Next generation imager performance model

Brian Teaney and Joseph Reynolds

US Army RDECOM CERDEC

Night Vision and Electronic Sensors Directorate

ABSTRACT

The next generation of Army imager performance models is currently under development at NVESD. The aim of this new model is to provide a flexible and extensible engineering tool for system design which encapsulates all of the capabilities of the existing Night Vision model suite (NVThermIP, SSCamIP, etc) along with many new design tools and features including a more intuitive interface, the ability to perform trade studies, and a library of standard and user generated components. By combining the previous model architectures in one interface the new design is better suited to capture emerging technologies such as fusion and new sensor modalities. In this paper we will describe the general structure of the model and some of its current capabilities along with future development plans.

1. INTRODUCTION

The flow of signals through an imaging system provides information relating to system performance, Figure 1 traces some basic system parameters through a notional system.

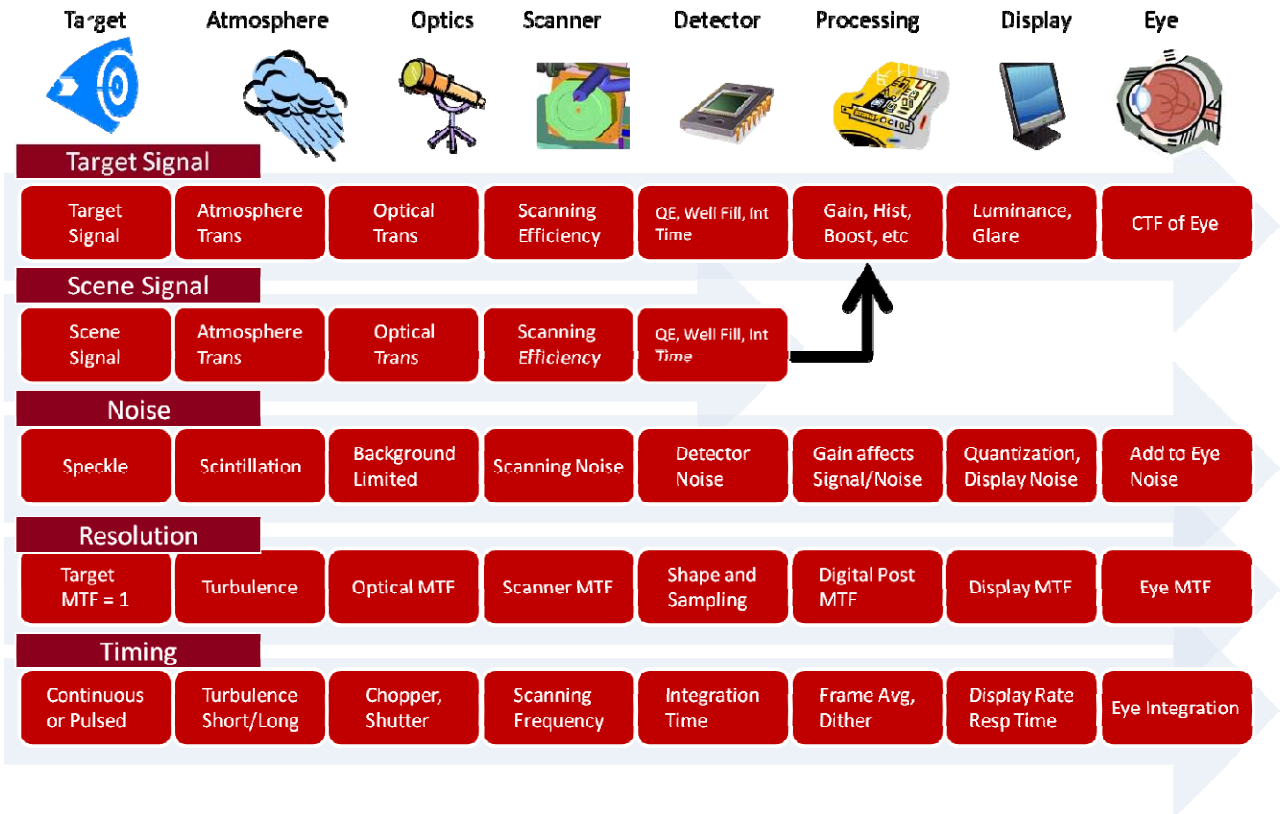


Figure 1: Illustration of signal flow in a system

For any system it is possible to determine specific values which contribute most significantly to the performance of the system. For instance, signal-to-noise may depend upon characteristics of both the scene and the imaging device. The

Infrared Imaging Systems: Design, Analysis, Modeling, and Testing XXI, edited by Gerald C. Holst, Keith A. Krapels,
Proc. of SPIE Vol. 7662, 76620F · © 2010 SPIE · CCC code: 0277-786X/10/$18 · doi: 10.1117/12.850876

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2010		**00-00-2010 to 00-00-2010**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Next generation imager performance model	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
