Fidelity Score for ATR Performance Modeling

Erik Blasch [1] , Eugene Lavely [2] and Tim Ross [1]
[1] Air Force Research Lab, Sensors Directorate, 2241 Avionics Cir, WPAFB, OH, 45433-7001
[2] BAE Systems, 6 New England Executive Park, Burlington, MA 01803

ABSTRACT

Automatic target recognition (ATR) performance modeling is dependent on model complexity, training data, and test analysis. In order to compare different ATR algorithms, we develop a fidelity score that characterizes the quality of different algorithms to meet real-world conditions. For instance, a higher fidelity ATR performance model (PM) is robust over many operating conditions (sensors, targets, environments). An ATR model that is run for one terrain, might not be applicable for all terrains, yet its operating manual clarifies its range of applicability. In this paper, we discuss a fidelity score that captures the performance application of ATR models and can be extended to different sensors over many operating conditions. The modeling quantification testing can be used as a fidelity score, validation metric, or guidance for model improvements. The goal is to provide a framework to instantiate a high fidelity model that captures theoretical, simulated, experimental, and real world data performance for use in a dynamic sensor manager.

Keywords: Performance modeling, Fidelity score, SAR, ATR

1. INTRODUCTION

The DOD community, like many (e.g. medical, economic, engineering, environment, and social), seek models that quantify causal relationships based on complex factors that afford decision making. Model development, as a process, has been researched since WWII; however the complex dynamic nature of the physical, technological, and social world requires validated updates to models to increase the model exploitation. The motivation for this paper is to focus on performance modeling, with specific issues addressing ATR model transition to operational use, as shown in Figure 1 Much of the formulation of the information follows from (1) the DoD modeling and simulation center [1], (2) statistics [1], and (3) ongoing COMPASE [2] center programs [2, 3].

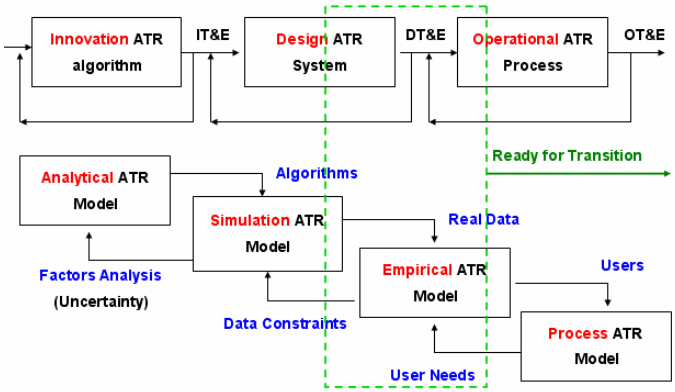

Figure 1. ATR Performance Model Cycle

The SEPerB (Sensor Exploitation, Performance and Behavioral Modeling) framework identifies model characteristics (analytical, simulated, or empirical), recommends performance metrics for each function or application, tests relevant operating conditions (OC) influencing performance, and quantifies the influence of any given OCs on these performance metrics [3]. Design of Experiments (DOE) [1] is one useful way to compare different performances; however, before ATR models can be tested, developers [4, 5] need to think of many real-world conditions to effectively produce robust algorithms [6].

In this paper, we characterize issues associated with validating a performance model with a fidelity score. Section 2 discusses the motivation. Section 3 lists issues in performance modeling. Section 4 examines a tool (Design of Experiments) to substantiate the fidelity score presented in Section 5. Section 6 gives examples and Section 7 draws conclusions.

[1] https://www.dmso.mil/public/
[2] COMprehensive Performance Assessment of Sensor Exploitation (COMPASE)

2. BACKGROUND

Waltz [7] presented a methodology for ATR test and evaluation. A system evaluation for command and control includes (1) analysis of technology and (2) targeting applications with metrics of: accuracy, resolution, and time delay. Issues associated with ATR comparisons include (1) ***Objectives*** – rank, quantify, qualify, (2) ***Alternatives*** – candidate based on sensed variables, and (3) ***Evaluation criteria*** – truth, utility, figures of merit. For an ATR model evaluation, parameters include constants (e.g. terrain), variables (e.g. sensor noise), and factors (e.g. target types).

ATR Performance modeling (ATR-PM) includes the sensor, ATR algorithm, and user(process) models, however the human models and metrics are not well defined [8]. To get appropriate PM, we need (1) *Math*: formal, symbolic, semantic, mental; (2) *Scenarios* – environment of operational complexity; (3) *Model fidelity* – geometry, measurement, accuracy; and (4) *Simulations* – scenario generator, models, controls, data base, communication and terrain issues.

A third major issue is Military Effectiveness through ***Figures of Merit***. Different metrics include (1) measures of effectiveness MOE(time), measures of performance MOP (P(Detection) and P(False alarm)), (3) Dimensionality (number of parameters), and (4) measures of Force Effectiveness, MOFE (cost). In order to instantiate ATR evaluations, we need to determine what types of models to test.

