned to various frequency bands. It can only detect an illumination by an emitter if the receiver is tuned to a frequency band that includes the frequency of the emitter's signal. The state of the receiver can be represented by a finite discrete set of frequency bands, along with one point representing the state in which the receiver is not tuned to any band. The current frequency band is written ω, and the set of all such bands (including the "off" state) is written Ω. The state space for the platform is therefore the product of the (continuous) two-dimensional plane and the set of frequency bands: $R = \mathbb{R}^2 \times \Omega$.

We will assume that the emitters run independently. A state for a single emitter consists of the following three variables:

1. The direction ϕ in which the emitter is illuminating its environment. The variable ϕ is an angle, and therefore its set of values is isomorphic to the unit circle or to the

half-open interval $[0, 2\pi)$.

2. The distance r between the platform and the emitter.
3. The angle θ of the platform with respect to the emitter.

The state space for emitter i will be denoted $E(i)$. The variables r and θ represent the position of the platform with respect to the emitter in polar coordinates. It is sometimes convenient to use cartesian coordinates, in which case they will be written $(x_e(i,t), y_e(i,t))$.

Since the emitters were assumed to be independent, the overall state space for the emitters is the product of the state spaces of the individual emitters $\prod_i E(i)$. Combining this with the state space for the platform, the overall state space for the receiver and the emitters is:

$$Q = R \times \prod_i E(i) = \mathbb{R}^2 \times \Omega \times \prod_i E(i) \tag{14}$$

The evolution function of the system has both continuous and discrete components. The continuous component evolution is determined by the following stochastic differential equation for emitter i:

$$\frac{d}{dt}\phi(i,t) = 2\pi/T_{eip} + w_\phi(i) \tag{15}$$

$$\frac{d}{dt}x_e(i,t) = x'(t) + w_{x_e}(i) \tag{16}$$

$$\frac{d}{dt}y_e(i,t) = y'(t) + w_{y_e}(i) \tag{17}$$

In this system of equations, $w_\phi(i)$, $w_{x_e}(i)$ and $w_{y_e}(i)$ are the noise affecting the respective derivative. These are assumed to be Gaussian white noise (with mean 0). It is also assumed that $w_\phi(i)$ is independent of the other two noise variables.

The discrete component of the evolution of the receiver is characterized by discontinuous "jumps" from one state to another. The receiver evolution is determined by the scan schedule. Each CDW that is selected by the scheduler generates an event, which in turn, causes a discontinuous jump from one frequency band, ω_i, to another, ω_j, at the time of the event.

The last component of this dynamic system is the output function. This function determines what function of the variables of Q are observed as the system evolves. As is typical of such systems, the objective is to infer the values of the state variables from what can be observed. The output function computes *detections*. Whenever the receiver satisfies the requirements for a detection, the function gives the data associated with the detection.

For each emitter there is an upper limit r_m on the distance at which it can be detected. When the platform is beyond the range of the emitter, then no detection is possible at any angle. The same would be the case if the emitter were not emitting either because it was not turned on or because the emitter was not deployed. For the purpose of this problem, it is not possible to distinguish an out of range emitter from one that is off.

The following are the requirements for detecting emitter i: the emitter must be operating and within range, the emitter must be illuminating the receiver, and the emitter frequency must be in the current receiver frequency band. The requirements above must hold for a period of time at least equal to τ_e. Putting the conditions above together we can formally define a detection as follows:

Definition 4.1 *Consider a dwell commanded by the* $CDW = (t_{ds}, \tau_d, \omega)$. *The time* t_d *is called the* time of detection *of emitter i, if there exists an interval* $(t_0, t_d) \subset (t_{ds}, t_{ds} + \tau_d)$ *such that for every* $t \in (t_0, t_d)$

1. $r(i,t) < r_m(i,t)$,
2. $|\theta(i,t) - \phi(i,t)| < B_e(i)$,
3. $f_e(i) \in \omega(t)$, *and*
4. $t_d - t_0 \geq \tau_e$.

The receiver actually measures the amplitude of the illumination continuously during the dwell time, and it determines that an illumination has occurred by comparing the measured signal with a noise threshold. For this reason the measurement noise will not be considered in the analysis of the model presented in this paper. A more elaborate model would, of course, consider the measurement noise in addition to the process noise.

When the receiver dwells on a specific emitter and does not detect that emitter, the output function returns $t_l(i)$, i.e., the *time of last look* at that emitter. The output function produces the set of detection times and look times for all emitters that the receiver looked as directed by the scheduler. In other words:

Definition 4.2 *The output function* $g : T \times I \to T \times \{l, d\}$ *is a function of time and frequency bands s.t.,* $g(t,i) = (t_d, d)$ *if target i was detected during last dwell and* (t_l, l)*, if target i was not detected during last dwell.*

For simplicity, we will say that the value of the function is either t_d or t_l. Since the value of this function will not change between dwells, the set of detection times will be denoted as T_d and the set of non-detections will be denoted as T_l.

Note that this definition assumes that when more than one emitter is transmitting on the same frequency band, the receiver can distinguish them from each other. In other words, it can determine which of the potential emitters in a band is actually illuminating the receiver. Because emitters differ from one another in the detailed structure of the illumination (i.e., the pulse pattern), this is not an unreasonable assumption, although it might not always be correct. If such discrimination is not possible, then the output function must be redefined so that it only computes the frequency band that is being detected, not the specific set of emitters.