US Army Research, Development, & Engineering Comd (RDECOM CERDEC),Night Vision and Electronic Sensors Directorate,Fort Belvoir,VA,22060-5806	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The next generation of Army imager performance models is currently under development at NVESD. The aim of this new model is to provide a flexible and extensible engineering tool for system design which encapsulates all of the capabilities of the existing Night Vision model suite (NVThermIP, SSCamIP, etc) along with many new design tools and features including a more intuitive interface, the ability to perform trade studies, and a library of standard and user generated components. By combining the previous model architectures in one interface the new design is better suited to capture emerging technologies such as fusion and new sensor modalities. In this paper we will describe the general structure of the model and some of its current capabilities along with future development plans.

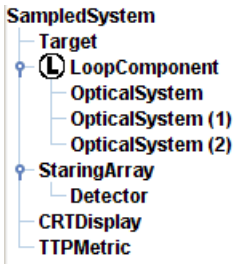
15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**Same as Report (SAR)**	**8**	

goal in developing this model is to provide an environment where the user can explore these relationships at any degree of complexity desired.

2. DESIGN

The following sections deal with a description of the model structure; Figure 2.a below shows a (basic) diagram representing the interaction between various levels of the model hierarchy. Each portion of the figure is discussed in more detail; note that portions of the model (graphics, etc) that do not directly impact the performance of its core function have been excluded from this discussion.



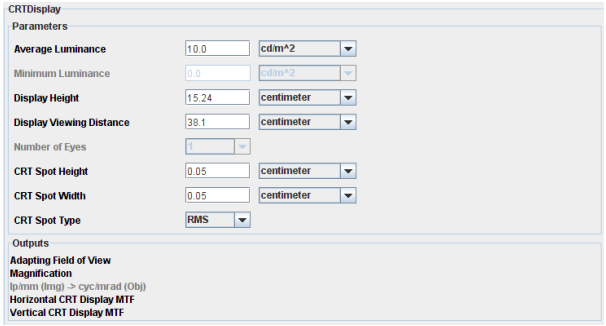
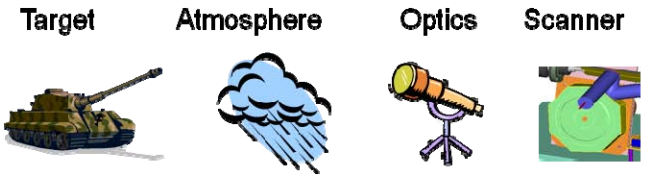


Figure 2: a. Example System Tree **b. Example component (CRT display)**

2.1. Components

A component captures the behavior of a particular element of a system. Figure 3.a gives broad examples of component definitions. From the generic components depicted in the figure more complicated representations can be constructed.