2.1. Modeling and Simulation

An ATR utility analysis requires understanding of user and ATR-PM. To explore the user-ATR-PM utility, adequate simulations need to be conducted to determine the interaction between the user and ATR system. From the *DoD Modeling & Simulation (M&S) Glossary*, there are defined terms for modeling and simulation. Physical and mathematical models can be developed to determine the ATR performance and process models can be used to determine the operational exploitation. Simulations are performed to assess the automatic and user ability within an ATR system.

Models

1. **Physical** - A physical model is a model whose physical parameters resemble the physical characteristics of the system being modeled. For an ATR system, there are both *static* components (such as target shape) and *dynamic* components of the target (i.e. speed and turning rate).
2. **Mathematical** - A mathematical model is a symbolic model whose properties are expressed in mathematical symbols and relationships. In the case of a static model, the target shape should be modeled (i.e. length and width) so that scaling and other functions can be invariant of the distance of the sensor to the target. Likewise, a dynamic model should include a mathematical relationship so that the acceleration, velocity, and position of the target are assessed. A good example is a sensor performance model.
3. **Process Models** - Process models are designed to replicate steps in a process or system. Process models allow users to define their tasks, work flows, or timing of events. Process models also include sensor management parameters of time, accuracy, and sensor steering to collect the information needs.
4. **Data Models** – Data models are sets of data collected from a sensor (e.g. IR images). The data could be the raw data or exploited data (i.e. radar returns processed to form a synthetic aperture radar (SAR) image). To determine an ATR performance model, it is important to understand the underlying characteristics of data and algorithms used to derive results.

Simulations

Simulations determine whether the ATR system is actually working in the manner in which they were designed as well as exploring extended operating conditions (EOCs). Determining the constraints over which the algorithm works are important to the development of the ATR-PM

1. **Constructive Simulation** - Constructive simulations are "simulations that involve simulated people operating simulated systems. Real people stimulate (make inputs) to such simulations, but are not involved in determining the outcomes." Constructive simulations require testing models in a robust manner such as a Monte Carlo analysis.
2. **Virtual Simulation** – A virtual simulation is "a simulation involving real people operating simulated systems. Virtual simulations inject human-in-the-loop in a central role by exercising motor control skills (e.g., flying an airplane), decision skills, or communication skills." Virtual Simulations include ground station simulators such as in a combat ATR ID

situations. There are user performance models which can determine the ability of process performance; however, levels of expertise and skill are needed to fully characterize the man-machine performance.

3. **Live Simulation** - Live simulations are defined as "simulations involving real people operating real systems." Field testing of the ATR system (in a cockpit and ground station) includes air to ground communications, collection times, and real world operating conditions.

Progressing from modeling to simulation to evaluation, it is important to determine the appropriate way to test ATR-PM process so that the models are well developed to meet real-world applications. Additionally, models and simulations need to be validated and verified. ***Validation*** includes ensuring that models simulate actual constructs designed with appropriate fidelity to foster efficacy in process and results. ***Verification*** is ensuring that the model was built according to the validated design. There are four ways to *verify* the system: (1) testing through measurement, (2) analysis with math, (3) inspection through documentation, and (4) demonstration by empirical evidence. To operationally use ATR systems, they must be validated and verified (V/V). We propose a fidelity score be appended to the ATR performance model that characterizes it limitations and expectations.

3. PERFORMANCE MODELS

3.1. Performance model overview

The SEPerB (Sensor Exploitation Performance/Behavioral) modeling concept defines and develops models across multiple sensor modalities and applications that affect past, current, and future systems. A key recognition of SEPerB is that there is no simple solution to the problem of performance modeling (PM), and that, instead, a comprehensive framework matched to the true complexity of PM is required. However, as Einstein is noted for fostering a parsimonious solution, we seek a complete and accurate simple model for pragmatic use. The world is complex, and as an operator makes decisions, we envision a set of models that reduce the dimensionality of complex interacting sensor systems.

The problem complexity follows from numerous considerations including

(i) the wide range or sensor modalities,
(ii) the domain-specific phenomenology and physics of each of these modalities,
(iii) the diversity and detailed nature of applications for data processing and interpretation from these modalities and
(iv) the diversity of scenarios and operating conditions in which these sensors and applications operate.

The model complexity derives from five fundamental areas (1) sensor, (2) target, (3) environment, (4) social, and (5) communication characteristics. Since the geopolitical nature is complex dependent, we first will focus on the first three, address the communication requirements, and leave the situation context for future analysis. Some problem areas are amenable to a detailed theoretical analysis (but often with strongly limiting assumptions to yield closed-form solutions). Other problem areas contain components that are modeled with high fidelity, but whose integrated high-level output depends on so many factors that simulation is the most practical route. Thus, instead of a "silver bullet" for performance modeling there must be a *spectrum* of PM solutions adaptive to the problem complexity and to the essentially infinite number of possible scenario and data realizations. Therefore, the objective of the SEPerB framework is to provide performance models for sensors, applications, and algorithms recognizing the ***complexity*** of this problem, for static environments, rational average users, and dynamic targets.

