Performance Metric Alerting: A New Design Approach for Complex Alerting Problems

Lee C. Yang and James K. Kuchar

***Abstract*—Alerting systems and related decision-making automation are widely used to enhance the safety and capability of controlled processes across many applications. Traditional alerting systems use physical metrics such as temperature, distance between vehicles, or time-to-impact as bases for making alerting decisions. Threshold values on these metrics are typically derived using an iterative process to ensure the achievement of desired performance goals, defined by higher-level metrics such as false alarm, safety, or success probabilities. We generalize this problem and develop two state spaces, one representing physical metrics and one representing performance metrics. A traditional alerting system operates completely within the physical space, using decision thresholds that have been developed offline during the design process by examining how the physical threshold translates across to the performance state space. The physical metrics thus act as an indirect means to control the performance of the system. We propose an alternate approach that enables the system to operate online in the performance space. Alerting decisions are based directly on the computed values of metrics such as false alarm probability rather than on surrogate physical metrics. These two design approaches are then contrasted in case studies of recently-developed alerting systems.**

***Index Terms*—Alarm systems, algorithms, decision support systems, threshold logic.**

I. Introduction

During the operation of many processes, threats may be encountered that require a human operator's attention. Safety or robustness against these threats is often enhanced through the use of automated alerting systems that independently monitor operations and warn the controller to intervene should it be necessary. Alerting systems are becoming increasingly pervasive and are used in applications including aerospace vehicles, automobiles, chemical or power control stations, air traffic control, medical monitoring systems and even business and financial markets. In addition to providing a final safety net for many processes, some alerting systems actually enable operating in regimes that would not be possible without them. Closely-spaced parallel approaches at airports in poor visibility, for example, are only allowed when certain automated alerting systems are present to provide the necessary level of safety [1]. The additional traffic throughput, then, is directly reliant on the existence and performance of an alerting system. Similarly, critical medical procedures may not be performed without the appropriate monitoring systems in place. Thus, alerting systems play key roles in advancing the capability and scope of many safety-critical processes.

At their core, all alerting systems are discrete decision-making elements, continually determining whether to remain silent or to warn the controller to take some action (whether that controller is a human or automation). Simple alerting systems may monitor a single parameter and a certain threshold must be crossed for a warning to be issued. Examples of simple systems include smoke or fire detectors, flow rate regulators and blood pressure or heart monitors. Complex alerting systems are also in use that make higher-level inferences about safety using multiple parameters. One example is the Traffic Alert and Collision Avoidance System (TCAS) currently in use on jet transport aircraft [2]. TCAS measures the relative range and altitude between aircraft to infer whether a collision is likely and uses a complicated set of algorithms to assign a level of urgency and to provide commands to the pilot to avoid a collision.

To be effective, the alerting system must issue a warning early enough that corrective action can be taken, but not so early that unnecessary nuisance alarms occur. This generates a tension in the design of these systems that is always present, regardless of the application. Failure to properly balance this tension leads to operator distrust of the system, inefficiencies, or accidents. The alerting system must also operate in real time, necessitating some concessions in modeling and computation complexity.

Due to the complexity of applications and the many competing constraints on design, alerting systems today are often designed in a relatively *ad hoc* manner, without the benefit of an overarching theoretical methodology. Sophisticated tools have been developed to design and analyze parts of an alerting problem, but there is little high-level direction of design efforts due to current limitations in alerting system theory. A brief overview of progress in alerting theory will help to put the state of this field in perspective.

A. Progress in Alerting Theory

Alerting systems have actually been in use since well before the advent of electronics, with examples ranging from bringing canaries into mines (to warn workers of asphyxiation danger) to the famous Nightingale Floorboards of Nijo Castle in Kyoto, Japan (to warn the Shogun of intruders in the night). These alerting systems and many other more modern ones, such as fire alarms, operate on the principle of triggering a warning when some state exceeds a certain threshold value. In the examples above, the canary's death signals the exceedance of some level of toxic air; the floors of Nijo Castle were designed to creak

Manuscript received October 13, 2000; revised December 19, 2001. This work was supported by the NASA Ames Research Center under Grant NAG2-1111. This paper was recommended by Associate Editor Y. Y. Haimes.

The authors are with the Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: atlantis@alum.mit.edu; jkkuchar@mit.edu).

Publisher Item Identifier S 1083-4427(02)02694-2.

when any weight over a certain value was placed on them. These are examples of classical decision theory problems in which there is a clear signal to be detected and two possible decision errors: false alarm or missed detection. As one analysis tool, signal detection theory (SDT), was developed in World War II to address radar targeting problems [3]–[5] and was later extended to human decision-making problems in general [6], [7]. SDT provides a formal method by which optimal decision thresholds can be determined, given a known signal value corrupted with sensor noise and is well suited to these types of classical, binary alerting problems.

Many current alerting problems, however, are too complex to be managed with SDT alone. Consider an aircraft collision warning system, for instance. Numerous dynamic states of information may be available, such as the positions, velocities and accelerations of the aircraft. If an alert is triggered, there may be many alerting levels or options for action to avoid a collision. The human pilot's response to an alert is also uncertain and may have a large impact on whether the alerting system is successful. It becomes extremely difficult to apply conventional SDT tools to problems of this scope, where the "signal" is not readily apparent, noise is complex and time-varying and a single decision threshold may not be easily defined.

Complex and successful alerting systems have been developed, despite this difficulty. But, the techniques that are applied are diverse and less formalized than SDT. A review of recent efforts in aircraft collision alerting, for example, found that over 60 decision-making methods have been proposed, tested, or implemented for that task [8]. Although the fundamental issues of false alarm and missed detection still apply, there is no overarching principled approach by which these complex systems are designed, leading to a wide assortment of alternate solutions. An accompanying effect is long development times. TCAS, mentioned in the previous section, required approximately ten years of development, followed by another decade of modifications in response to problems observed in the field. TCAS is based on relatively simple assumptions that aircraft move in straight lines and can only maneuver vertically and the system can only provide accurate warnings less than approximately one minute before a collision. Significantly more complex alerting systems are being proposed and the engineering tools to develop them are needed. The Advanced Conflict Management System, for example, will surpass TCAS and use predictions on the order of several minutes with more complex flight trajectories and maneuvering options [9], [10].

One step toward advancing alerting theory has been the creation of a linkage between SDT and formal state-space modeling methods from the dynamics and control field [11]. This linkage provides the means for recasting a complex, dynamic problem as an analog to an SDT problem, allowing conventional SDT techniques to then be applied. Another recent development has been the identification of formal categories or philosophies of alerting design methods [12]. Decisions made very early in the design process have a large impact on the ultimate performance of a system. A generalized understanding of the relationships between these decisions and performance is just beginning to be developed. Ultimately, a formal taxonomy of alerting methods for complex applications will help inform future efforts so that the most appropriate design path is used for a given problem.

The design of complex alerting systems would be greatly facilitated through the development of a cohesive modeling and design approach based on more formal principles. This paper outlines the fundamental considerations for alerting systems in general and proposes a design process based directly on these principles. To begin, an overview of alerting system operation is provided in the next section, including a more detailed discussion of SDT and other recent developments. Then, two different alerting system design processes are examined and a new approach to design is proposed to make the best use of increasing advances in computational ability. Finally, two example alerting systems are discussed to illustrate the differences that the design process has on the operation of the system.

II. Generalized Model of Alerting System Operation

Alerting systems must predict that an undesirable event will occur before it actually does—the entire reason for having the alerting system is to allow the controller enough time to respond and maintain safe operation. Thus, alerting decisions must be based either directly or indirectly on some form of future prediction of the state of the process. The main factors affecting alerting system performance can be captured through the use of state-space trajectory techniques extended to alerting problems by the authors as described below. The state-space view of alerting developed by the authors facilitates a clear and generalizable description of the important states and metrics (including uncertainties) for any given problem and provides a means by which state trajectories can be examined relative to hazards.

First, a physical state space $\mathbf{X}$ is defined to provide a consistent basis for modeling the current and future states of the process relative to safe and unsafe subsets of this state space. The states that compose $\mathbf{X}$ can be thought of as the set of parameters utilized by the alerting system to characterize the dynamics of threat situations. Undesirable or unsafe states are called hazard space and denoted $\mathbf{H}$. In a transportation system, for instance, $\mathbf{X}$ could include the positions and velocities of vehicles and $\mathbf{H}$ could be any relative vehicle positions smaller than a certain distance. In a medical example, $\mathbf{X}$ could represent states such as heart rate and blood pressure during surgery and there would be regions of $\mathbf{H}$ defined for extremely high or low values of these states. In business, $\mathbf{X}$ could describe product inventory or stock value, with certain limits defined and set apart as $\mathbf{H}$.

