

PROCEEDINGS OF SPIE

[SPIDigitalLibrary.org/conference-proceedings-of-spie](https://spiedigitallibrary.org/conference-proceedings-of-spie)

Performance prediction from EO system measurements using IRWindows and NV-IPM

Alan Irwin

Alan Irwin, "Performance prediction from EO system measurements using IRWindows and NV-IPM," Proc. SPIE 10625, Infrared Imaging Systems: Design, Analysis, Modeling, and Testing XXIX, 1062502 (26 April 2018); doi: 10.1117/12.2315492

SPIE.

Event: SPIE Defense + Security, 2018, Orlando, Florida, United States

Performance prediction from EO system measurements using IRWindows and NV-IPM

Alan Irwin

Santa Barbara Infrared, Inc., 30 S. Calle Cesar Chavez, Suite D, Santa Barbara, CA, USA 93101

ABSTRACT

The US Army Night Vision and Electronic Sensors Directorate (NVESD) has developed the Night Vision Integrated Performance Model (NV-IPM) for conducting system trade studies and performance evaluations for EOIR imaging systems. Many programs of record carry range performance requirements that utilize the targeting task performance (TTP) metric together with system level objective measurements of an imager. The imaging system measurements of signal intensity transfer function (SITF), 3-dimensional noise (3D Noise), instantaneous field of view (IFOV), and modulation transfer function (MTF) are combined within the measured system component that can be directly implemented in NVIPM for performance and specification evaluation. IRWindows 4TM (IRWindows) is a software package produced by Santa Barbara Infrared, Inc. (SBIR) and is used for testing electro optical systems in a variety of laboratory, production, and field environments. In this correspondence, we detail how IRWindows performs the required measurements for TTP evaluation and generates a measured system component for use in NVIPM to predict range performance of an IR imaging system. Further, we demonstrate how the range performance from system measurements is in agreement between different EOIR laboratories.

Keywords: NVIPM, IRWindows, testing, TTP, modeling

1. INTRODUCTION

The Night Vision Integrated Performance Model (NV-IPM) was developed by the US Army's Night Vision and Electronic Systems Directorate (NVESD), and it combines and supersedes their previous generations of performance models, including NVThermIP, SSCamIP, and IINVD. It provides an easy-to-use and flexible user interface for developing performance models, including detailed target and background definitions, atmospheric and transmission characteristics, detailed sensor system definitions, observer models, and performance metrics¹. The encapsulation of performance parameters and algorithms into independent objects creates an environment in which a system engineer or scientist can quickly create a model to predict system performance from a complex set of environmental, system, and observer models. Additionally, the dependencies of system performance on critical parameters can be easily characterized by using built-in tools to iterate across those parameters. The results are sensitivity graphs that can determine optimum parameter values².

Design and early characterization of EO sensor systems are areas where modeling has been particularly useful, in that the measured system-level characteristics of an imager can be used to predict the performance of a real-world system in modeled environments. These system level measurements can replace the discrete values in a parametric model and calculate performance predictions.

NV-IPM implements a Measured System Component that can store the results of a set of imager tests, and then use that component to define the imager in a larger performance model. The format and contents of this component are publically available³.

IRWindows4TM (IRWindows) is a software platform developed by Santa Barbara Infrared, Inc (SBIR). for automating the testing of EO systems^{4,5}. IRWindows automates measurements of a unit under test (UUT) by presenting calibrated stimulation to the UUT, collecting response data, analyzing the data, storing results, and generating/displaying reports. This SW platform, coupled with the necessary EO stimulating sources and (in most cases) appropriate projection optics, is deployed in laboratory environments, production floors, and repair depots. It supports high speed production testing, detailed first article characterization and analysis, as well as simple go/no-go analysis in a robust and flexible environment. Development and analysis can be performed locally or on independent test stations.

Traditionally, these measurements produce system-level metrics of the imager's performance, typically independent of both the device's operational environment and specific mission.

SBIR has had a long-standing interest in implementing model-based capabilities into IRWindows to enable predicting the performance of a measured system under its intended operational conditions. In other words, an actual *performance* measurement⁶.