Since a detection can only occur during a dwell, one simplification of the output function is for the output function to produce a sequence of data structures, one for each CDW. This is, in fact, what many receivers produce. The output is a data structure called a *pulse descriptor word* (PDW) which contains detailed information about the illumination(s) that were detected during the period of time covered by the CDW.

4.2.2 CDW Generator

Another part of the plant is CDW Generator. Two parts of the CDW (cf. Equation 1) are fixed for each emitter: τ_d and ω. The time of the start of the next dwell, $t_{ds}(t + \Delta t)$, is re-computed at the time of the assumed beginning of the current dwell, $t_{ds}(t)$. Since the CDW Generator shifts starts of dwells, this module also introduces additional dynamics to the system.

If the emitter illumination was perfectly periodic, if there were no delays, if both the platform and the emitter were stationary, and if we knew the distance between the emitter and the platform then we would be able to predict exactly the time of the next dwell as the time of the current dwell, $t_{ds}(t)$ (assuming the receiver dwells on each illumination), plus the revisit time for that receiver, τ_{rv}. In order to compensate for delays, we need to adjust τ_{rv}. We use the following model to achieve this goal.

$$\tau_{rv}(i, t + \Delta t) = \begin{cases} \tau_{rv}(i, t), & \text{if } \; t < t_{ds}(i, t) + \tau_{rv}(i, t) \\ \bar{\delta}(i, t)^2 \cdot T_{eip}(i), & \text{otherwise} \end{cases} \tag{18}$$

$$t_{ds}(i, t + \Delta t) = \begin{cases} t_{ds}(i, t), & \text{if } \; t < t_{ds}(i, t) + \tau_{rv}(i, t) \\ t + \Delta t, & \text{otherwise} \end{cases} \tag{19}$$

$$\bar{\delta}(i, t) = \begin{cases} 1, & \text{if } \; \delta(i, t) > 1 \\ \Delta(i), & \text{if } \; \delta(i, t) < \Delta(i) \\ \delta(i, t), & \text{otherwise} \end{cases} \tag{20}$$

$$\Delta(i) = (T_{eit}(i)/T_{eip}(i))^{0.5} \tag{21}$$

In these equations δ is the input from the Emitter-Level Controller (see Section 4.4). The value of $\Delta(i)$ determines the smallest revisit time. If the system always used the revisit time determined by $\Delta(i)$, it would be equivalent to a FSS. The time increment Δt used in our simulations was $10\mu sec$.

4.2.3 Scheduler

The scheduler is responsible for adjusting the workload and selecting CDWs to be executed by the receiver. The CDWs generated by the CDW Generators are fed into the scheduler's queue. However, since the CDWs are generated by a number of independent generators (one per emitter), there may be too many CDWs for this receiver to execute. The limit on the capacity of the receiver is defined by the instant workload; it is violated (i.e, the scheduler is overloaded [15]) if the value of the workload is higher than one (see Equation 3). In other words, the set of tasks violates the *schedulability constraint* [16]. The instant workload is defined as the sum of the duty cycles of the CDWs in the queue:

$$W(t) = \sum_{i \in \{CDW\}} \frac{\tau_d(i)}{\tau_{rv}(i,t)} \tag{22}$$

where $\{CDW\}$ denotes the set of CDWs in the queue. The scheduler adjusts the workload by adjusting the revisit time for each CDW in the queue as follows:

$$\tau'_{rv}(i,t) = \tau_{rv}(i,t)/c(t) \tag{23}$$

where $c(t)$ is the *coverage* parameter computed by the sensor-level controller.

We use a priority-based scheduler, where priorities are computed dynamically. The scheduler is invoked at *schedule time*, t_s. The schedule time is defined as the completion time of the current dwell, i.e., $t_{ds}(t-1) + \tau_d(t-1)$ or the beginning of the next dwell in the queue.

A scheduling decision $\omega(t_s)$ is made based upon the current value of the QoS (which will be defined in Section 4.3), where a specific frequency band j is chosen if the QoS for it is the largest.

$$\omega(t_s) = j, \tag{24}$$

$$QoS(j, t_s) = \max_i QoS(i, t_s) \tag{25}$$

4.3 The Quality of Service Metric

The QoS is a metric that determines the level of quality exhibited by the system at any point in time. It can be used to specify the required level of service of the system and to compare the performance of different algorithms. As noted in the Section 4.2 above, the purpose of the system is to infer the state from the limited information in the output function. The sensor system is continually computing its current best estimate of the state variables. Obviously, it cannot know the state variables exactly, and not all state variables are equally important. In many situations, the most important fact about an emitter is that it is emitting within range of the platform, while in other cases this fact is less important than the location of the emitter relative to the platform. The first situation is known as the *detection problem* while the second one is usually called the *tracking problem*, although a more accurate description of the second problem is the *location problem* since the emitter is not moving.

Because the problem to be solved is the estimation of the state variables, the QoS function should measure the accuracy of the estimates, such that the more important state variables are given more weight in the metric than the less important ones. Of course, the actual accuracy is not known, but it is possible to estimate it. Since the dynamic system for each emitter is a linear stochastic dynamic system (cf. Section 4.2.1), the overall measure of accuracy is the (co)variance of the error estimates. The dynamic system is 3-dimensional so the covariance matrix is a positive definite matrix of order 3. For emitter i, this matrix is block diagonal with two blocks as follows:

1. *Position uncertainty.* The variance of the (error of the) position estimate is due to the motion of the emitter as well as the uncertainty in the position of the platform. The motion of the emitter and other effects not accounted for by the differential equations (16,17) are represented by the noise terms $w_{x_e}(i)$ and $w_{y_e}(i)$. Although the variance of these noise terms can, in general, vary with time, we will assume that they are time-independent and that these two noise terms are uncorrelated. It follows that the variance of the position estimate has the form $\sigma_p^2(i,t)I$, where I is the unit matrix. It also follows that $\sigma_p^2(i,t)$ increases linearly with time, if the emitter is not being observed.