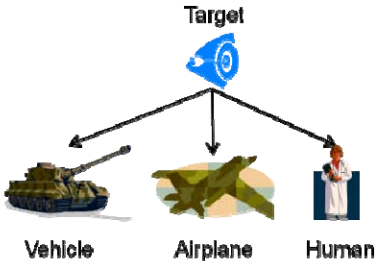


Figure 3: a. Examples of component definitions **b. Extensions of target**

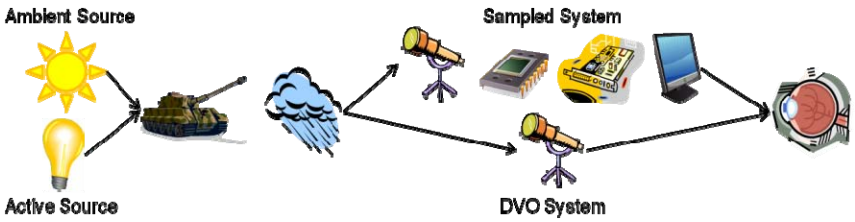
Consider for example extending the 'Target' component into several types of targets containing characteristics beyond the scope of the original object. These expanded representations of the original target component, shown in Figure 3.b, may each describe the intrinsic behavior of the target in a different way or expand upon the initial definition to provide additional functionality.

The components comprising the model are arranged in a tree structure; information (parameters and outputs) flows freely down the tree. Causality is enforced; if the tree were drawn as in Figure 3.a the atmosphere could request information from itself or the target but not the optics or scanner components.

Organizing the model in this way allows the user to include only the relevant portions of the system when evaluating a particular design. For instance, the user may only be interested in the flux at the aperture of the system, in this case only a description of the target and atmosphere (along with any other pertinent components) are required to execute the model.

2.1.1. Parallel Components

Parallel components are designed to provide multiple streams of data through the model. Figure 4.a shows two examples where parallel components could be used in the model.



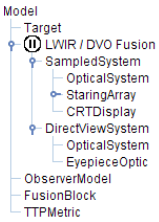


Figure 4: a. Examples of parallel components **b. Illustration of structure**

The left-most branch shows two sources of illumination – an ambient and an active source; these two sources exist simultaneously and both contribute to subsequent calculations. The behavior of signals (flux, etc) may be handled based upon pre-set rules; the function of these rules is discussed in detail in later sections. Signals may have multiple dimensions, for instance the active source described in Figure 4.a may have both spectral and spatial characteristics.

Branches may not be governed by pre-defined rules. Consider the second branch in Figure 4.a where the tree splits between a 'Sampled System' and a 'DVO System.' The behavior of subsequent components would be well defined in the case of one system or the other. These calculations would be executed in exactly the same fashion except that there will be a copy for each of the branches (two in this case). Eventually the model should reach a point which defines how the two sets of signals are to be combined. In the event where such a point is never reached the behavior of the parallel component will default back to that of a component trade space (see Section 4.1.2).

The system tree which defines the 'Sampled System' / 'DVO System' fusion is shown in Figure 4.b. In this case a component called 'Fusion Block' is defined which merges the parallel signals generated by the two systems. This means that any calculations which occur between the split and the fusion block which depends upon values generated in the system will perform multiple calculations (one set for each of the systems).

2.1.2. Organizational Components

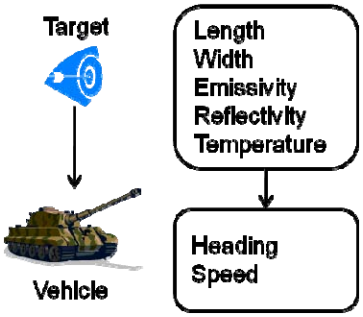
Organizational components are provided to create groupings of components which would otherwise be unassociated in the structure of the model. These additional layers provide a simple framework for saving portions of a system under a common architecture.

2.2. Parameters

Parameters describe the individual measurable attributes of a component and are input by the user to describe the specific system under evaluation.

Note that a parameter is not necessarily at the lowest level of description for a component. Depending upon the level which the author of the component has broken down the behavior, parameters may in fact be comprised of several lower level values which are either too numerous or too cumbersome to include.