As an example of this complexity, consider an ATR system. ATR systems are composed of numerous inter-dependent processes. Typical processes include (i) data processing and feature extraction, (ii) feature prediction, (iii) score computation via a data misfit measure, data match metric, data assignment cost, etc. (iv) open-loop or closed-loop algorithms for optimal hypothesis discovery or other 'reasoning' methods, and others. Appropriate MOPs can be defined for each of these processes. Typically, a subset of the operative processes will be amenable to analytical performance analysis while others may be much more difficult to characterize. However, for these it may sometimes be instructive to approximate the process with a simpler or idealized proxy model that is amenable to analysis. Due to this complex *system* behavior (arising from interacting components) it is difficult to develop *ab initio* performance models for the top-level outputs of these system applications.

A SEPerB PM for an algorithm, processing chain, or tabulated results derived from an algorithm application to synthetic or real data. SEPerB models capture the behavior of sensor exploitation making them suitable for systems engineering; e.g., modeling subsystems in a systems integration activity, exploring human-machine or software component interfaces, or simply as a means of defining the complex functionalities of an exploitation system under development. SEPerB models are tools for user-design and execution of 'What If' experiments to support various applications.

3.2. Performance model applications

SEPerB models are designed to facilitate numerous applications. They can be used to assess the relative significance of operating conditions (OCs) that significantly impact ATR performance and provide sensitivity measures for them. Outputs from SEPerB models may be used to assist in designing data collection experiments, assessing application performance, verifying algorithms, and providing data inputs for algorithm development. They may also be used to drive improvements in application design. For example, in the case of ATR algorithm design, tunable design factors include feature selection and cost function formulation. SEPerB models may be directly used online in exploitation as a contributing component to applications such as integrated sensing and planning, sensor resource management, dynamic replanning and others. In addition, they may be used as a means of determining confidence levels in exploitation results as is necessary to support fusion and adaptation. SEPerB modes may also be used for general CONOPS development. Performance models may adopt numerous forms that we now describe.

With the above perspective, it may be argued that the SEPerB framework not only provides valuable input information for ***constructive simulation*** but also fills a critical gap between ***constructive simulation*** and ***virtual simulation***. The latter evolves a simulation in time using real applications. Real applications, in turn, often have a great deal of overhead (in terms of architectures, coding, detailed knowledge requirements, etc.) and the assumptions, inputs, parameters may not be transparent or easily accessible. These "facts on the ground" present a hurdle for performance modeling obscuring the underlying physical and algorithmic considerations that are the true performance drivers. The SEPerB framework attempts to address this by using and/or developing *proxy* applications and algorithms that capture the physical essence of the problem without the accompanying overhead. The physics-to-algorithm transparency not only serves the purpose of performance modeling, but promotes the analysis of new and existing algorithms, and facilitates the development of new applications. SEPerB will prove to be useful for prototyping applications as well as performance modeling, and their fidelity or validation assignments will enable users to evaluate their appropriateness for *real* use.

The SEPerB approach develops new models and attempts to incorporate available legacy models (where appropriate) that characterize the expected range of performance. The performance model range of operation through confidence intervals, will allow real users to know the most and least favorable outcomes. Both verified new and legacy models are placed within the ***validation hierarchy***. Legacy models will be modified to the extent possible to conform with the SEPerB usage and interface specification. These standards are posted for developers from other areas with an interest in placing their models in the SEPerB framework are invited to adhere to these standards to facilitate the process. This policy presents an important opportunity to increase the power of SEPerB as well as increase the utility of SEPerB to the ATR community. From an implementation perspective, this is enabled by placing SEPerB within an appropriate and well-documented software engineering setting and with interface document controls.

3.3. Performance model types

SEPerB models provide performance characterization and/or performance prediction for sensors, applications and supporting algorithms for these applications. Example *sensors* include SAR, GMTI, LADAR, IR and UGS. Example *applications* include automatic target recognition (ATR) [9], target tracking (kinematic and feature-aided) [10], image registration (image to image, image to model), and change detection. Example *algorithms* include feature extraction (for ATR), data association (for tracking), bundle adjustment (for image registration), and anomaly detectors (for change detection).

One class of SEPerB models is *theoretical* or *analytical* in which a performance metric y is given by a function $\boldsymbol{f}(\boldsymbol{x})$. For example, y may represent probability of correct classification, $\boldsymbol{x}$ is the corresponding vector of operating conditions and all other parameters that influence the value of y, and f is the theoretically derived functional form linking y and $\boldsymbol{x}$. Ideally, the derivation is physics and first-principles-based. As a SEPerB model, the assumptions used in the derivation and corresponding range of validity of the model must be identified. Further, the SEPerB validation level must be identified and the evidence used to instantiate that standard must be provided.