This paper explores the initial results of integrating the NV-IPM modeling tool into IRWindows. We focused primarily on data transfer between the two systems rather than a thorough analysis of the devices being tested. That will be the focus of future work and will take advantage of this effort.

The results of this work serves as a practical guide for anyone interested in putting test results into an NV-IPM model.

2. FORMATS & INTERFACES

2.1 Measurement Component

The key part of interfacing with NV- IPM is the measurement system component (MSC). As an independent file, the MSC is basically an XML file (see partial example in Figure 1).

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE nv-ipm SYSTEM "nv-ipm.dtd">
- <Model>
  - <Component>
    <Class>org.NVESD.NVCore.DefaultLibrary.MeasuredSystem</Class>
    <Name>Test Case</Name>
    <Author>Alan Irwin</Author>
    <Company>SBIR</Company>
    <MetaText>Created 4/5/2018 at 2:18 PM</MetaText>
  - <Parameters>
    - <Parameter>
      <Name>Horizontal Detector Count</Name>
      - <DoubleParameterEntry>
        <Value>1024</Value>
        <Units/>
      </DoubleParameterEntry>
    </Parameter>
    - <Parameter>
      <Name>Vertical Detector Count</Name>
      - <DoubleParameterEntry>
        <Value>850</Value>
        <Units/>
      </DoubleParameterEntry>
    </Parameter>
    - <Parameter>
      <Name>Horizontal Field of View</Name>
      - <DoubleParameterEntry>
        <Value>24</Value>
        <Units>degrees</Units>
      </DoubleParameterEntry>
    </Parameter>
    - <Parameter>
      <Name>Vertical Field of View</Name>
      - <DoubleParameterEntry>
        <Value>18</Value>
        <Units>degrees</Units>
      </DoubleParameterEntry>
    </Parameter>
```

Figure 1. File structure of the MSC. Only the initial few parameters are shown. The indenting is part of the display since there are no leading blanks allowed in the file. The file name is **IRWindows Test.xml**

There are 15 parameters in the input parameter section, and 4 parameters in an override section (see Table 1).

Table 1. Parameters used in the MSC

Parameter name	Section	Type	Units
Horizontal Detector Count	Parameters	Double	none
Vertical Detector Count	Parameters	Double	none
Horizontal Field of View	Parameters	Double	degrees
Vertical Field of View	Parameters	Double	degrees
Horizontal Detector Pitch	Parameters	Double	micrometer
Vertical Detector Pitch	Parameters	Double	micrometer
Sigma TVH	Parameters	Double	Kelvin
Sigma VH	Parameters	Double	Kelvin
Sigma V	Parameters	Double	Kelvin
Sigma H	Parameters	Double	Kelvin
Display Contrast	Parameters	Double	none
Frame Rate	Parameters	Double	none
Horizontal Pre-Sample MTF	Parameters	Array	cycles/pixel
Vertical Pre-Sample MTF	Parameters	Array	cycles/pixel
Normalized Response	Parameters	Array	Normalized Response
HSpatial: cycles/mrad -> cycles/pixel	Overrides	Double	cycles/mrad
VSpatial: cycles/mrad -> cycles/pixel	Overrides	Double	cycles/mrad
Horizontal Sample Frequency	Overrides	Double	cycles/pixel
Vertical Sample Frequency	Overrides	Double	cycles/pixel

IRWindows maintains a library of UUT definitions based on a customer's product line or parts definition. Tests can be configured in as independent modules using UUT parameters as well as test specific conditions. These test configurations can be run independently or arranged in sequences based on the needs and hardware available at a test station. Results from these tests are stored in a database which is available to other tests or reporting functions.

A translation module was developed in the IRWindows programmer environment (the core engine did not need to be modified) that collects the results from specified tests as well as parameters from a defined UUT and stores them into the formatted XML file of an MSC. The required parameter values can be collected from measurements, from the UUT definition, or specifically called out as an override. This IRWindows translation module can be run independently or added to any test sequence and run at any point in the overall sequencing of tests.

The NV-IPM measured component can be created into a directory used by NV-IPM, and is then available to the modeling software. The component can be used to build a performance model.

2.2 Model

We then extended the IRWindows translation module with the ability to insert the MSC into an existing NV-IPM model file. This allows the researcher to create a model in advance, defining the environment and observer elements into which the MSC will be placed for analysis. The opening portion of a typical model file is shown in **Figure 2**. The basic operation of the translation module is as described in the previous section, but the module is directed to add the component into an existing file rather than create a new MSC file. The IRWindows translation module searches the

specified file for an MSC (class: **org.NVESD.NVCore.DefaultLibrary.MeasuredSystem**) and replaces the parameter values with the data set that has just been collected.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE nv-ipm SYSTEM "nv-ipm.dtd">
- <Model>
  - <Component>
    <Class>org.NVESD.NVCore.DefaultLibrary.Model</Class>
    <Name>Working test model</Name>
    <Author>Default Author</Author>
    <Company>Default Company</Company>
    <MetaText/>
  - <Sub-Components>
    - <Component>
      <Class>org.NVESD.NVCore.ParallelComponent</Class>
      <Name>Target / Background</Name>
      <Author>Default Author</Author>
      <Company>Default Company</Company>
      <MetaText/>
    - <Sub-Components>
      - <Component>
        <Class>org.NVESD.NVCore.DefaultLibrary.Target</Class>
        <Name>Target</Name>
        <Author>Default Author</Author>
        <Company>Default Company</Company>
        <MetaText/>
      - <Parameters>
        - <Parameter>
          <Name>Target Reflectivity</Name>
          - <ArrayParameterEntry>
            <Units>Reflectivity</Units>
            <Value>0.4,0.5,0.75,0.9,1.2</Value>
            <Value>1.0,1.0,1.0,1.0,1.0</Value>
          </ArrayParameterEntry>
        </Parameter>
      </Parameters>
    </Component>
  </Sub-Components>
</Component>
</Model>
```