To improve transparency only parameters which are needed by subsequent calculations are required inputs. These relationships are monitored as the component tree is constructed to provide the user immediate feedback relating to parameters which are required based upon the possible states of the model.



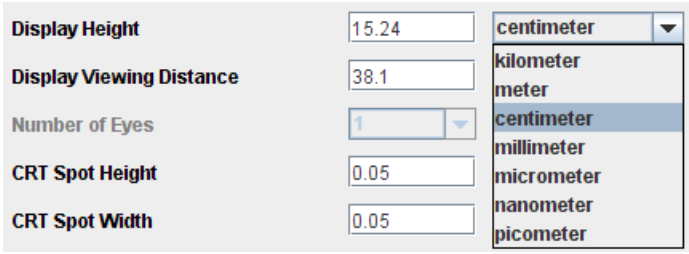


Figure 5: a. Target parameters **b. Example parameters (CRT Display)**

The units of a parameter can be broken down into two sub-types; units which are invariant to position in the system (length or temperature conversions remain constant in any system) and those which depend upon the system to provide a

context (spatial frequency may be referenced to object space, image space, eye space, etc). The requirement for a parameter used only in establishing the current context may vary depending upon the organization of the system and the requirement that the model operate in that particular context.

2.3. Outputs

Outputs are the results of calculations which may depend upon parameters or other output terms. Shown in Figure 6.a are some basic target outputs, each of these values has been described in a relationship which requires at least one input term. If this were not the case (the value could be calculated without any inputs) then the output would be a parameter.

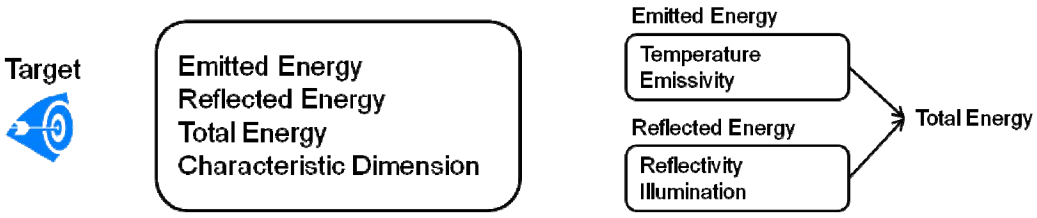


Figure 6: a. Target Outputs **b. Sample Calculation**

To illustrate how parameters and outputs interact Figure 6.b shows a simplified version of the total energy calculation. The output term total energy requires two inputs, emitted energy and reflected energy. Emitted energy is governed by familiar physical relationships and depends in turn upon the temperature and emissivity of the target. Reflected energy depends upon the reflectivity of the target and the illumination present at the target. Recalling that Figures 5 and 6 did not contain a parameter or output value for 'illumination' we conclude that the value for illumination must come from elsewhere in the model.

Each output in the model can be overridden; subsequent requests for the output would utilize the user provided value. Outputs may also be included as entries in parameter trade spaces, discussed in further detail in Section 4.1.1, in such a case the user may create a case where both calculated and provided values are used in successive iterations.

2.3.1. Contexts

A major concern in designing the model was enforcing the ordering of components such that the final system made physical sense. The notion of contexts was developed to serve a dual role of enforcing component ordering and enhancing model flexibility. Contexts are developed based upon the structure of the component tree; each component may have a local (default) unit for each context and can define any number of conversions within its scope.