We may divide SEPerB *objects* into three major categories. These correspond broadly to ***products*** (e.g. models, databases, outputs of SEPerB applications, documentation, recommendations), ***applications*** (e.g. software codes used to produce or manipulate SEPerB products, auxiliary functions and/or utilities for producing data required for construction of SEPerB models), and finally ***frameworks*** (e.g. the environment in which SEPerB products and applications reside and operate). A key motivation for the design and implementation of the SEPerB framework and the supporting products and applications is the notion of reproducibility. SEPerB products are intended for use in a diverse array of applications some of which may be mission-critical. Therefore, an understanding of the origin and confidence-level of information inputs and/or model inputs is essential for assessing the utility and reliability the user application. SEPerB is designed from the outset to satisfy this requirement. However, the ultimate characterization of input model validity is the reproducibility of the model. Therefore, SEPerB models are designed to be reproducible and transparent. That is, all the tools and knowledge base used to produce a SEPerB model is provided in conjunction with the model itself. In this way, the user has the opportunity to perform their own validation of the model utility, and even to modify the tools/inputs used to create the model to satisfy the user-specific requirements.

SEPerB frameworks serve multiple purposes: synthetic data generation capability, data reduction via signal and image processing for use in various applications (e.g. image registration, automatic target recognition, change detection), application codes themselves for generation of results required for performance metric evaluation, methods and codes for computing performance metrics, and others.

SEPerB models are not just simply models, but may be either *outputs* or *components* of a performance modeling *system*. The SEPerB components are *dynamic* in that they (i) may share inputs/outputs, (ii) be combined in various ways to generate new data products, and (iii) are designed to be extensible to accommodate evolving needs. The model is a transfer function which relates inputs to outputs. The input parameters include: data, information and knowledge. Raw data can be combined into information and information, given context yields knowledge. Input examples include (1) data: SAR image, (2) information – fused tracks, and (3) knowledge – tracked target that is not observed. These varying inputs can be affected by noise, clutter, and anomalies which result from sensor, target, or environmental factors. The SEPerB outputs include (i) data streams necessary for performance model development or computation, (ii) on-the-fly performance metric valuations, and (iii) a static (but extensible) database of performance metric valuations. In this way, the SEPerB framework provides a library of functions that may be used as core components in the design, development or testing of numerous applications such as automatic target recognition, change detection, target tracking, image registration and others.

3.4. Preferred approaches to constructing SEPerB models

The basis for a model may have a more or less rigor, from mathematical or logical necessity, to physics based, to empirical with a large highly representative data set, to empirical with limited data, to simple engineering judgment. More rigor is better.

- SEPerB models should leverage other SEPerB models and their components where appropriate.
- SEPerB models capture the behavior of sensor exploitation making them suitable for systems engineering; e.g., modeling subsystems in a systems integration activity, exploring human-machine or software component interfaces, or simply as a means of defining the complex functionalities of an exploitation system under development.
- SEPerB models may generate tabulated results based on simulations. However, the simulations are based on well defined theory for the algorithms. This theory may be a proxy for a more complex theory or a full system implementation.
- SEPerB models are validated according to well-defined DOE criteria. There are a small number of validation levels - from completely *ad hoc* to rock solid. Instantiations of more descriptive terminology may be appended. The evidence for a given level of validation accompanies any SEPerB model in a fidelity score

4. DESIGN OF EXPERIMENTS (DOE)

The theory and implementation of *Design of Experiments* (DOE) is a useful approach for planning and analysis of tests in a structured and efficient way. The objective of DOE is to define a systematic experimentation approach for generating data adequate for deriving relevant information. DOE methodology is successfully applied to numerous fields including product design and simulation, prototype testing, performance testing, process development and optimization, and others [1]. One example design is *factorial designs* in which several factors are varied simultaneously according to a special prescription.

There are a number of advantages to this design approach compared to the more traditional one-factor-at-a-time approach including (1) the ability to detect and estimate interactions among the factors under study and (2) the possibility to reduce the number of tests necessary to obtain enough information.

A good DOE is intended to optimize information content, not just acquire a large amount of data. Information inference generally requires assumptions about the system that generated the data. Using these assumptions and a physical theory it may be possible to develop a mathematical model of the system. Ideally, a rigorous theory has functional forms and constants that are derived from first principles, and the experiment serves either to inform the development of the theory or to validate its predictions. Often however, there are unknown relationships and constants in a system models, and the goal of a data collection or experiment is to acquire data the required to estimate these unknowns.

Collecting the appropriate data is where issues of practicality emerge. The overwhelming advantage of a designed experiment is that the system under test is actively manipulated so that fewer data points need to be accumulated than with a passive approach, and increases the utility of the resulting data for inference of high quality information. A useful DOE addresses problems such as independence of data observations, effects of separation when variables are changed together, and determining a reliable model connecting system operating variables with system outputs.

Design of Experiments is an important academic area and there are numerous books [1], journal publications, and software applications[3] that are available. The Matlab statistics toolbox supports, for example, factorial designs, fractional factorial designs, response surface designs, D-optimal designs, and others.