The alerting system uses a set of decision metrics based on the states in $\mathbf{X}$ to determine whether an alert is issued. Continuing the transportation example, relevant metrics could be the projected miss distance of two vehicles or the time until collision based on current position and velocity. The choice of appropriate metrics depends on the ability to reflect changes in the threat condition. The better the correlation between the metrics and the threat condition, the higher the potential for an effective alerting system. The alert occurs when some combination of metrics passes alerting threshold levels. This can be modeled by subdividing $\mathbf{X}$ to show what state values result in alerts. This subset of $\mathbf{X}$ is termed alert space and denoted $\mathbf{X}^{\mathbf{A}}$.

As a graphical illustration, Fig. 1 shows a generic two-dimensional (2-D) state space $\mathbf{X}$ where the current state of the system is shown at $\mathbf{x}(0)$. Regions of hazard space and alert space are also shown. By definition, no alerts are generated when $\mathbf{x}$ is outside alert space.

When the state trajectory first enters alert space (point $\mathbf{x}(1)$ in Fig. 1), an alert is given. At this point, the alerting logic has decided that an intrusion into hazard space is likely (solid line) if nothing is done to warn the controller. By initiating the alert, it is expected that some action will be performed (depicted at point $\mathbf{x}(2)$) to alter the course of the state trajectory in order to avoid the hazard (dashed line). There is usually some delay from point $\mathbf{x}(1)$ to point $\mathbf{x}(2)$ as the controller determines the appropriate response to take and due to potential latencies in the system dynamics. Because an alert is a precursor warning to avoid a hazard, alert space should encompass all regions of hazard space. Selecting the proper size of alert space is a key consideration in the design of the alerting system. If $\mathbf{X^A}$ is too large (or equivalently, when the system overpredicts the level of threat at the current state), there may be unnecessary nuisance alarms; if $\mathbf{X^A}$ is too small (or the system underpredicts the level of threat), hazard space may not be avoidable once an alert is issued.

A. Alerting Performance

Uncertainty is inherent in the prediction of any future outcome and the same holds true in the design of alerting systems. Due to random effects, there are variabilities and uncertainties in the future states of any process and this produces errors in the prediction of a hazard and causes problems in the design of an effective alerting system.

Methods to quantify the impact of uncertainty on decision-making have been developed and applied over the past several decades. As discussed above, of particular relevance to alerting systems is signal detection theory (SDT) [3]–[7]. Specific performance metrics are defined in SDT, including the probability of false alarm, $P(FA)$, probability of correct detection, $P(CD)$ and probability of missed detection, $P(MD)$. By plotting $P(CD)$ against $P(FA)$ as a function of the signal detection threshold, the performance tradeoffs of the system can be observed in a receiver operating characteristic (ROC) curve. The ROC curve can then be used to determine the appropriate threshold setting to best balance false alarms against missed detections for the given application.

SDT as originally posed required that the value of the signal and the probability density function of the noise are known and was based on a single decision metric. These assumptions are often sufficient in fixed decision-making problems, but are problematic in dynamic and more complex multivariable cases where the system states and uncertainties change with time. SDT has since been reformulated to more directly relate to discrete alerting decisions in dynamic systems [11]. The use of state-space modeling described above is central to this connection. It can be shown that the probability of entering hazard space along the nonalert trajectory (the solid line in Fig. 1) is analogous to $1 - P(FA)$ in SDT and the probability of entering hazard space along the alert trajectory (the dashed line in Fig. 1) is analogous to $P(MD)$ in SDT. A corresponding metric to $P(CD)$ has also been defined, called the probability of successful alert, $P(SA)$. The value of $P(SA)$ is the probability that hazard space is not entered after an alert, or $P(SA) = 1 - P(MD)$. These connections allow for techniques analogous to ROC curves to be applied to alerting decision-making problems in which the future state trajectory of a process can be predicted with known uncertainty distributions [11].

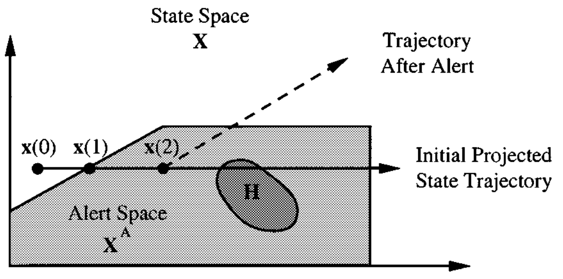

Fig. 1. Example state-space diagram.

The combination of $P(SA)$ and $P(FA)$ provides a fundamental basis for quantifying the performance of an alerting system and these two metrics have been used in recent analyses [13]–[15]. These metrics, at a minimum, are needed to examine the false alarm/missed detection tradeoff that invariably occurs. Additional metrics could also be defined if more detail is desired. For example, it may be of interest to separately measure the probability that an alert actually causes a hazardous incident to occur, contrasted against the total rate of false alarms or missed detections [13]. Other metrics may certainly be warranted as well, such as computational requirements, implementation and hardware cost, or operator workload.

B. Trajectory Models

The application of SDT can have valuable benefits in the design and evaluation of alerting systems. However, use of SDT techniques assumes that the underlying dynamic models and uncertainty distributions are known accurately. Typically, such accuracy may be limited, leading to additional performance losses, discussed as follows.

To begin, first consider a model denoted $\mathbf{T}$ that represents the best engineering estimate of the future trajectory of the process, including uncertainties. $\mathbf{T}$ is what would be used in an offline analysis, for example, to assess the performance of the system. Due to sensor and computational limitations, complete knowledge of the uncertainties and the most accurate prediction of the future state often cannot be attained by the alerting system itself. Instead, an approximate working trajectory model, $\mathbf{W}$, is actually used by the alerting system for its decision making. For example, in air traffic collision alerting systems, it is typical to assume that aircraft travel in straight lines. The working model then is a simplification of actual aircraft trajectories, which may include turns or other maneuvers.

Ideally, one would like $\mathbf{W}$ to be an exact copy of the best-estimate probabilistic trajectory, $\mathbf{T}$. The probability of entering hazard space predicted from the working model would then be the same as the probability during offline evaluation. Unfortunately, differences between $\mathbf{W}$ and $\mathbf{T}$ make this unlikely except for a short time step into the future. For example, two models,

$\mathbf{W}$ and $\mathbf{T}$, are depicted in Fig. 2. As shown, the working trajectory model used by the alerting system underpredicts the actual variability in the future trajectory that is assumed during evaluation. This may then lead to observing late alerts or missed detections from the system. Additional errors are introduced when the best-estimate trajectory model $\mathbf{T}$ is not an accurate representation of what actually occurs in operation. These errors can be reduced by attempting to make the evaluation scenarios in $\mathbf{T}$ match as well as possible what is actually expected to occur. However, some residual error will in general always be present.

After a recent survey of alerting methods, it is apparent that working models of trajectories can generally be reduced to one of three categories: single path, worst case and probabilistic [8]. In the single path approach, the current states are projected into the future along a single trajectory without direct consideration of uncertainties. An example would be extrapolating a vehicle's position based on its current velocity vector [Fig. 3(a)]. The single path projection method is straightforward and provides a best estimate of where the state will be, based on the current state information. In situations where state trajectories are very predictable (such as when projecting only a few seconds into the future), a single path model may be quite accurate. Single path projections, however, do not directly account for the possibility that the process or environment may not behave as expected—a factor that is especially important in long-term decision making. Generally, this uncertainty is managed by introducing a safety buffer (e.g., minimum miss distance between vehicles) to reduce the likelihood of missed detections.

The other extreme of dynamic modeling is to examine a worst case projection. Here, it is assumed that the state trajectory could follow any of a range of behaviors. If any one of these trajectories could result in entry to hazard space, then an alert is issued. The result is a swath of potential trajectories which is monitored [Fig. 3(b)]. Worst case approaches are conservative in that they can trigger alerts whenever there is any possibility of entering hazard space within the definition of the worst case trajectory model. If such trajectories are unlikely, protecting against them may result in a high false alert rate. Accordingly, the worst-case approach may be appropriate when it is desirable to be conservative, or when dynamics are constrained within known bounds. Note that a simple threshold test against the current state of the process (e.g., a blood pressure alarm for a patient) is essentially using a worst case trajectory model, under the assumption that passing the threshold could result in a serious medical emergency.