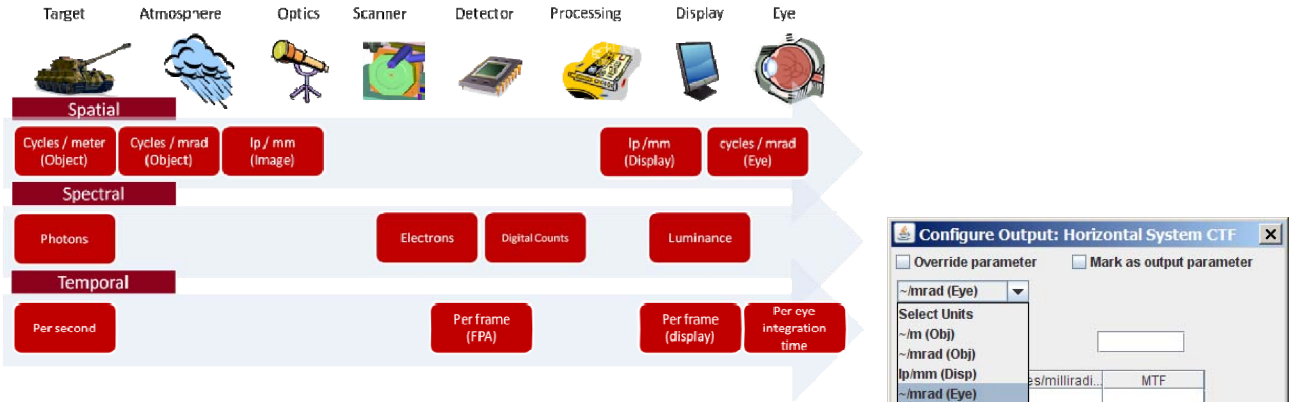


Figure 7: a. Example Contexts **b. Context based unit selection**

Figure 7.a shows the evolution of three notional contexts through a scanning-sampled imager. The ordering of components is enforced via these contexts; each context which is added to the system will tighten the ordering requirements.

First, contexts enforce the ordering of components by placing constraints upon the units entering and exiting a particular component. Once a component introduces a conversion (say from cycles/mrad to lp/mm in the Optical system

from the example above) the incoming required units are fixed and the model will not execute until all of the conversions line up properly.

In addition, a component may have a local context which must match the incoming (most recent) units. For example the detector may not define a conversion with respect to the spatial context but there is an implied dependence upon incoming units of lp/mm at the image plane. In this case placing a detector before the optical system where the incoming units do not match the local units would prevent the model from executing.

Contexts also define the behavior of generic components in the model. A generic component by definition has neither a conversion nor a local context. In this case the context (available units) is filled in based upon the location of the generic component in the tree.

2.3.2. Requests

To best track calculation requirements each output retains a list of parameter requests. These requests contain the specific details about how the parameter is to be used in the calculation (required units, etc) and a list of parameter references which will be used at various points throughout the model execution. Depending upon the current model state the request will have either a specific parameter value to use in the next iteration or instructions to search for the appropriate parameter reference.

3. FEATURES

In addition to a more flexible structure the new model incorporates many new features which facilitate system design.

3.1. Trade studies

Trade studies are the ability to compare any number of combinations of parameters and components to determine an effect on system performance. There are two types of trades present in the model currently, parameter trades and component trades.

3.1.1. Parameter Trades

Parameter trades allow the user to select groups of parameters from any component or set of components in the system to vary simultaneously. The parameter trade interface is shown in Figure 8. Because parameters in a trade space may come from various components the values are collected into a common list and a request to edit a particular entry will open the full list of parameters in the trade.

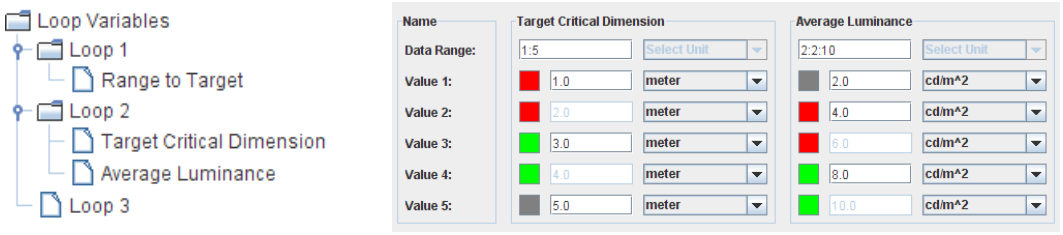


Figure 8: Parameter Trades

In the example shown the model contains two parameter loops, the first trade space contains range to target and the second trade space varies target critical dimension and average luminance simultaneously.