2. *Beam direction uncertainty.* The rotation of the emitter is a mechanical system so it is subject to mechanical noise. The noise term is written $w_\phi(i)$ in Equation 15. We assume that $w_\phi(i)$ is independent of $w_\phi(j)$ for $j \neq i$. We also assume that the variance of $w_\phi(i)$ is time-independent. It follows that the variance $\sigma_b^2(i,t)$ of the (error of the) beam direction estimate increases linearly with time, if the emitter is not being observed.

The QoS metric for each emitter is defined by linearly combining the position and beam direction variances: $QoS(i,t) = a_i\sigma_p^2(i,t) + b_i\sigma_b^2(i,t)$. The constants a_i and b_i determine the importance of emitter i relative to other emitters. These constants also determine whether the emphasis for emitter i is on detecting the emitter or on locating the emitter. For the rest of this paper, we will emphasize detection of emitters. As a result we will set $a_i = 0$ for every emitter i. We call $QoS(i,t)$ the *emitter-level* QoS metric.

When the receiver is tuned to a frequency band containing the frequency of emitter i, then the emitter is being measured. The measurement is extended over a period of time (the dwell time), and the output of the receiver is a continuous output over this period. The measurement is therefore a continuous-time stochastic dynamic system just like the platform/emitter dynamic system. Unlike the dynamic system describing the platform and emitters, which is a linear dynamic system, the measurement process is nonlinear. As a result of the nonlinearity, the gain due to the measurement varies with the beam direction as well as the distance from the emitter. For example, when the emitter is illuminating the platform, the gain will be much higher than when the emitter is not.

One can summarize these observations as follows:

1. When emitter i is not being observed, its emitter-level QoS increases linearly with time.
2. When emitter i is being observed, its emitter-level QoS decreases with time, but the decrease depends on the result of the measurement.

A similar model for sensor management was introduced in [17], but in that model the measurement process was assumed to be linear. As a result, the reduction in the variance was a linear function of time during a measurement.

As discussed in Section 4.2.1, we assume that the receiver measurement is processed by the receiver to produce a discrete two-valued response rather than a continuous amplitude response. This simplifies the effect of a measurement on $QoS(i,t)$ to just two cases: a large gain when the receiver detects an illumination and a small gain when the receiver does not detect an illumination.

To obtain an explicit formula for the emitter-level QoS metric, we make some assumptions and approximations. We first assume that the beam direction noise variance is inversely proportional to the rotation period. In other words, we presume that the error accumulates at the same rate *per rotation* for every emitter. Therefore, if one is not observing the emitter, its emitter-level QoS increases at the rate $\frac{b_i k}{T_{eip}(i)}$, for some constant k. We write $P(i)$ for $b_i k$.

We next assume that when the emitter is detected, its beam direction variance drops to 0. This is only approximately correct, since a detection only determines the beam direction within an angular segment determined by the beam width.

The last case to consider is an observation of the emitter that does not detect it. In this case one has eliminated the possibility that the beam direction is in the interval covered by the beam width. This assumes that the dwell time is the minimum necessary for a detection. A longer dwell would reduce the variance by more than the beam width. Under these assumptions, an observation reduces the beam direction variance by an amount approximately proportional to the ratio of the beam width to the rotation period. For simplicity we assume the same constant of proportionality k and importance constant b_i as above. Under these assumptions, the net effect of such an observation on the emitter-level QoS is to reduce it by $P(i) \cdot \frac{T_{eit}(i)}{T_{eip}(i)}$.

We now summarize this analysis with the following formula for the emitter-level QoS metric:

$$QoS(i,t) = \frac{P(i)}{T_{eip}(i)} \cdot (t - t_d(i)) + d(i, t_l) \tag{26}$$

where

$$d(i,t_l) = \begin{cases} P(i) & : \quad t_l = StartTime \\ 0 & : \quad t_l \in T_d(i) \backslash StartTime \\ d(i, t_l^-) - P(i) \cdot \frac{T_{eit}(i)}{T_{eip}(i)} & : \quad t_l \in T_l(i) \backslash T_d(i) \end{cases} \tag{27}$$

In this formula, $t_d(i)$ denotes the time of last detection of emitter i, and $P(i)$ denotes the weight associated with this emitter. $P(i)$ is proportional to the importance of the emitter to the EW mission; it may also be thought of as priority of the emitter. In addition, $T_d(i)$ denotes the set of detection times for this emitter at the start time, and $T_l(i)$ denotes the set of observation times for this emitter. We consider the observation time to be the time at the end of the dwell, so $T_d(i)$ is a subset of $T_l(i)$. Note that the function $d(i, t_l)$ is updated only at observation times.

The emitter-level QoS metrics can be combined to form the overall QoS metric (called the *sensor-level* QoS). The combination is usually either the maximum (or minimum) of the individual QoS metrics, or the (weighted) average of the individual QoS metrics.

Other researchers, cf. [18, 19, 4, 6], used an entropy based metric derived from information theory. Sensor allocation is then formulated as a linear programming problem. While this approach has the advantage of being generic and is well based in the theory of information, the linear programming formulation does not address the dynamics of this scenario. This scenario is closer to a dynamic programming formulation [20], rather than linear programming. The real-time systems community uses the term “QoS” in a somewhat different way. The QoS there means “the level of service”, i.e., the percentage of CPU time that is assigned to a particular task (cf. [8, 21, 22]).