In the probabilistic modeling method, uncertainties are explicitly used to develop a set of possible future trajectories, each weighted by its probability of occurrence. For example, a distribution of future vehicle positions could be obtained by modeling uncertainties in guidance [Fig. 3(c)]. A probabilistic approach provides an opportunity for a balance between relying too heavily on the state adhering to a single trajectory versus relying too heavily that the state exhibits a worst case behavior. The advantage of a probabilistic approach is that decisions can be made on the fundamental likelihood of entering hazard space—safety and false alarm probabilities can be assessed and considered directly. The probabilistic method is also the most general, since single path and worst case models can be considered subsets of probabilistic trajectories. The single path trajectory corresponds to a case in which the state will follow a given (e.g., maximum likelihood) trajectory with probability 1; the worst case model is one in which the state may follow any trajectory with equal likelihood.

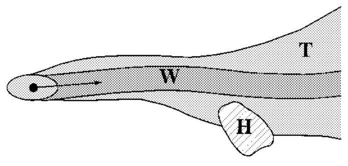

Fig. 2. Example working trajectory model (W) and evaluation model (T).

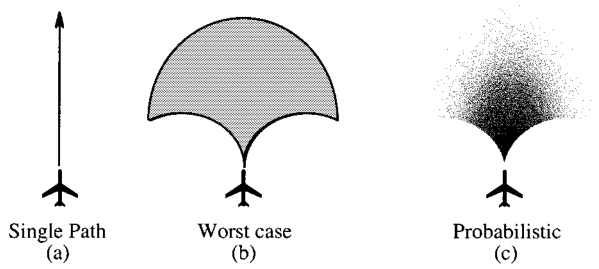

Fig. 3. State propagation methods.

III. Alerting System Design Methods

Having introduced several underlying performance issues relevant to alerting systems, it is appropriate to discuss the larger-scale design process typically used in recent applications. Following this examination, an alternative design approach is proposed to improve alerting system performance and the efficiency of the engineering development process.

A. Typical Alerting System Design Process

Fig. 4(a) diagrams the general, iterative design process often used today in setting alerting threshold metrics, termed here the physical metric method. The design process originates with the working trajectory model, $\mathbf{W}$, that the alerting system uses to estimate the future states of the process.

Based on this model, physical metrics are then computed in the space $\mathbf{X}$ (e.g., temperature, distance, or time to impact) to specify the alert threshold criteria that delineate alert space. These physical metrics are typically readily available through sensors (either directly or through some additional filtering or estimation) and so until recently have been the only practical choices for use in a real-time system. Advances in computing power, however, are now opening up new possibilities to use more complex, derived decision metrics. This opportunity is discussed in the next section.

Continuing for now to focus on the physical metric design approach, the alerting thresholds in $\mathbf{X}$ are initially set based on a combination of analysis and user expertise, but usually require some fine tuning from test scenarios through simulations or through observed performance in the field, as shown in Fig. 4(a). These test scenarios form the best-estimate set of trajectories, $\mathbf{T}$, that the system is to be exposed to and evaluated against.

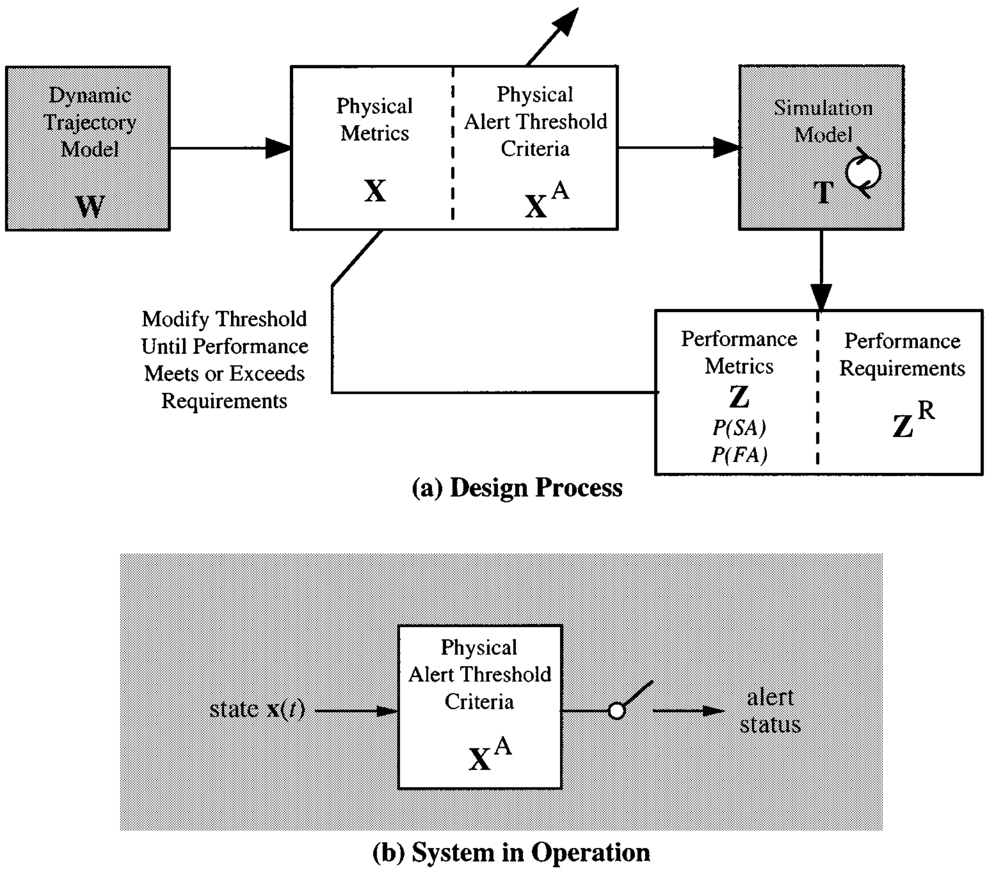

Fig. 4. Physical metric design process and operational process.

The values used to describe and evaluate the resulting performance themselves form a state space which is denoted as $\mathbf{Z}$. A 2-D example of $\mathbf{Z}$ could have states of probability of successful alert $P(SA)$ and probability of false alarm $P(FA)$. What is ultimately happening is that the alert space in $\mathbf{X}$ is being mapped into performance space in $\mathbf{Z}$. In a transportation application, for instance, the alerting thresholds (in the state space $\mathbf{X}$) might be based on the current range (r), range rate ($\dot{r}$) and predicted miss distance (m) between vehicles. Given a set of evaluation scenarios, the probabilities of false alarm and successful alert would be some function of these threshold values: $P(FA) = g(r, \dot{r}, m)$ and $P(SA) = h(r, \dot{r}, m)$, respectively. The functions $g()$ and $h()$ would be specific to the scenarios used for evaluation in $\mathbf{T}$. The resulting performance of the system can be thought of as forming a new vector, $\mathbf{z^A} = [P(FA)\ P(SA)]$, in the state space $\mathbf{Z}$. In general, the system performance can be expressed as a mapping from the threshold settings in physical state space, $\mathbf{X^A}$, to the performance state space:

$$\mathbf{z^A} = f\left(\mathbf{X^A}\right) \tag{1}$$

where $f()$ is a function that also depends on the evaluation scenarios in $\mathbf{T}$. The governing function, $f()$, can generally not be explicitly expressed or defined during the design process. Thus, it becomes difficult to predict the outcome of changes or even to make informed comparisons between different sets of simulation runs.

The specific performance requirements that must be met are represented by a subset of $\mathbf{Z}$ denoted $\mathbf{Z^R}$. For example, $\mathbf{Z^R}$ could be the region of performance state space where the probability of false alarm is less than 0.05 and the probability of successful alert is greater than 0.99. If these requirements are not met, then the metrics of the alerting threshold are iteratively adjusted until satisfactory performance is achieved. In some cases, it may also be necessary to modify the decision metrics that are used or to alter the working trajectory model (e.g., through the addition of new sensors that provide additional state information).