In the SEPerB approach, the DOE methodology structures Monte Carlo simulation experiments in terms of the factors, and their levels that are varied. The objective is to optimize the resulting data set for ANOVA (analysis of variance), multivariate adaptive regression splines (MARS) which may be viewed as a generalized ANOVA, and behavioral model analysis. An application of DOE to SEPerB is a change detection simulation study. In this case, example factors include target size, aspect, RCS and sensor heading difference in multiple passes). The DOE methodology defines the minimum required combinations of factors and their levels for the simulations runs. The output from these runs and associated covariates represent a well chosen set of input-output pairs for subsequent analysis. The more factors included, the higher the fidelity score.

5. PERFORMANCE MODEL FIDELITY SCORE

There often exists a trade-off between ***fidelity*** or ***realism*** of the performance model prediction and the *complexity* of the process used to generate that prediction. The decision-point for model use will depend on the end-user requirements, and SEPerB models are designed to incorporate this flexibility. In the SEPerB framework, we propose a fidelity score that relates the quality of the model to the known instantiation of the model. There are many factors to include in the fidelity score, but we characterize them as parameter confidence, analytic complexity, and empirical verification/validation. Parameter confidence is related to the constants used in models that capture the behavior of the model. The analytic complexity is related to the amount of information captured by the model. Finally, the model has to be tested over real world operating conditions . The resulting score (which is somewhat subjective) should capture the corporate knowledge. An example would be

Analytic	Parameter	Empirical V/V	Fidelity Score
1	1	1	3
:	:	:	:
n	m	k	n + m + k

In this rating system {n + m + k} is the highest, 1 is the lowest and 0 means we know not enough information to comment. As an example, confusion matrices [2] have been used by many to capture the nature of an ATR algorithm. Based on the empirical development of the system, a 2 class problem would be { 0 1 0 1 } which demonstrates the weakness of the model. In a 20 class problem, the score might be {0 2 0 2} which still does not foster confidence for a user. If however, some analytic performance of clutter and environment were included in the analysis, followed by testing to achieve a ROC [11] of minimum performance, the score might be {1 2 2 5} which would suggest that the 20-class, analytic, V/V model might have

[3] See the MATLAB Design of Experiments tool http://www.mathworks.com/access/helpdesk/help/toolbox/stats/stats.html

some fidelity for an image analyst who must engage targets. The fidelity score is intended to be conservative so that the user community does not hold false expectations do to over-marketing of capabilities.

5.1. SEPerB defined model characteristics needed for operational readiness

If ATR developers are providing a performance model then there needs to be supporting evidence of how the model was developed. For a user to take advantage of an ATR performance, certain model characteristics are required,

What a SEPerB model is (required elements)
Title or name (for consistent reference) and version number or date
Owner (who produced or maintains the model)
Code (the actual software model itself)
Documentation, to include as a minimum
- Applicability (place the model in the overall sensor exploitation hierarchy)
- Inputs (including operating conditions)
- Outputs (including figures-of-merit)
- Functionality (what does the model do)
- Pedigree (model was developed from what physics / data sets / legacy models)
- Validation (level, history, organization)
- Users / Application directory (who all is using this model and for what kinds of applications)

For example:

(documentation) SEPerB models have self-documentation contained in the code
(documentation - applicability) SEPerB models should identify their place in an overall sensor exploitation taxonomy, e.g., be linked to a 'table of contents' or 'checklist' defining the modalities, applications and algorithms that are members of the SEPerB framework
(documentation - applicability) SEPerB models should specify assumptions, caveats, and ranges of applicability
(documentation – applicability) SEPerB models list sensor data, exploitation process, missions or scenarios, desired performance objectives and how accurate are the models outputs are
(documentation) SEPerB models should be installed in a web site and periodically upgraded. There will be numerous releases as capabilities are extended and/or improved.
(documentation - inputs) SEPerB models should identify OCs leveraging COMPASE foundation taxonomies [2]
(documentation - outputs) SEPerB models define performance metrics leveraging COMPASE foundation measures [2] and output an online characterization of the uncertainty of all performance metrics output by the model.
(documentation - pedigree) SEPerB models should leverage the physics governing the image, signal, or reduced data
(documentation - pedigree) SEPerB models should be as theoretical as possible. The theory may be implemented (coded) for numerical valuations.
(documentation - pedigree) A SEPerB model should be completely reproducible
(code) SEPerB code should be available as shareware.
(code) Where appropriate, SEPerB models should use extensible databases. They provide utilities (tools) to access and read data from these databases, as well as to add to them. The databases may contain sensor information, performance model information, target information, etc.
(code) SEPerB models should be linked via a common interface and format
(code) SEPerB models should provide a completely inclusive software implementation, set of data inputs, etc. That is, it should be useable without dependencies other than standard operating system utilities.

5.2. Validation definitions

There are at least three major criteria that we may identify that will define the ***utility*** of SEPerB to end-users. *First*, for any given application requiring performance evaluation (e.g. a simulation, a sensor management optimizer, experiment or CONOPS design, etc.) the SEPerB model ideally supports the sensor modalities, range of expected or allowable operating conditions. *Second*, for these modalities and conditions, the performance evaluation that the SEPerB model generates has been validated with respect to a well-defined validation standard. A key contribution of SEPerB is that these validation

standards are proposed and defined. As the fidelity of SEPerB models increase over time, the associated validation standard is expected to change accordingly. *Third*, for many SEPerB models, there exists a trade-off between the level of the validation standard and the complexity of the model. The user should have the flexibility to choose the point on this trade-off curve appropriate to their application. However, we should note that for many applications it will be possible to directly resolve the trade-off problem by appropriate design and use of off-line and on-line modeling.