4.4 Controllers

As shown in Figure 3, the system includes two controllers - an Emitter-Level Controller and a Sensor-Level Controller. Other architectures are known in the subject literature (cf. [23, 24]), however we had to choose a different architecture due to the need to include compensation for dynamic changes in the environment.

4.4.1 Emitter-Level Controller

There is one Emitter-Level Controller for each CDW Generator, i.e., one per emitter. The goal of this controller is to compensate for the delays due to both platform movement and scheduling. In the system described in this paper we simply used a proportional controller, with the proportional parameter denoted as $K(i)$. The control output $\delta(i,t)$ generated by this controller is a parameter in the CDW Generator module. The reference input for the Emitter-Level Controller should be set to an expected level of uncertainty. Since we have different priorities for particular emitters, we set the reference input to the value of priority of a given emitter, $P(i) \in [0,1]$.

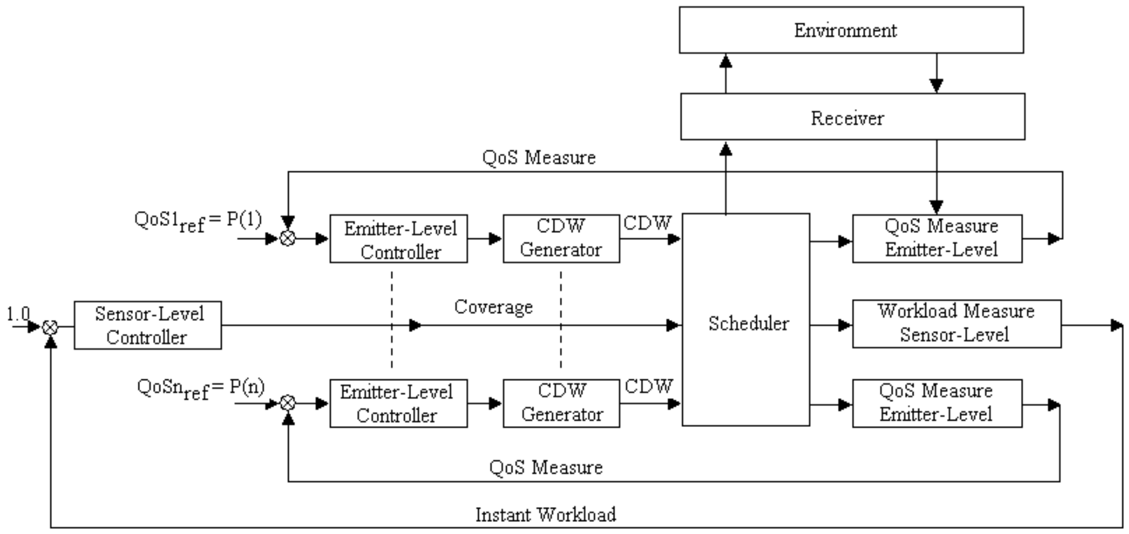

Figure 3: Control Based Sensor Management Architecture

4.4.2 Sensor-Level Controller

The goal of the Sensor-Level Controller is to compensate for changes in the demand for the workload (overload) of the receiver. In other words, the goal of this controller is to satisfy the constraint of schedulability [16] when the scheduler is overloaded (similarly as in [15]). The QoS for the Sensor-Level Controller is the overall instant workload defined by Equation 22. The reference input for the Sensor-Level Controller is set to 1. It is the maximal capacity for scheduling. We used a PID controller for sensor-level control. The proportional, integral and differential parameters for the PID controller are denoted as K_p, K_i and K_d, respectively. The output of this controller is denoted as $c(t) \in [0,1]$.

The idea of controlling the demand for a scheduler is also known in the real-time scheduling community [25, 26, 27].

4.4.3 Emitter-Level Controller: Design Issues

In this section we analyze the behavior of a scheduling system under various approaches to scheduling. First we discuss the FSS and then compare its behavior with a DSS. This analysis is intended to provide guidelines for designing controllers, i.e., selecting the control parameters K_p, K_i and K_d. First we focus on the FSS. As we discussed in Section 3, the FSS dwells follow a fixed dwell pattern. The pattern is designed in such a way that the dwells are very frequent, only one emitter illumination time apart. An example of a FSS dwell pattern is shown in Figure 4 (top). We can see from this Figure that only one or two of the 180 CDWs result in a detection. This is very inefficient.

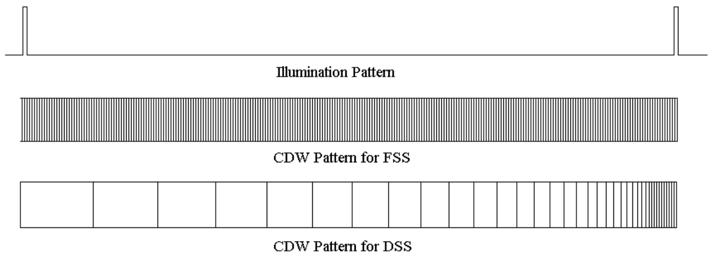

Figure 4: CDW Patterns for FSS and DSS

A pattern for a DSS is shown in the bottom part of Figure 4. This pattern is described by the model of the CDW Generator (Eqs. 18 - 20) and by the Emitter-Level Controller. This pattern shows dwells generated by the CDW Generator after the first detection of an illumination. The time span is equal to the time between two consecutive illuminations.