Once the system has been designed in the manner discussed above, the alerting thresholds are encoded and used in the real-time operation of the alerting system [Fig. 4(b)]. In operation, a physical metric system takes in the current measured or estimated state of the process, compares these physical states against the predefined alerting thresholds, $\mathbf{X^A}$ and issues an alert as appropriate. This is relatively simple and generally can be performed in real time.

Returning to the design process, Fig. 5 shows a conceptual illustration of mapping the alerting thresholds in the physical state space of $\mathbf{X}$ to the state space of performance measures, $\mathbf{Z}$. The alert space in $\mathbf{X}$ is denoted by the region $\mathbf{X^A}$ and the required performance region to be met in $\mathbf{Z}$ is designated $\mathbf{Z^R}$. When $\mathbf{X^A}$ is mapped into $\mathbf{Z}$, it becomes a single state vector $\mathbf{z^A}$ as described by (1). If $\mathbf{z^A}$ is outside the region of $\mathbf{Z^R}$, as depicted in the leftmost illustration of Fig. 5, then the performance requirement is not met and the threshold metrics need to be ad-

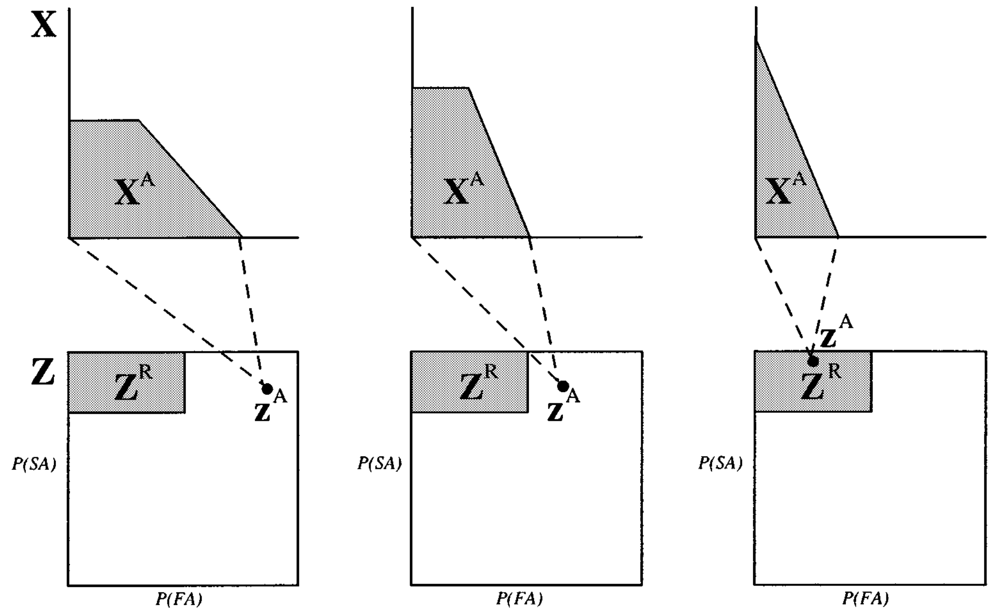

Fig. 5. Mapping to performance state space.

justed until suitable performance is obtained. This is shown in the series of drawings going from left to right and represents the iterative search and fine tuning of the feedback loop from Fig. 4(a). Notice that $\mathbf{X}^{\mathbf{A}}$ is changed in each step.

There are several important insights when the design process is portrayed in the manner shown in Figs. 4(a) and 5. Because of the feedback loop in the design process, the final settings of the alerting thresholds (e.g., r, $\dot{r}$ and m) are actually surrogate values for the desired performance metrics. This is quite similar to a neural network scheme where the evaluation scenarios define a reference model from which the thresholds are adjusted to meet or optimize performance metrics [16]. Even if the evaluation scenarios or the dimensions of alert space are changed, the specification for the minimum level of performance ($\mathbf{Z}^{\mathbf{R}}$) would likely remain unaltered. Thus, the alerting thresholds defined in $\mathbf{X}$ are actually indirect controls of the alerting performance in $\mathbf{Z}$. Differences between the working and evaluation models are indirectly managed through the iterative, trial-and-error design process.

If an acceptable $\mathbf{X}^{\mathbf{A}}$ cannot be found to satisfy the performance constraints, then there are four possible corrective options. The first is to change to a different set of threshold metrics (i.e., change the states that define $\mathbf{X}$). The threshold metrics may not have been appropriate for the types of situations that are encountered, or else there may have been an insufficient number of metrics to handle the complexity of situations. This is analogous to regression modeling in which a different set of metrics may provide a better correlation with known data.

A second option is to partition out the thresholds to handle more situation-specific groupings: different sets of threshold criteria are used for different threat conditions. Take the example shown in Fig. 6(a), where one alerting threshold, $\mathbf{X}^{\mathbf{A}}$, is used for three different types of threat encounters, with the corresponding mapping functions $f_1()$, $f_2()$ and $f_3()$. In this case, only $f_1()$ maps adequately to the required performance specifications. If $\mathbf{X}^{\mathbf{A}}$ were to be utilized for all three encounter situations, the overall performance of the system would be a weighted average of each of the individual outcomes. If situations 2 and 3 were rare relative to situation 1, it is possible that the system design shown in Fig. 6(a) would satisfy overall performance constraints. Should situation 2 or 3 be encountered in operation, however, performance would not meet the specifications.

In an attempt to improve system performance, Fig. 6(b) shows two modifications to the alerting logic. First, the original alerting threshold ($\mathbf{X}^{\mathbf{A}}$) was split into two forms, $\mathbf{X}_1^{\mathbf{A}}$ for the types of threat encounters represented by $f_1()$ and $\mathbf{X}_2^{\mathbf{A}}$ for encounters represented by $f_2()$. Second, a different state space, $\mathbf{X}'$, was formed based on different metrics than $\mathbf{X}$ (e.g., through the use of different sensor information). In the space of $\mathbf{X}'$, a third alerting threshold $\mathbf{X}_3'^{\mathbf{A}}$ was defined for the conditions represented by $f_3()$. These new thresholds then allow the system performance to map into the desired specification region $\mathbf{Z}^{\mathbf{R}}$. However, this performance comes at the cost of an increased number of threshold metrics designed and tailored specifically for different types of encounters. Finally, how these new thresholds would be determined is complex and would probably involve a series of trial-and-error tests. The difficulty in determining how to adjust thresholds to achieve performance specifications is one of the drawbacks of the physical metric method.

The third corrective option is to limit the use of the alerting system to specific types of encounters. In problematic situations, the alerting system may need to be inhibited to prevent false alarms. For example, a collision warning system may have to be modified to not alert during proximate situations when two aircraft are flying on parallel courses. This corrective measure in fact is used with TCAS during closely-spaced parallel approaches to reduce false alarm rate. Inhibiting alerting systems, however, reduces their utility because certain threat conditions may no longer be protected.

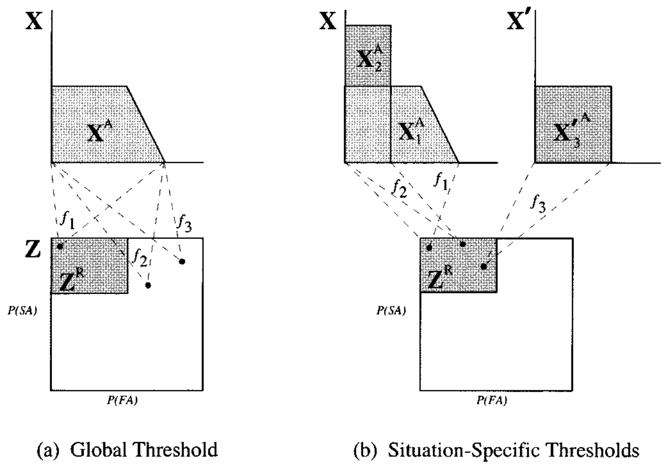

Fig. 6. Use of situation-specific thresholds.

The fourth and final corrective option is to utilize a different threat resolution strategy. Given that the performance is partly based on the ability to avoid a hazard after an alert is issued, it is natural to assume that some metric such as $P(SA)$, which reflects the performance of a specific hazard avoidance action, is included in the performance state space. The idea would be to use a different or a more drastic avoidance maneuver, modifying the dashed line in the state space diagram in Fig. 1.