The level of validation is determined by: (1) the type of model, (2) the amount, quality, and OC coverage of the "training" data (for model types 2-4) and "test" data (model 4-8), and independence of the validating process (tester and test data), shown in Table 1.

TABLE 1: Model types

Model Type	**Description**
1. Theoretical Model	Theoretical model derived from first principles with assumptions explicitly identified and corresponding range of applicability defined. Model maybe high-level measure of performance for complex application, or for just a component of an application. Range of validity demonstrated with appropriate numerical experiments (e.g. a Monte Carlo sim.).
2. Look-up Table from real data and real application	Look-up table for performance metric derived from real data and a real application for a selected corresponding set of operating conditions. Algorithms and data sources used to compute metrics all explicitly defined.
3. Look-up Table from synthetic data and real or proxy application	Look-up table for performance metric derived from synthetic data and a real or proxy application for a selected corresponding set of operating conditions. Algorithms and data sources used to compute metrics all explicitly defined. Use of bootstrap and extrapolated data can increase model performance by extending OC coverage
4. Behavioral Model fit to real or synthetic data (empirical)	Behavioral model derived from tabulated results. Such models are derived from statistical learning techniques and attempt to capture the underlying behavior of an output (e.g. a measure of performance) as a function of the underlying predictor variables (e.g. operating conditions. Such models are interpolation models and are validated using well-establish statistical methods such as cross-validation.
5. On-the-fly MOP synthesis from reduced data products (e.g. ATR scores)	As an example, confusion matrices may be generated for ATR scores. However, these matrices depend on the target list and not-in-library specification. It would be impractical to pre-compute all the possibilities. However, these can be synthesized in near real-time using pre-computed scores.
6. Tested	Field tested over multiple passes to ensure robustness
7. Cooperative	Includes a multi-sensor sweet and interactions to other systems
8. Adaptive	Varies sensor parameters do to changing conditions, such as warming up of chips

The validation levels are described in the following table. Each level of validation also requires the characteristics of all lower validation levels. "Good" training data or test data consists of a large quantity of highly representative data that fully covers the OC space claimed by the model. An OC space is "fully covered" if the data set might reasonably have resulted from a random sampling of the OC space based on a uniform distribution.

TABLE 2: Validation Level

Validation Level	**Description**
0 (un-validated)	The model contains all required elements of a Type 1 model
1 (functionality only)	A model that has been tested with a representative instance of each type of input data and produced all specified outputs without software failure (i.e., crashing).
2 (limited)	Some testing of the model has been conducted and the test results fall within the model's claimed accuracy for the tested conditions.
3 (standard)	A model of Type 1/2 or 3 with good training data that was tested with a good test data set, except of small quantity, or any model tested with a good test set.
4 (elite)	A Type 1 model tested with a real performance goals (model 5 or higher)

5.3. SEPerB relevance

The SEPerB method develops models and the supporting framework, but also provides a descriptions and assessment of the problem context to increase model fidelity. Examples of the latter include (i) identifying the relationship of SEPerB models to existing models, (ii) identifying complementary capabilities of SEPerB models to these existing models, and (iii) providing recommended next steps and upgrades relative to existing capabilities such as ease of use, as shown in Figure 2. These recommendations are taken to be general definitions of work that would be useful to perform, even if it cannot be done under the current SEPerB effort. This is in keeping with the view that SEPerB seeks to provide upgrades to existing tools, even if the upgrade is not the best possible, and to develop required new capabilities even if the increment is not the optimal that can be envisioned. While the SEPerB method seeks to develop capabilities of general utility, the direction of the effort is also strongly influenced by application priorities.

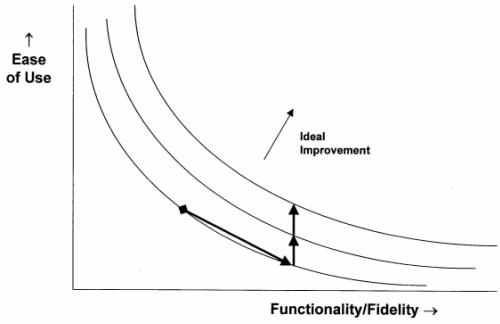

Figure 2. model improvement

The fidelity performance score for a SEPerB model depends on certain self-evident criteria. These include:

(1) the ***operation validation method*** and a clear understanding of how the validation relates to the real word,
(2) the ***predictive capability*** (i.e. the extent to which the performance space is covered),
(3) quantification of what ***model fidelity*** (how many OCs can be accommodated)
(4) ***model usability*** (i.e. how transparent is the model use to the non-expert),
(5) ***portability*** (i.e. what platforms does the software run on and does it require commercial products?) and
(6) ***features*** (e.g. to what extent are needed inputs, fusion paradigms, and integration of CONOPS to exploit the model)