The question is whether such a dynamic CDW pattern will detect every illumination in spite of phase shifts due to the motion of the platform (as well as the emitter). In Section 3.3 we showed an example in which the shift in apparent time between illuminations was approximately 0.1 sec, which is about 2% of the receiver illumination period. This is equal to three illumination times and thus it is not negligible. In order to detect an illumination, the revisit time within the vicinity of the illumination must be less than or equal to the value of illumination time minus dwell, which in this example is about 0.5% of the illumination period. Consequently, when $t_{ds}(i,t) - t_d(i) > 0.98 \cdot T_{eip}(i)$ the condition that guarantees detection is:

$$\tau_{rv}(i) < 0.005 \cdot T_{eip}(i) \tag{28}$$

The dwell pattern depends on the dynamics of the plant and on the controller's parameter $K(i)$. In Table 1 we show how many CDWs are needed for one illumination period depending on the value of $K(i)$ and on the dynamics of the system (amount of shift in the illumination pattern).

$K(i)$	$\frac{t_{ds}(i,t)-t_d(i)}{T_{eip}(i)}$	Number of CDWs
0.08	0.1191	198
0.4	0.8268	63
0.6	0.8851	42
0.8	0.9125	31
0.98	0.9604	9

Table 1: Number of CDWs for Different Controllers and Dynamics

If we knew the distance to a given emitter, provided that we know the illumination period of the emitter and the speed of the platform motion, we should be able to pick an appropriate $K(i)$, i.e., such that guarantees detection while not overloading the receiver with too many requests for dwells (number of CDWs). However, if this kind of knowledge is not available, we need to select a more conservative approach. For an emitter with a long illumination period we can choose a controller with a large $K(i)$ resulting in fewer CDWs. For a system with a shorter illumination period, we can choose a controller with a small $K(i)$, which can generate more CDWs but will give a higher probability of detection.

4.4.4 Sensor-Level Controller: Design Issues

The discrete representation of the PID controller is given by the following equation.

$$C(Z) = K \cdot \frac{Z^2 + a_1 \cdot Z + a_2}{Z^2 - Z} \tag{29}$$

where

$$K = K_p \cdot (1 + K_i + K_d) \tag{30}$$

$$a_1 = -\frac{1 + 2K_d}{1 + K_i + K_d} \tag{31}$$

$$a_2 = \frac{K_d}{1 + K_i + K_d} \tag{32}$$

The plant model can be approximated as:

$$P(Z) = \frac{1}{z - 1} \tag{33}$$

The close-loop discrete transfer function of the whole control system is:

$$H(Z) = \frac{K \cdot (Z^2 + a_1 \cdot Z + a_2)}{Z^3 + Z^2 \cdot (K-2) + Z \cdot (1 + K \cdot a_1) + K \cdot a_2} \tag{34}$$

Based upon this model, we can design a system with poles and zeros at $z1 = 0.91, p1 = 0.885, p2 = 0.565$, i.e., such that the poles and zeros are located within the unit circle, which results in a stable system. For instance, using Routh-Hurwitz method we can find out that the PID parameters $K_i = 0.5$, $K_i = 0.1$ and $K_d = 0.0$.

5 Simulation Results

We simulated a scenario in which a platform was moving along the straight line shown in Figure 5. The speed of the platform was 1000 miles/hour. It took the platform 230 sec to travel the path.

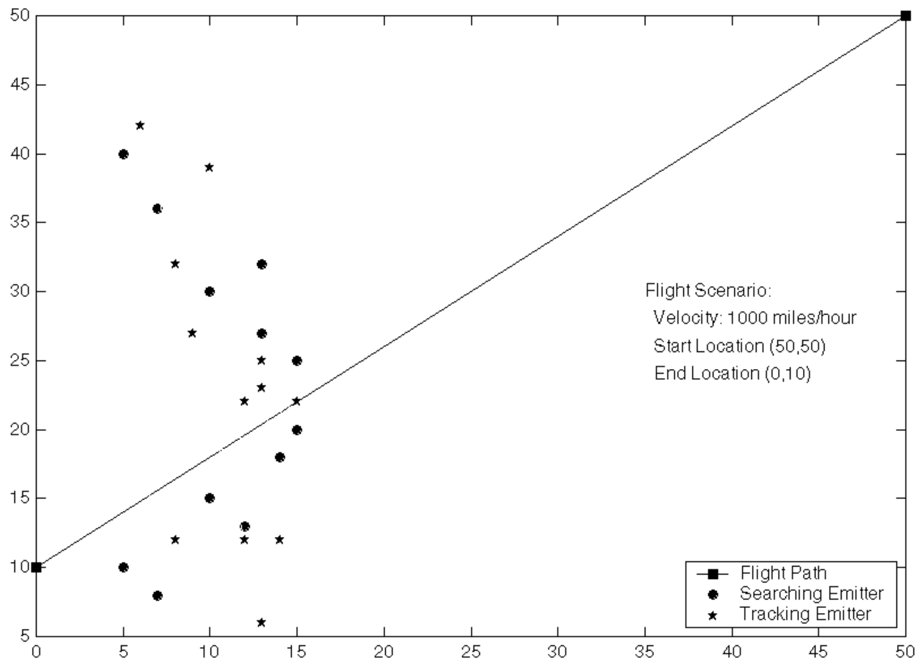

Figure 5: Simulation Scenario

There were twenty four emitters in the scenario. The characteristics of the emitters are shown in Table 2. We simulated both light and heavy load scenarios. Since for a light load the performance of a dynamic scan scheduler is roughly the same as for a FSS, we focused on heavy workloads.