It is difficult to provide a general principle as to how or in what order the four possible corrective actions outlined above should be implemented. The methods to use depend largely on the specific alerting problem under investigation. Whichever corrective actions are taken must generally be applied through trial-and-error, further illustrating the design challenges inherent in physical metric alerting methods.

B. Performance Metric Alerting (PMA) System Design

In the previous section, the relationship between performance metrics and physical metric alerting thresholds was examined. It was explained that the performance requirement values (in space $\mathbf{Z}$) were actually driving the threshold settings (in space $\mathbf{X}$) in the physical metric approach. Given current computing capability, the performance metrics may now be obtained directly in real-time in some cases, thereby negating the need to implement the additional iterative steps during design to map physical metrics into performance metrics.

In Fig. 7(a), a new, more direct approach to the alerting process is presented, termed the performance metric alerting (PMA) method [17]. Here, the performance metrics (in the space of $\mathbf{Z}$) are directly computed in real time by the alerting system and alerting decisions are based explicitly on them rather than on ersatz metrics in the space of $\mathbf{X}$. Thus, the core performance values determine when and where to alert.

In the PMA concept of Fig. 7(a), the working trajectory model used by the alerting system is the same as the model upon which performance would be judged ($\mathbf{W} = \mathbf{T}$). In doing so, the alerting algorithm is obtaining a direct prediction of the likelihood of hazards and also the ability to avoid them. These values can then be utilized as the threshold metrics in the performance state-space of $\mathbf{Z}$ with the alerting criteria (performance requirements) denoted by $\mathbf{Z^R}$. In general, $\mathbf{Z}$ could be composed of the states of $P(SA)$ and $P(FA)$, though other performance metrics could be used. Overall system performance can still be tested through a simulation if desired. This could lead to modifications in the trajectory models that are used, as diagrammed in Fig. 7(a).

In actual operation [Fig. 7(b)], the PMA system performs a probability estimation using the sensed state $\mathbf{x}$ in combination with the working probabilistic trajectory model. This probability estimation can be performed online using an analytical solution or Monte Carlo simulation [18]–[20], depending on the complexity of the problem and on computational constraints.

The probabilistic trajectory model that is used may itself be a function of $\mathbf{x}$. For example, a baseline trajectory model may be used in most cases to propagate the current state into the future and thereby estimate the probability of entering hazard space. If additional information were available to the alerting system (e.g., specific intentions of the operator to turn the vehicle), then the trajectory model can be modified online and a new probability estimate can be attained.

The probabilistic outputs of the trajectory model are the performance measures of interest, such as $P(SA)$ or $P(FA)$, which can be represented by the state $\mathbf{z^A}$. This state is then compared against the performance criteria that form the alerting thresholds and an alert is issued only when the appropriate performance measures are satisfied.

In order to compute the state of the process in space $\mathbf{Z}$, it is necessary to use significantly more complex computations than are required for physical metrics. Whereas physical metrics may be immediately available from sensors, performance metrics must be derived through a series of computations performed on a probabilistic trajectory model. Since the alerting system must estimate these metrics online, there are stringent requirements on computation ability if PMA is to be used. PMA has not been a practical option until recently; however, current computational power is now enabling PMA to be used in some real-time applications, as is described through an example in the next section.

As indicated by the dashed line in Fig. 7(a), it may also be possible to map the probabilistic performance values to other metrics in a physical state space, $\mathbf{X}$, with its corresponding alert space $\mathbf{X^A}$. This may allow for easier interpretation of the threshold logic. However, this translation may not always be possible unless a one-to-one mapping of variables exists. The problem is akin to the same type of dilemma associated with inverse kinematics.

C. Discussion of the Two Design Approaches

The physical metric process leaves open many different possible variables for use as metrics in the physical state space. Several surveys of air traffic alerting algorithms developed through the physical metric process found scores of different metrics being used by different researchers as a basis of alerting decisions [8], [12]. Yet, the core performance specifications are nearly always in terms of false alarm and safety probabilities. In essence, the physical metric approach is bypassing an accurate dynamic model either completely or partially while leaving the fine tuning to pattern matching. The reason for the required

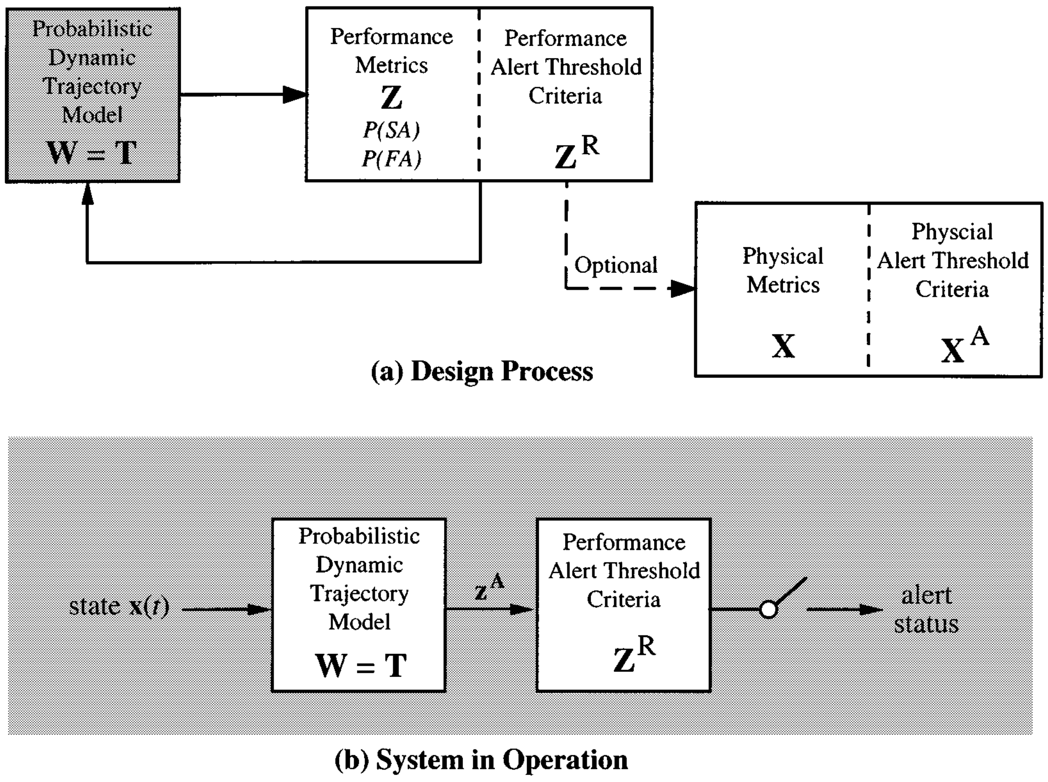
Probabilistic Dynamic Trajectory Model
W = T
Performance Metrics
Z
P(SA)
P(FA)
Performance Alert Threshold Criteria
Z^R
Optional
Physical Metrics
X
Physcial Alert Threshold Criteria
X^A
(a) Design Process
state x(t)
Probabilistic Dynamic Trajectory Model
W = T
z^A
Performance Alert Threshold Criteria
Z^R
alert status
(b) System in Operation

Fig. 7. Performance metric design process and operational process.

mapping is because of the disparity between the working trajectory model, $\mathbf{W}$ and the best-estimate model, $\mathbf{T}$. For example, a collision avoidance time-to-impact metric is typically based on the assumption for $\mathbf{W}$ that the closure rate between two vehicles is constant. The threshold value of time to impact at which an alert is generated, however, is generally determined through iteration over a series of simulations with $\mathbf{T}$ where closure rate actually does vary. Without the ability to obtain an accurate prediction of a hazard encounter directly from its own trajectory model (since $\mathbf{W} \neq \mathbf{T}$), the physical metric alerting logic is forced to trial-and-error methods and performance in general deteriorates.

However, there may really be no need for this mapping since a prediction of alerting performance may be obtained directly by using probabilistic trajectory modeling. This can occur when $\mathbf{W} = \mathbf{T}$. The working trajectory model, $\mathbf{W}$, is therefore a representation of the same simulation scenarios that were used to evaluate performance in Fig. 4(a). In order to do this, $\mathbf{W}$ must be allowed to exhibit any trait that would have been characterized in the evaluation simulations, including the likelihood of human errors and blunders.