Tradeoffs exist between the differing characteristics, however the fidelity score matched to the documentation would determine the model utility. For example, there is a trade-off in "fidelity" and "validity". Fidelity is about how good you claim a model is. Validity is about whether your claim is true or not. For example, it's easier to highly validate a low fidelity model. As a model increases in fidelity (i.e. more OCs), it is imperative to validate the claim through testing. Once the verification/validation test is completed, a fidelity score is attached to the model. As an example of a fidelity score, the levels below, shown in Table 3, indicate the fidelity of a validated model to robust performance in the field.

Table 3: Fidelity Score (lower is better)

	Sensor	Target	Environment	Example
0	**Analytic – Many OCs**	**Analytic - Many OCs**	**Analytic -Many OCs**	**Extended OC***
1	**Experimental**	**Experimental**	**Experimental**	**Exercise**
2	Experimental	Experimental	Simulated	Moving Collection
3	Experimental	Simulated	Simulated	Fixed Collection
4	Simulated	Experimental	Simulated	Fixed Sensor
5	Simulated	Analytic	Simulated	3D Sim. data
6	**Analytic**	**Experimental**	**Simulated**	**MSTAR**
7	Analytic	Simulated	Simulated	Stationary Sensor
8	Analytic	Analytic	Simulated	Matlab
9	Analytic	Analytic	Analytic	**Textbook**

* Extended OCs include the previous levels incorporated into a simulation environment based on analytical physical models, since not all real-world events happen in a single exercise.

As an example of further breakdowns, Table 4 shows Level 8.

TABLE 4: Level 8 Fidelity Score

	Sensor	Target	Environment	Example
8	Analytic	Simulated	Simulated	Stationary Sensor
8.1	10	20 + Target/variants	6 (forest, open, bldg)	Data Set
8.2	5 (range, power, angle)	20 Target/variants	6 (forest, open, bldg)	Data Set
8.3	5 (range, power, angle)	20 Target/variants	2 (forest, open)	Data Set
8.4	3 (range, power, angle)	20 Actual Targets	2 (forest, open)	Data Set
8.5	3 (range, power, angle)	5 Actual Targets	2 (forest, open)	Data Set
8.6	3 (range, power, angle)	5 Actual Targets	1	Data Set
8.7	3 (range, power, angle)	4 RCS	1	Textbook
8.8	3 (range, power, angle)	1 RCS	1	Textbook
8.9	1	1	1	Single Case

The model fidelity taxonomy is only a first cut for the ATR community to establish a V/V operational test and evaluation standard to deliver products to users. One method of increasing confidence in the fidelity score is its reproducibility.

5.4. Reproducibility

Performance modeling and assessment is a useful and necessary activity for a vast scope of algorithms and applications. Because of this, specific implementations of performance models or characterizations are highly diverse, and depend on the specific problem domain, algorithm, or application under consideration. In addition, performance models may be drawn from a continuum of possible models, all with varying levels of fidelity and validity. Therefore, in our view, a performance model will have the greatest utility if it satisfies two criteria. First, the performance model should be assignable to a model class using agreed upon standards (e.g. validation criteria). Second, the entire set of assumptions, data and scientific apparatus used to generate that model are provided in conjunction with the model itself (e.g. fidelity criteria). In other words, a performance model should be *reproducible*. To increase a fidelity score, we quantify the benefits of being reproducible. If validated models are reproducible, it would add considerable confidence to model utility.

Jon Claerbout, a well-known exploration geophysicist at Stanford University, has championed the concept of reproducible research in the computational sciences, and we believe his proposals are of value to the performance modeling community. More specifically, Claerbout is concerned with fields in which mathematical and numerical analysis may be applied to develop scientifically motivated signal processing and imaging problems, but for which mathematical analysis alone is not able to predict fully the behavior and suitability of algorithms for specific datasets. For this reason, experiments are necessary and such experiments ought, in principle, to be reproducible, just as experiments in other fields of science. Useful extended discussion of this topic can be found in Donoho *et al* [12] and Schwab *et al* [13].

As applied to performance modeling, the end-product should not just be the performance model itself, but instead the software environment that, applied in the right way, produces the model, and which, hopefully, could be applied to other datasets to produce similar models. In this case, the scientific findings are not just the model, but the knowledge of parameter settings for this complex software environment that led to the model. Therefore, reproducibility of data (models) requires having the complete software environment available for others to use, and the full source code available for inspection, modification, and application under varied parameter settings. Additional documentation includes

SEPerB Objects	Description	SEPerB Objects	Description
SEPerB Fidelity Score		SEPerB documentation	
SEPerB database		SEPerB references	
SEPerB package		SEPerB computing environment	
SEPerB model		SEPerB framework	
SEPerB function		SEPerB recommendations	
SEPerB application		SEPerB website	

6. EXAMPLE SEPERB PRODUCTS

6.1 Classification matrices

Classification or confusion [2] matrices will normally be defined with respect to a given algorithm such as a specific ATR. At the highest level these matrices will be organized by sensor type (e.g. EO, IR, SAR), sensor mode and exploitation method. Any given matrix will be associated with a set of OCs such as target (type, range, aspect), sensor (resolution), and environment (sand, camouflage). In computing these matrices, real or proxy ATRs may be used to generate the requisite ATR scores. The software required to synthesize these scores into confusion matrices is provided. This software is an example of a SEPerB *application*. At a higher level, the classification matrices resulting from hypothesis-level fusion may also be synthesized with respect to a set of well-defined assumptions.