Emitter ID	f_e GHz	T_{pri} sec	T_{pw} sec	T_{eip} sec	B_e $degree$	x, y $[mile, mile]$
1	2.25	$2.0 \cdot 10^{-3}$	$4.5 \cdot 10^{-5}$	10.0	4.0	[5.00, 10.0]
2	3.25	$3.0 \cdot 10^{-3}$	$3.5 \cdot 10^{-5}$	10.0	4.0	[7.00, 8.00]
3	5.25	$4.0 \cdot 10^{-3}$	$2.5 \cdot 10^{-5}$	10.0	4.0	[5.00, 40.0]
4	4.25	$3.5 \cdot 10^{-3}$	$1.3 \cdot 10^{-5}$	15.0	3.0	[7.00, 36.0]
5	6.25	$2.5 \cdot 10^{-3}$	$2.6 \cdot 10^{-5}$	15.0	3.0	[10.0, 30.0]
6	8.25	$1.5 \cdot 10^{-3}$	$3.9 \cdot 10^{-5}$	15.0	3.0	[13.0, 32.0]
7	7.75	$2.5 \cdot 10^{-3}$	$2.0 \cdot 10^{-5}$	18.0	3.5	[10.0, 15.0]
8	4.75	$3.5 \cdot 10^{-3}$	$3.0 \cdot 10^{-5}$	18.0	3.5	[12.0, 13.0]
9	5.75	$4.5 \cdot 10^{-3}$	$4.0 \cdot 10^{-5}$	18.0	3.5	[15.0, 25.0]
10	3.75	$0.8 \cdot 10^{-3}$	$4.0 \cdot 10^{-5}$	12.0	3.0	[13.0, 27.0]
11	2.75	$1.6 \cdot 10^{-3}$	$3.0 \cdot 10^{-5}$	12.0	3.0	[15.0, 20.0]
12	6.75	$2.4 \cdot 10^{-3}$	$2.0 \cdot 10^{-5}$	12.0	3.0	[14.0, 18.0]
13	11.25	$2.0 \cdot 10^{-4}$	$4.5 \cdot 10^{-6}$	5.0	2.0	[8.00, 12.0]
14	13.25	$3.0 \cdot 10^{-4}$	$3.5 \cdot 10^{-6}$	5.0	2.0	[13.0, 6.00]
15	15.25	$4.0 \cdot 10^{-4}$	$2.5 \cdot 10^{-6}$	5.0	2.0	[10.0, 39.0]
16	17.25	$3.5 \cdot 10^{-4}$	$1.3 \cdot 10^{-6}$	4.0	2.0	[6.00, 42.0]
17	16.25	$2.5 \cdot 10^{-4}$	$2.6 \cdot 10^{-6}$	4.0	2.0	[9.00, 27.0]
18	14.25	$1.5 \cdot 10^{-4}$	$3.9 \cdot 10^{-6}$	4.0	2.0	[8.00, 32.0]
19	12.25	$2.5 \cdot 10^{-4}$	$2.0 \cdot 10^{-6}$	6.00	2.5	[12.0, 12.0]
20	14.75	$3.5 \cdot 10^{-4}$	$3.0 \cdot 10^{-6}$	6.00	2.5	[14.0, 12.0]
21	16.75	$4.5 \cdot 10^{-4}$	$4.0 \cdot 10^{-6}$	6.00	2.5	[13.0, 25.0]
22	13.75	$0.8 \cdot 10^{-4}$	$4.0 \cdot 10^{-6}$	3.0	2.0	[12.0, 22.0]
23	15.75	$1.6 \cdot 10^{-4}$	$3.0 \cdot 10^{-6}$	3.0	2.0	[15.0, 22.0]
24	11.75	$2.4 \cdot 10^{-4}$	$2.0 \cdot 10^{-6}$	3.0	2.0	[13.0, 23.0]

Table 2: Characteristics of Simulated Emitters

The strength of the signal at the sensor was changing according to the following equation:

$$A_r = \frac{A_e \cdot e^{-(\theta-\phi)^2/{B_e}^2}}{r^2} \tag{35}$$

where A_r and A_e are the signal strengths at the sensor and emitter, respectively, ϕ and θ are the angle of the beam line of the emitter and the angle between the sensor and the emitter, respectively, B_e is the beam width, and r is the distance between the sensor and the emitter. The illumination was detected if the strength A_r of the signal at the emitter exceeded the threshold Th_r of the sensor.

In Figure 6 we show the distribution of the QoS for a FSS and for a DSS. The workload for this scenario was 2.7967. From this figure, we can see that the values of the QoS for the DSS is much lower than for the FSS. Since the QoS represents the uncertainty about the state of an emitter, the lower QoS is better than the higher one.