The approach shown in Fig. 7(a) requires a direct modeling of the uncertainties in the state trajectories of the process, which in turn can help determine the impact and influence of each source of uncertainty on the alerting performance. It should also be noted that the physical metric method tends to develop a *global* threshold setting as opposed to a *situation-specific* threshold (one that is individually tailored to the current encounter situation). Due to the offline nature in which thresholds are set, the physical metric system achieves the desired performance specifications when considered over the sum of the evaluation scenarios. In contrast, the performance metric approach ensures that the desired performance is met every time that an alert is issued. It can be shown that a global threshold exhibits a higher level of uncertainty and reduced overall level of performance than a situation-specific solution [17]. The result of a global threshold is a compromised design that achieves the desired performance constraints *on average*, but which may not achieve them in a given specific situation.

When operating a performance metric system as shown in Fig. 7(b), there is a need to continuously update the dynamic model, $\mathbf{W}$, utilized by the alerting system to keep up with the current situation. As long as the uncertainties in the trajectories can be modeled, the update process is a natural progression of new states and other data that is brought in to modify $\mathbf{W}$. At any instant in time, the current states are projected into the future using $\mathbf{W}$ and the probabilistic performance metrics, such as $P(FA)$ and $P(SA)$, are computed. The decision to alert is then made directly from these performance estimates. So, with PMA, the trajectory model is tailored or adjusted to best match the current scenario in real time but the ultimate alerting thresholds ($\mathbf{Z}^{\mathbf{R}}$) are static. In contrast, in physical metric alerting, the trajectory model may be fixed, but the alerting thresholds may be modified in response to the specific situation being encountered (recall the discussion of Fig. 6). But, because it is not directly apparent how these physical thresholds should be modified to still meet performance specifications, the overall system performance may suffer. For example, TCAS (designed using physical metrics) uses a single trajectory model based on the currently-estimated range between aircraft and their closure rate, regardless of aircraft altitude. The alerting thresholds themselves, however, are adjusted depending on aircraft altitude to account for higher closure rates and reduced altimeter accuracy.

The PMA approach is situation-specific since the alert decision is based entirely on current information specific to the encounter. All knowledge of the current situation, including the effects of uncertainties, is contained in $\mathbf{T}$. The idea behind the performance metric approach is to allow the computation of the threat condition online as the situation occurs, using robust decision thresholds that are invariant with the situation. It is analogous to computer chess programs that wait for a move to be made; then based on the current configuration, they propagate the probable moves of each chess piece (out to a finite number of moves ahead) and make a decision based on the results. Even in the limited confines of the chess board and the incredible processing power of today's supercomputers, it is nearly impossible to determine what all the moves should be prior to the start of the game (except possibly for initial standard opening "book" moves). There are just too many possible configurations even on the discrete space of a chess board. Instead, the simulations are performed by the computers on the run as the situation unfolds and the decisions are situation-specific based on the current configuration.

By keeping $\mathbf{W} = \mathbf{T}$, the alerting decision is tailored specifically to the best estimate of the current conditions of the encounter. Any changes to the system state or knowledge of the future trajectory are accounted for directly and implemented as the situation occurs. This resolves the problem of pre-determining separate threshold metrics for every possible encounter situation as was shown in Fig. 6. PMA does require, though, a significant increase in real-time computation. However, given advances in computational power, such an approach is now becoming feasible in some cases as is discussed below. In the following two sections, case studies of the physical metric and performance metric approaches are presented to compare and contrast the design methods that are used.

IV. Example of the Physical Metric Design Approach

The Traffic Alert and Collision Avoidance System (TCAS) has been implemented on U.S. jet transports since the early 1990s as concern grew over the potential of future midair collisions. TCAS has been credited with several "saves" in near-collision situations and has also been accused of causing false alarms that nearly led to accidents [21]. The system is quite complex; the discussion here only focuses on a simplified set of metrics and logic. The interested reader is referred to [2], [22]–[25] for more detailed descriptions of TCAS.

In abbreviated terms, the TCAS logic calculates a collision threat in the horizontal and vertical dimensions separately and issues alerts if both criteria are met. The algorithm is based on the relative range (r) and range rate ($\dot{r}$) and also the relative altitude (h) and altitude rate ($\dot{h}$) between two aircraft. In the framework presented in this paper, these metrics form the physical state space $\mathbf{X}$. TCAS uses a two-stage process with a cautionary alert called a Traffic Advisory (TA) and a warning alert called a Resolution Advisory (RA). RAs provide vertical avoidance commands; TAs are merely attention-getting alerts and lack any resolution guidance. The following discussion focuses on RA alerts only.

Though the TCAS thresholds are more complex, they can be summarized by what is commonly referred to as the Tau (τ) Criterion:

$$\frac{r - DMOD}{-\dot{r}} < \tau \qquad (2)$$

where τ is a threshold parameter with units of time and DMOD is a safety buffer distance. Essentially, if the predicted time to reaching a distance of DMOD between aircraft is less than a threshold time τ, an RA alert is issued. Similar metrics are used in the vertical dimension. The TCAS logic assumes a straight line, single path working trajectory model and DMOD acts as a buffer to account for possible deviations or sources of error. Metrics such as DMOD and τ effectively determine the frequency with which RAs are given. Reducing these values will reduce the alert rate and number of disruptions caused by false alarms [25]. However, the tradeoff is the risk of missed or late alerts due to insufficient warning time. The desire is a balance between false alarms and collision protection that TCAS is intended to provide.

To achieve this balance, the various design metrics were set offline using an *ad hoc* iterative approach through Monte Carlo simulations of aircraft encounters. Modifications were also made from data and user comments during actual in-flight operations. In one set of evaluations of TCAS, for instance, a large database of pairwise aircraft encounters was generated from actual recorded tracks in the United States airspace [26], [27]. Using this database, ten types of vertical encounter geometries were defined (Fig. 8) which were considered to encompass typical aircraft maneuvers. In evaluating the performance of the system, a large number of different simulation runs were used to cover each of these ten encounter classes, leading to millions of simulation runs [25]–[27]. Changes to the threshold metrics were then suggested due to the results of these simulations in terms of false alarms and collision rates (metrics in the performance space $\mathbf{Z}$). In the framework presented above, these different encounter scenarios form the evaluation trajectory model $\mathbf{T}$, since it is on their basis that the alerting system is deemed acceptable or not.

As was the case in the example in Fig. 6(a), TCAS is unable to satisfy performance constraints using a single threshold setting for all encounter situations. The logic was modified with a number of if-then branches to manage different encounters and additionally the values of metrics such as τ and DMOD vary as a function of altitude and flight condition. Thus, there is a large amount of tuning of the alerting threshold metrics, even for a seemingly simple design in which aircraft are assumed to fly on straight, constant velocity paths and only use vertical evasive maneuvers. Still, TCAS is a remarkably effective system, especially considering the limited amount and quality of information upon which it must base its decisions.

These modifications to the alerting system can involve a tedious process of breaking up and grouping the scenarios to cover all possible encounter geometries and flight conditions. In a

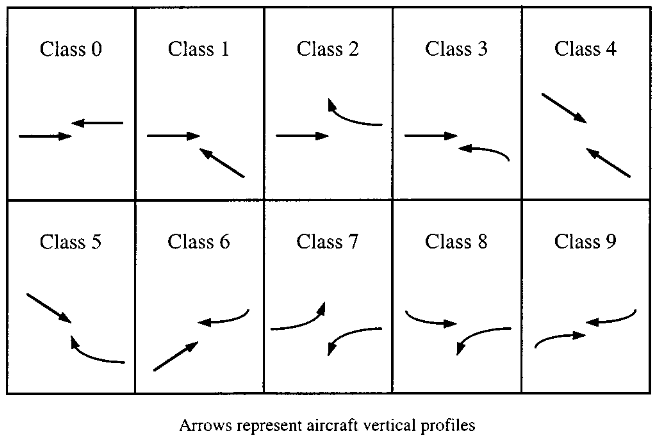

Fig. 8. TCAS evaluation situations (adapted from [27]).

more general air traffic conflict alerting environment when waypoints, future intention information, many aircraft and three-dimensional (3-D) encounters are all fused together, it becomes extremely difficult to utilize such a scheme to amalgamate all the individual situations and develop separate thresholds for each one. In the physical metric approach, it may be necessary to perform the task iteratively for each scenario in order to map out a different alert space, $\mathbf{X}_i^{\mathbf{A}}$, for individual encounters. This would be true unless, of course, one could pick a set of threshold metrics that would allow settings to be virtually invariant of the encounter situation. In fact, this is the principle behind the direct performance metric approach shown in Fig. 7.