6.2 Databases and information measures

One objective of the SEPerB effort is to identify theories and data products that are informative for performance modeling. One approach is to define the information content for any given data set relative to a given objective. Ideally, the information encodes all relevant variables including, for example, the sensor physics, target signatures, noise processes and CONOPS. For example, one may consider the distance between intrinsic target signatures (such as scattering centers in the case of SAR) across targets and views. This may have some utility but also has strong limitations due the fact that the corresponding measured data may have ambiguities. That is different sets of target signatures may produce nearly identical data observations leading to non-uniqueness in the estimation of the intrinsic target signatures. Therefore, a *more robust* information datum may simply be distance measures of data due to two different hypotheses. Noise processes will always be operative, and therefore probability density functions with respect for these distance measures may also be defined (as a function of the noise parameters). One may also simply generate PDFs consisting of scores. It then becomes useful to consider distance measures for PDFs, and the Kullback-Leibler distance (also known as cross entropy or relative entropy) is an obvious candidate [10].

Improved understanding of distances as in the components of a distance matrix can be obtained via multidimensional scaling (MDS). For example, d_{ij} may be the distance measure between two cost functions or two PDFs where the indices indicate the respective hypotheses. MDS seeks a lower-dimensional approximation of the data so as to preserve the pair-wise distances as well as possible. These distance metrics are appended to the classification matrices to increase model fidelity. Thus the performance model becomes more than just a matrix from a regression analysis, but affords decomposition and structural knowledge (i.e. sensor metrics) [14].

7. CONCLUSIONS

This paper overviews issues associated with detailing the ATR performance model quality that would support delivery of an ATR model from a developer to an operational user. In the proposed fidelity score, we include (a) a validated physical model, and (2) a verified set of usable documentation and reproducible code, and (3) a robustly tested performance model over many operational conditions. Future work will incorporate fidelity score into designed ATR models.

8. REFERENCES

[1] Kuehl, R., *Design of Experiments: Statistical Principles of Research Design & Analysis*, Duxbury Press; 1999.

[2] Ross, T.D., L. A. Westerkamp, R. L. Dilsavor, J. C. Mossing, "Performance Measures for Summarizing Confusion Matrices – The AFRL COMPASE Approach" in *Proc. of SPIE - 4727*, April 2002.

[3] Blasch, E., M. Pribilski, B. Daughtery, B. Roscoe, & J. Gunsett, "Fusion metrics for dynamic situation analysis", *Proc. SPIE* 2004.

[4] Lavely, E. M., "Feature association and occlusion model estimation for synthetic aperture radar" *Proc. SPIE 5427*, 2004.

[5] Blasch, E. P, J. J. Westerkamp, L. Hong, J. R. Layne, F. D. Garber, and A. K. Shaw, " Identifying moving HRR signatures with an ATR belief data association filter," *Proc. SPIE Int. Soc. Opt. Eng. 4053,* 2000.

[6] Pierson, W. E, Jr and T. D. Ross, "Automatic target recognition (ATR) evaluation theory: a survey" *Proc. SPIE 4053*, 2000.

[7] Waltz, E. L. and J. Llinas, *Multi-Sensor Data Fusion*, Artech House, Norwood, MA, 1990.

[8] Blasch, E. P. "Performance metrics for fusion evaluation", *Nat. Symp. On Sensor and Data Fusion*, 2003.

[9] Lavely, E. & P. Weichman, "Model-based and data-based approaches for ATR performance prediction," *Proc. SPIE. 5095*, 2003.

[10] Blasch, E. P, "Information-theory-based feature-aided tracking and identification algorithm for tracking moving and stationary targets through high turn maneuvers using fusion of SAR and HRR information," *Proc. SPIE*. 4727, 2002.
[11] Hanley, J.A. "Receiver Operating Characteristic (ROC) Curves", in *Ency. of Biostatistics*, Eds. P. Armitage & T. Colton, 1999.
[12] David Donoho, Mark Duncan, Xiangohmo Huo, Ofer Levi-Tsabari, *About Wavelab*
[13] Schwab, M.; Karrenbach, N.; & Claerbout, J; "Making scientific computations reproducible" *Computing in Science & Engineering*, Vol. 2, Issue 6, Nov.-Dec., pp.61-67, 2000.
[14] Ross, T.D., "ATR Theory Issues," in *Proc. of SPIE*, April 2004.