The second parameter trade space is shown in more detail; we have five values over which we wish to vary target critical dimension and average luminance. The user may opt to enter a value for each entry in the parameter trade space. However, in order to speed up the input process within a parameter loop, the user may optionally create up to four color-coded sub-loops within each column of data. These sub-loops are provided to simplify the input process; the model will automatically optimize around identical parameter values prior to execution.

During the execution of the model a parameter trade will notify any associated requests each time the current parameter values is changed. These changes propagate through the system to produce the new model state. From this

new state each of the outputs is queried to determine whether or not the value is re-calculated based upon the change in state; by only recalculating values which directly depend upon a particular input the model is able to execute much more quickly.

3.1.2. Component Trades

Component loops allow the user to vary entire components in the system. The user may wish to compare the performance of similar components where organization using parameter trade spaces would be cumbersome or to compare entirely different sensor modalities.

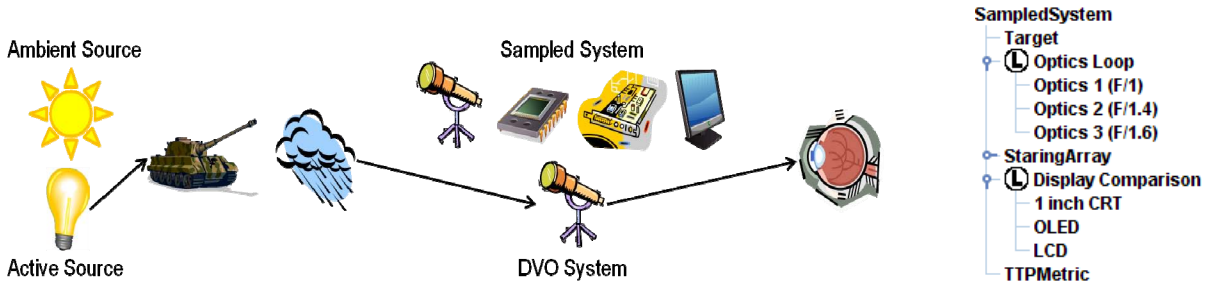


Figure 9: a. Component Trades **b. Tree Illustration**

Figure 9.a shows an illustration of a single path through a system containing looped components. In this case both the sources and the systems exist as component trade spaces.

Shown in Figure 9.b is a 'Sampled System' with component trade spaces as it would appear in the model. In this particular example there are two component trade spaces present. The first trade space compares the effect of varying the optical system upon the desired outputs. The second compares the effect of three different display types upon the desired outputs. If any of the desired outputs depend upon the optical system and the display then the model will produce nine values for that particular output.

Similar to the parameter trade the component trade notifies associated requests each time the state of the model changes. Each of the parameters associated with the out-going component notify their associated requests that the reference they have is no longer valid. Next the parameters of the incoming component notify their associated request that they should update references. Again, only calculations depending on a value changed during the component switch are reevaluated.

3.1.3. Optimized Loops

In order to minimize the execution time of the model the loops are prioritized based upon the complexity of incrementing a particular loop. The first step in optimizing the loop ordering is to optimize the elements inside each parameter trade space and each component trade space. Then, using the average complexity of each parameter trade space and component trade space we can proceed to optimize the entire structure.

The first step in optimizing the ordering of a trade space is to determine the parameters whose order can be optimized. Any parameter whose value at a particular point has multiplicity greater than one is a candidate for optimization. The ordering inside the space of parameters containing only points of multiplicity one are ignored during the optimization process.

To optimize both the parameter and component trade spaces the number of operations associated with switching a particular value is determined. Depending upon the parameters which comprise the trade space there may be some commonality between the required calculations, these overlapping values are neglected in evaluating the total associated cost as the complexity of the trade space itself is invariant to the order the elements are traversed in these cases.