V. Example of the Performance Metric Design Approach

A prototype alerting logic for detecting air traffic conflicts was developed by the authors using the performance metric design approach discussed above [17]–[19]. The design of this system is useful as an example to discuss how the performance metric approach can be implemented. Whereas TCAS uses physical metrics such as range or time as the basis for alerting decisions, the system discussed here uses performance metrics related to false alarm and success probabilities to determine whether to issue an alert. Although shown in contrast to TCAS, the prototype system is not meant to replace or otherwise compete against TCAS. Instead, it serves as an illustrative example only. It should also be mentioned that other performance metric designs are appearing. See, for example, [20] or [28].

In the design of the prototype system, it was presumed that state information from surrounding aircraft (position, speed and heading) was available through inter-aircraft data link communication. These metrics can be considered to form the physical state space $\mathbf{X}$ since they are useful in describing trajectories and whether hazard space is entered. Hazard space was represented by a traffic conflict, defined as a situation in which one aircraft enters a cylindrical protected zone around another aircraft five nautical miles in radius and extending 1000 feet above and below the aircraft.

The aircraft trajectory working model included uncertainty in the current position estimate, future along- and cross-track position variability and the potential for and magnitude of course changes. The trajectory model was continually modified in response to any additional intent information (e.g., through a data link of the aircraft's future flight plan). If such intent information was not available, more uncertainty was injected into the trajectory model to dilute the confidence with which an alerting decision was made. Thus, the working model used by the system included as much information as possible regarding the future trajectory of each aircraft. A more complete discussion of the trajectory models can be obtained from [18] and [19].

The probability of a conflict, $P(C)$, was defined as the probability that one aircraft will enter another's protected zone given that no alert is issued and that the aircraft follow the working trajectory model. To calculate $P(C)$, the positions of the two aircraft must be projected into the future to determine the likelihood of a protected zone violation. Due to the complexity of the dynamics involved, Monte Carlo simulations were used to estimate this probability.

Each Monte Carlo run consisted of examining the trajectories and determining whether separation minimums of the protected zone were violated. The trajectories varied randomly with each run according to the uncertainty distributions chosen to define the trajectory model. After a certain number of Monte Carlo runs, a count of the number of protected zone intrusions was made. Dividing the number of intrusions by the total number of Monte Carlo runs was then an estimator of $P(C)$. As implemented, $P(C)$ was estimated to within 0.015 (with 99% confidence) after approximately 1 s of computation time on a Silicon Graphics Indigo workstation. This is based on 10 000 simulation runs assuming a binomial process. The computation time was small enough that the system could be implemented in real-time human-in-the-loop studies.

A multistaged threshold approach was used to provide a series of alerts to indicate trends in conflict hazard. The multi-stage approach allowed the type of alert to be tailored to the level of threat. Low-probability threats resulted in relatively passive alerts such as changing the color of a traffic symbol on a cockpit display. High-probability, urgent threats produced aural warnings to actively inform the pilots of the conflict. The appropriate stage to use in a given situation depended on the state vector in the performance space $\mathbf{Z}$, rather than in a physical state space $\mathbf{X}$.

The performance space $\mathbf{Z}$ had two dimensions. The first was $P(FA)$, which is related to the probability of conflict that is expected should the aircraft continue along their planned paths. The higher the value of $P(C)$, the smaller the value of $P(FA)$. The second dimension in $\mathbf{Z}$ was represented by the flexibility with which a conflict could be successfully avoided with 95% confidence. This involved determining what types of standard maneuvers (turns, climbs, descents, or speed changes) would resolve the conflict with probability 0.95, computed through additional Monte Carlo simulations. These maneuvers served as benchmarks for estimating the ability of the aircraft to avoid a conflict. The number of available avoidance maneuvers then was defined as the second dimension of $\mathbf{Z}$. Probability of collision could also have been used in this example as a performance metric if desired. Whereas TCAS was concerned with collision avoidance (and thus used probability of collision as a performance metric), the prototype system here was intended for earlier, less urgent warnings so that less aggressive avoidance actions would be required. Thus, the use of a probabilistic maneuver flexibility metric in $\mathbf{Z}$ is appropriate in this case.

Fig. 9 shows a schematic of the performance state space, including the alerting requirements, $\mathbf{Z^R}$, for the four alert stages. When aircraft are far from one another, $P(FA)$ will be close to 1 and many maneuvers could be used to successfully avoid a conflict. Thus, the state would be located in the upper right corner of Fig. 9. As a threatening intruder aircraft nears another, $P(FA)$ generally decreases as it becomes more certain that an alert will be needed. Additionally, flexibility is lost as the aircraft get closer or if there are other aircraft in the vicinity, resulting in movement downwards in Fig. 9. As the state crosses the boundaries shown in Fig. 9, the alert level would change to reflect the increased threat level. The prototype alerting logic based on the performance metric approach has been successfully used in a number of human-in-the-loop flight simulation experiments at the NASA Ames Research Center; see, for example, [29] and [30].

The specific makeup of the different regions of $\mathbf{Z^R}$ in this example is not of importance here; rather, the intent is to demonstrate the real-time use of higher-level performance metrics (e.g., $P(FA)$) as the alerting thresholds instead of lower-level physical surrogate metrics (e.g., time to impact, or miss distance).

VI. Conclusion

The development of formalisms behind the design of complex hazard alerting systems would greatly facilitate the development of effective systems in an efficient manner. This paper outlines several key performance considerations and modeling approaches, including the application of state space methods and the need to consider the relationship between the working model used by an alerting system and the evaluation model of the environment in which it will operate.

Ultimately, any alerting system is designed based on predictions of future events. It is the accuracy of these predictions compared against what actually would occur that determines how effective the system is. Prediction accuracy, in turn, is directly affected by what trajectory models and decision metrics are used. Physical metrics such as temperature, pressure, distance, or time to impact have traditionally been used as the basis for defining and setting alerting thresholds across a variety of applications. These metrics are often directly measured by sensors and are easily processed online and thus are natural choices for use as decision metrics. However, a different set of metrics is usually applied to evaluate system performance. Performance specifications are generally based on event outcomes, including false alarm rates and probabilistic measures of safety, as these measures are at the essence of the alerting design tradeoff. The traditional alerting system design process is shown to fundamentally involve tuning the physical decision metrics so that the system meets the desired performance specifications. Because this tuning is done iteratively offline, the system in operation is less flexible in adapting to a specific situation. Although this design process can and has led to effective alerting systems, a more direct approach is proposed in this paper.

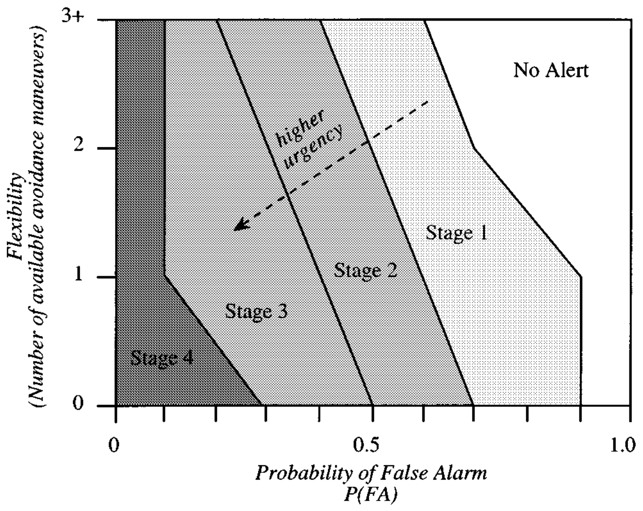
3+
2
1
0
Flexibility (Number of available avoidance maneuvers)
No Alert
higher urgency
Stage 1
Stage 2
Stage 3
Stage 4
0
0.5
1.0
Probability of False Alarm P(FA)

Fig. 9. Performance state space Z for prototype alerting system.