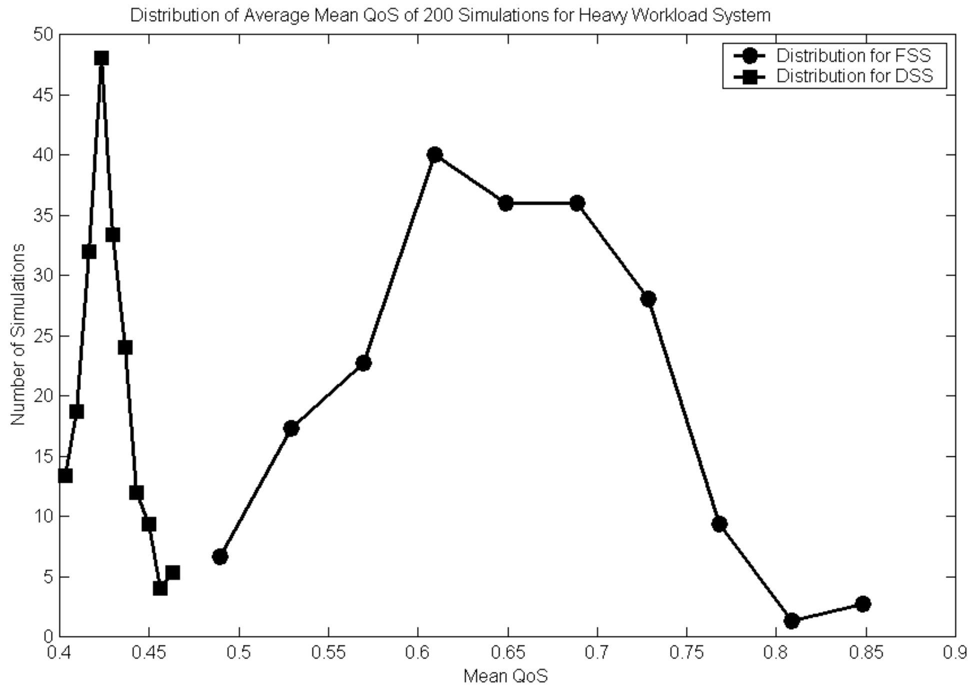

Figure 6: Distribution of Mean QoS; Workload = 2.7967

In Figure 7 we show the distribution of the average maximal value of the QoS. From this figure we can see that the system responds to the controllers very well. The maximal value of the QoS does not cross the reference value of $P(i)$, which in this experiment was set to 1 for all emitters. The performance of the DSS is much better that of the FSS. The FSS's maximal QoS shows a tendency of growing. The maximal QoS is an indication of the stability of the system. Roughly, the system is stable if the value of the QoS is bounded. Figure 7 shows that in the simulations the DSS showed a stable behavior.

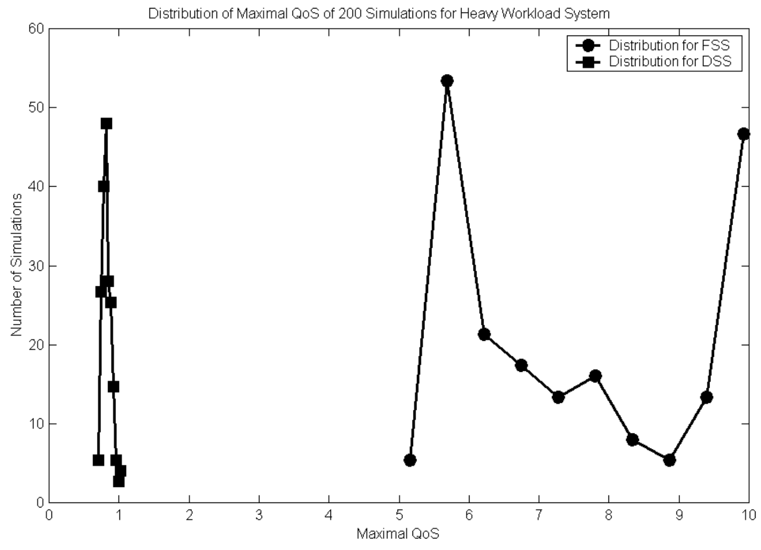

Figure 7: Distribution of Max QoS

6 Conclusions and Future Research

From the results of our simulations we can see that the performance of the control based DSS is dramatically better than that of the FSS in both accuracy and the number of CDWs especially when system is overloaded. The QoS for the control based DSS is much lower and, at the same time, the number of CDWs is reduced to only one tenth of the CDWs in the FSS. This is understandable since our DSS does not assume that the illumination period is constant. Instead, it settles on a fixed period only through the application of its control law. This gives it the robustness that is especially important when the environment undergoes dynamic changes.

Although by using the control theory approach we achieved great improvement over the fixed scan scheduler, we still have room for optimization of this solution. For instance, we are implementing one more control loop to directly control the time spent for the computation of schedules. This is a very important issue in our application since the time scale of the scheduling is comparable to the time scale of computation. And thus the issue of complexity of computation needs to be addressed. Furthermore, we need to consider a more complex scenario in which there are multiple sensors as well as multiple emitters. This leads naturally to a distributed control architecture.

As we mentioned briefly at the end of Section 4.1, we also plan to implement the full control architecture proposed in [3]. This requires implementing two additional control loops: the adaptation loop and the reconfiguration loop. These control loops operate at longer time scales than the feedback control loop. The adaptation loop modifies the Controller when the environment changes in more dramatic ways than simply the detection of a new emitter, such as a missile attack or the detection of an unexpected kind of emitter. The reconfiguration loop modifies the software system by replacing and/or changing the connections of components. This control loop operates at an even longer time scale than the adaptation loop. Reconfiguration is a response to changes in the environment that cannot be satisfied by simply modifying the Controller, such as changes in battle strategies.

Still another problem that needs to be addressed is the development of rules for mapping applications to the control theory based architectures. The mapping is currently being performed in an ad hoc manner based on experience. A more systematic approach would make it much easier for the control theory metaphor to be applied by the general software community. We made some advancements in this direction in [17].

Acknowledgments

This research was partially supported by the Defense Advanced Research Projects Agency under contract F330602-99-C-0167. We are grateful to many people for helpful conversations and suggestions, most notably Kevin Passino, Mark MacBeth, Paul Zemany, Artan Simeqi and Fengming Zhang.