Note that when a component trade space or a parameter trade space which includes an output is encountered when traversing that the average complexity over the trade space is added to the total complexity. For the case where the parameter space contains an output the average complexity is the number of times where the calculation is actually executed divided by the total number of elements in the space. Subsequent calculations which are only executed due to the variation of the output are averaged in a similar fashion.

Having determined the most complex candidate parameters the trade space is divided into subspaces which are each associated with one of the values assumed by the candidate. This division of the subspaces continues for each of the candidate parameters. The final step in optimizing an individual trade space is to order the subspaces (at every level) to exploit common values on the boundaries.

The optimization of component trade spaces follows a very similar process. However, the nature of component trade spaces only guarantees that the components share a common parent at some level of the library hierarchy not that the components are identical in every respect. Because the components share a common ancestry the optimization is performed based upon the parameters (and outputs) which are included in the common parent. From here the determination of candidate parameters, subspaces and subspace ordering can be performed.

Once the optimizations of the trade spaces are complete the model proceeds with reordering of the trade spaces to minimize the total execution time.

3.2. Custom Components

To make the model as flexible as possible the option to introduce custom components from a variety of sources is included. Users may provide a file containing a description of the parameters and outputs of a component which is incorporated into a library of components available at runtime. Users may opt to construct custom components which expand the existing structure or create entirely new branches within the model. For instance, the user may opt to create an entirely new type of display to model an emerging technology or expand upon existing components to enhance model functionality.

3.2.1. Component Libraries

The model includes a library of default components which encompass the functionality of the previous NVESD models. When constructing new components users are given the opportunity to expand the base library and create new libraries entirely. For example, a collection of displays or focal plane arrays could be created and exported for easy use in future modeling efforts.

3.2.2. Core Routines

A collection of common calculation routines, NV Core, is available for use in constructing these custom components. These additional routines vary in their scope and application; common MTF shapes, integration routines, and FFT calculations are all included. Using this library aids in reducing the likelihood of typographical errors and streamlines the component authoring process.

3.3. Selectable outputs

Prior to running the model the user must select all desired outputs. During execution only calculations which are required by outputs are actually executed. For instance, if the user wished to only look at the target signal compared to the path radiance contribution at the aperture then all calculations which occurred after the optical system would not be executed since they have no bearing upon the desired output(s); and, because only required calculations are performed the model can execute much more quickly. Further, the addition of these selectable outputs allows the user more control over model execution. Rather than a large list of outputs only those the user has deemed relevant are produced.

4. FUTURE PLANS

Beyond the capabilities described above additional features are under development.

4.1. Internal Plotting

Once the model has been executed the user may opt to plot results inside the model structure. In order to provide the most flexibility the abscissa and ordinate may be selected from any of the looped values which have a bearing on the output and multiple series may be included based upon remaining outputs.

4.2. Report Generation

Users would be able to describe a ‘typical’ output file which could be generated each time the model is run. The model could be programmed to cull an existing model tree to select certain desired outputs.

4.3. Optimization

Preliminary work indicates that the inclusion of rudimentary optimization of parameters based upon a user defined cost function(al) is possible. Early attempts focused at the maximization of range performance with respect to varied viewing distance and sensor gain (both as independent parameters and considered in tandem).

4.4. Geometry Module

To provide additional flexibility a geometry module has been proposed which would act as a pre-processing step. The geometry module would allow the user to exercise considerably more control over the placement objects in the scene.

5. CLOSING REMARKS

The next generation of Army imager performance models is intended to provide a more useful engineering tool to the EO/IR community. The new model incorporates many new design features including: the ability to create and store component libraries, trade studies and a more flexible user interface. In developing this new interface we intend to combine the existing Night Vision models under a common architecture. If the reader has any suggestions for further improvements please feel free to submit any comments or suggestions to (nvlmodels@nvl.army.mil).