In the proposed approach, the core event-based performance specifications themselves are used as the decision metrics or alerting thresholds. This bypasses the need to map physical metrics into the performance specifications and can lead to a more efficient and more effective system design. The critical prerequisite in order to use the proposed approach is that the performance specifications must be defined at the start and must be calculated during system operation. Given advances in computational power, it is now becoming practical to estimate probabilistic measures such as false alarm probability online and thus it may be feasible to base alerting decisions on these fundamental metrics. Performance metrics other than probabilistic measures are also certainly possible (e.g., economic costs). The key concept for using performance metric design is to clearly identify those metrics that best define whether a given alerting decision is acceptable and then compute and use those metrics as the decision metrics in operation.

To summarize, both the physical and performance metric methods typically require some form of probabilistic modeling to predict event outcomes. The difference, however, is that physical metric systems apply this modeling offline in order to set the alert thresholds, while performance metric systems apply this modeling online during operation to trigger alerts. The benefit to performing the modeling online is that situation-specific information can generally be included more directly than is possible during an offline evaluation, simply due to the large number of possible situations that could be encountered *a priori*. An analogy is determining chess moves before a game versus determining them online in response to the current situation.

Our intent is to illuminate those issues that are at the core of alert decision-making and how the design of systems may be more directly tied to those issues than has been the case in many current applications. Certainly there are applications for which the performance metric approach would not be effective or desirable. Yet, it is important to begin articulating the potential design options that exist, with the goal of generating a useful taxonomy of alerting system designs. We hope that efforts continue in the future across many disciplines to explore these and other methods when developing alerting systems.

REFERENCES

[1] E. M. Shank and K. M. Hollister, "A statistical risk assessment model for the precision runway monitor system," in *Proc. ATCA Conf.*, 1992.

[2] "Minimum Performance Specifications for TCAS Airborne Equipment," Radio Technical Committee on Aeronautics (RTCA), Washington, DC, Doc. RTCA/DO-185, 1983.

[3] W. P. Tanner Jr. and J. A. Swets, "A decision-making theory of visual detection," *Psychol. Rev.*, vol. 61, pp. 401–409, 1954.

[4] J. Swets and R. Pickett, *Evaluation of Diagnostic Systems*. New York: Academic, 1982.

[5] M. Barkat, *Signal Detection and Estimation*. Boston: Artech House, 1991.

[6] T. Sheridan, *Telerobotics, Automation and Human Supervisory Control*. Cambridge, MA: MIT Press, 1992.

[7] C. Wickens, *Engineering Psychology and Human Performance*. New York: Harper Collins, 1992.

[8] J. Kuchar and L. Yang, "A review of conflict detection and resolution modeling methods," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, pp. 179–189, Dec. 2000.

[9] W. E. Kelly, "Conflict detection and alerting for separation assurance systems," in *18th Digital Avionics Systems Conf.*, St. Louis, MO, Oct. 1999, pp. 6.D.1-1–6.D.1-8.

[10] "Application of Airborne Conflict Management: Detection, Prevention and Resolution," Radio Technical Committee on Aeronautics (RTCA), Washington, DC, Doc. RTCA/DO-263, 2000.

[11] J. Kuchar, "Methodology for alerting-system performance evaluation," *AIAA J. Guid., Navig., Dynam.*, vol. 19, no. 2, pp. 438–443, Mar.–Apr. 1996.

[12] L. F. Winder and J. K. Kuchar, "Generalized philosophy of alerting with application to parallel approach collision prevention," in *AIAA Guidance, Navigation and Control Conf.*, Montreal, QC, Canada, Aug. 6–9, 2001.

[13] L. F. Winder and J. Kuchar, "Evaluation of collision avoidance maneuvers for parallel approach," *AIAA J. Guid., Contr., Dynam.*, vol. 22, no. 6, pp. 801–807, 1999.

[14] P. Samanant, M. Jackson, C. Haissig, and B. Corwin, "CASPER/AILS: An integrated DGPS/ADS-B airborne alerting system for closely spaced parallel approaches," in *AIAA/IEEE PLANS Conf.*, Mar. 2000.

[15] M. Prandini, J. Hu, J. Lygeros, and S. Sastry, "A probabilistic approach to aircraft conflict detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, pp. 199–220, Dec. 2000.

[16] S. Haykin, *Neural Networks*. New York: Macmillan College, 1994.

[17] L. Yang, "Aircraft conflict analysis and real-time conflict probing using probabilistic trajectory modeling," Ph.D. dissertation, Dept. Aeron. Astron., Mass. Inst. Technol., Cambridge, 2000.

[18] L. Yang and J. Kuchar, "Prototype conflict alerting logic for free flight," *AIAA J. Guid., Contr., Dynam.*, vol. 20, no. 4, pp. 768–773, July–Aug. 1997.

[19] ——, "Using intent information in probabilistic conflict analysis," in *AIAA Guidance, Navigation and Control Conf.*, Boston, MA, Aug. 10–12, 1998.

[20] D. Isaacson and H. Erzberger, "Design of a conflict detection algorithm for the center/TRACON automation system," in *16th Digital Avionics Systems Conf.*, Irvine, CA, Oct. 26–30, 1997, pp. 9.3-1–9.3-9.

[21] V. Mellone and S. Frank, "Behavioral impact of TCAS II on the national air traffic control system," in *Seventh Int. Symp. Aviation Psychology*, Apr. 27, 1993.

[22] R. Ford, "The protected volume of airspace generated by an airborne collision avoidance system," *J. Navig.*, vol. 39, no. 2, pp. 139–158, 1986.

[23] ——, "The conflict resolution process for TCAS II and some simulation results," *J. Navig.*, vol. 40, no. 3, pp. 283–303, 1987.

[24] T. Williamson and N. Spencer, "Development and operation of the Traffic Alert and Collision Avoidance System," *Proc. IEEE*, vol. 77, Nov. 1989.

[25] C. Miller, T. Williamson, J. Walsh, L. Nivert, and J. Anderson, "Initiatives to improve TCAS-ATC compatibility," *J. ATC*, July–Sept. 1994.

[26] M. McLaughlin and A. Zeitlin, "Safety Study of TCAS II for Logic Version 6.04," U.S. Dept. Transportation, Rep.DOT/FAA/RD-92/22, 1992.

[27] A. Drumm, "Lincoln Laboratory Evaluation of TCAS II Logic Version 6.04a," MIT Lincoln Lab., Lexington, vol. I, 1996.

[28] D. Brudnicki, K. Lindsay, and A. McFarland, "Assessment of field trials, algorithmic performance and benefits of the User Request Evaluation Tool (URET) conflict probe," in *16th Digital Avionics Systems Conf.*, Irvine, CA, Oct. 26–30, 1997, pp. 9.3-35–9.3-44.

[29] W. Johnson, V. Battiste, S. Delzell, S. Holland, S. Belcher, and K. Jordan, "Development and demonstration of a prototype free flight cockpit display of traffic information," in *AIAA/SAE World Aviation Congr.*, Anaheim, CA, Oct. 13–16, 1997.

[30] P. Cashion, M. Macintosh, A. McGann, and S. Lozito, "A study of commercial flight crew self-separation," in *16th AIAA/IEEE Digital Avionics Systems Conf.*, Irvine, CA, Oct. 26–30, 1997.

Lee C. Yang received the B.S. degree from the University of Washington, Seattle, in 1989 and the S.M. and Ph.D. degrees from the Massachusetts Institute of Technology (MIT), Cambridge, in 1994 and 2000 respectively.

He is currently a Member of the Technical Staff, Charles Stark Draper Laboratory, Cambridge, and at MIT. His research includes modeling and simulation, dynamics and control, and vehicle conflict detection and resolution.

James K. Kuchar received the S.B., S.M., and Ph.D. degrees from the Department of Aeronautics and Astronautics, Massachusetts Institute of Technology (MIT), Cambridge, in 1990, 1991, and 1995, respectively.

He has been on the faculty at MIT since 1995, where he is currently an Associate Professor. He teaches courses on flight simulation, decision aiding and alerting systems, computational tools, and estimation. His research interests focus on information management, including decision aiding and alerting system design, air traffic conflict detection and resolution, safety assessment, modeling of humans and automation, advanced display concepts, and flight simulation.

Dr. Kuchar was awarded the William E. Jackson Award from RTCA and the Charley Wootan Award from the Council of University Transportation Centers in 1995 for his work on alerting system performance modeling. He has been active on several committees with the Society of Automotive Engineers and RTCA to develop industry standards for aircraft automation and recently served on a National Research Council committee to evaluate space launch range safety. He is a member of Sigma Xi and the American Institute of Aeronautics and Astronautics.