References

[1] S. Musick and R. Malhotra. Chasing the elusive sensor manager. In *Proceedings of the IEEE 1994 NAECON, Vol. 1*, pages 606–613, 1994.

[2] L. Y. Pao and M. Kalandros. The effects of delayed sensor requests on sensor manager systems. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, pages 1127–1135, 1998.

[3] M. M. Kokar, K. Baclawski, and Y. Eracar. Control theory-based foundations of self-controlling software. *IEEE Intelligent Systems*, pages 37–45, May/June 1999.

[4] J. Llinas, K. J. Hintz, and G. O. Beale. Applications of automatic control theory to sensor scheduling. Technical Report NAWCADWAR-93002-50, Naval Warfare Center, Aircraft Division, Warminster, PA, 1992.

[5] J. Nash. Optimal allocation of tracking resources. In *Proceedings of the 1977 Conference on Decision and Control, Vol. 1*, pages 1177–1180, 1977.

[6] W. Schmaedeke. Information based sensor management. In *Proceedings of the SPIE Conference on Target Recognition and Data Fusion, Vol. 1955*, pages 156–164, 1993.

[7] L. Palopoli, L. Abeni, F. Conticelli, M. D. Natale, and G. Buttazzo. Real-time control system analysis: An integrated approach. In *IEEE Real-Time Systems Symposium*, pages 131–140, Orlando, FL, 2000.

[8] C. Lu, J. A. Stankovic, T. F. Abdelzaher, G. Tao, S. H. Son, and M. Marley. Performance specifications and metrics for adaptive real-time systems. In *IEEE Real-Time Systems Symposium*, pages 13–23, Orlando, FL, 2000.

[9] D. C. Steere and et. al. A feedback-driven proportion allocator for real-rate scheduling. In *Symposium on Operating Systems Design and Implementation*, pages 145–158, 1999.

[10] C. Lu, J. A. Stankovic, G. Tao, and S. H. Son. Design and evaluation of a feedback control edf scheduling algorithm. In *IEEE Real-Time Systems Symposium*, pages 56–67, Phoenix, AZ, 1999.

[11] J. W. S. Liu. *Real-Time Systems*. Prentice Hall, 2000.

[12] M. Klein, T. Ralya, B. Pollak, R. Obenza, and M. G. Harbour. *A Practitioner's Handbook for Real-Time Analysis: Guide to Rate Monotonic Analysis for Real-Time Systems*. Kluwer Academic Publishers, 1993.

[13] M. G. Harbour, M. H. Klien, and J. P. Lehoczky. Fixed priority scheduling of periodic tasks with varying execution priorities. *Proceedings of IEEE Real-Time Systems Symposium*, pages 116–128, 1991.

[14] J. P. Lehoczky, L. Sha, and Y. Ding. The rate monotonic scheduling algorithm - exact characterization and average case behavior. In *IEEE Real-Time Systems Symposium*, pages 166–171, 1989.

[15] P. Mejia-Alvarez, R. Melhem, and D. Mosse. An incremental approach to scheduling during overloads in real-time systems. In *IEEE Real-Time Systems Symposium*, pages 283–294, 2000.

[16] D. Seto, J. P. Lehoczky, L. Sha, and K. G. Shin. On task schedulability in real-time control systems. In *IEEE Real-Time Systems Symposium*, pages 13–21, 1996.

[17] M. M. Kokar, K. M. Passino, K. Baclawski, and J. E. Smith. Mapping an application to a control architecture: Specification of the problem. *Lecture Notes in Computer Science*, 1936:75–89, 2001.

[18] K. J. Hintz. A measure of the information gain attributable to cueing. *IEEE Transactions on Systems, Man and Cybernetics*, 21, No. 2:237–244, 1991.

[19] K. J. Hintz and E. S. McVey. Multi-process constrained estimation. *IEEE Transactions on Systems, Man and Cybernetics*, 21, No. 1:434–442, 1991.

[20] D. A. Castanon. Approximate dynamic programming for sensor management. In *Proceedings, IEEE Conference on Decision and Control*, pages –, 1997.

[21] R. Rajkumar, C. Lee, J. Lehoczky, and D. Siewiorek. Practical solutions for QoS-based resource allocation problems. In *IEEE Real-Time Systems Symposium*, pages 296–306, 1998.

[22] D. Hull, A. Shankar, K. Nahrstedt, and J. W. S. Liu. An end-to-end QoS model and management architecture. In *IEEE Real-Time Applications Symposium*, pages 176–189, 1997.

[23] C. Aurrecoechea, A Cambell, and L. Hauw. A survey of QoS architectures. In *4th IFIP International Conference on Quality of Service*, pages 138–151, 2001.

[24] S. G. Bier and P. Rothman. Intelligent sensor management for beyond visual range air-to-air combat. In *Proceedings of the IEEE 1988 NAECON*, pages 264–269, 1988.

[25] G. Buttazzo, G. Lipari, and G. Abeni. Elastic task model for adaptive rate control. In *IEEE Real-Time Systems Symposium*, pages 286–295, 1998.

[26] D. Rosu, K. Schwan, and S. Yalamanchili. FARA - a framework for adaptive resource allocation in complex real-time systems. In *IEEE Real-Time Technology and Applications Symposium*, pages 79–84, 1998.

[27] D. Rosu, K. Schwan, S. Yalamanchili, and R. Jha. On adaptive resource allocation for complex real-time applications. In *IEEE Real-Time Systems Symposium*, pages 320–329, 1997.