Figure 2. File structure of a model. Only the initial part of the file is shown. The MSC is deep within the file structure. Note that the indenting is part of the display since there are no leading blanks allowed in the file. The file name is **Working test model.xml**

NV-IPM can then be used to directly run the model without any further data entry.

2.3 Command line

In addition to the data transfer available through the MSC, NV-IPM provides a command line interface which can be used to automate the execution of a model. From a single command line, the core processing of the NV-IPM can be started, loaded with a specified model file, the model is executed, and the results stored in a set of output files.

Once more, the IRWindows translation module was expanded to give a developer the option to automatically use the command line interface. The specified NV-IPM model file is loaded with the measured data, and then NV-IPM is executed on that model file. The stored results can be used by the developer directly, or they can be retrieved by IRWindows for storage, further analysis, and display.

3. APPLICATION

3.1 IRWindows

A sample camera was used to test the IRWindows interface into NV-IPM. Specifically, the UUT was an IRCameras IRC806 with pour-filled Dewar assembly, using an SLS sensor, and filtered for LWIR (see Figure 3). The basic specifications are summarized in Figure 4, which is the UUT definition used in the IRWindows side of the application.



Figure 3. IRCameras IRC800 SLS LWIR with pour filled Dewar assembly

Configuration	
Frame Capture	IRC Camera SDK
Center of Image	228, 224 pixel
Fixed	
IRDetector	SLS
SensorBand	LWIR
Horizontal Sensor Count	456
Vertical Sensor Count	448
Pitch	20.0 μm
Fill Factor	0.95
Frame Rate	60.0 Hz
Integration Time	0.6 ms
Radiometric Normalization Temperature	298.0 K
FNumber	1.4
Quantum Efficiency	4 points $\mu\text{m frac}$
Nominal	
Focal Length	25.0 mm

Figure 4. UUT definition of the IRC806 camera used in the test applications

An SBIR laboratory system configured for both focused image testing and flat field testing was used to perform the system measurements on this camera (see Figure 5). For tests requiring flat-field/flood IR illumination, the camera is positioned to directly view an 8-inch extended area blackbody, while for focused imaging tests, the camera views an appropriate target pattern (back-illuminated by a smaller blackbody source) through a 6-inch aperture/30-inch EFL reflective collimator.

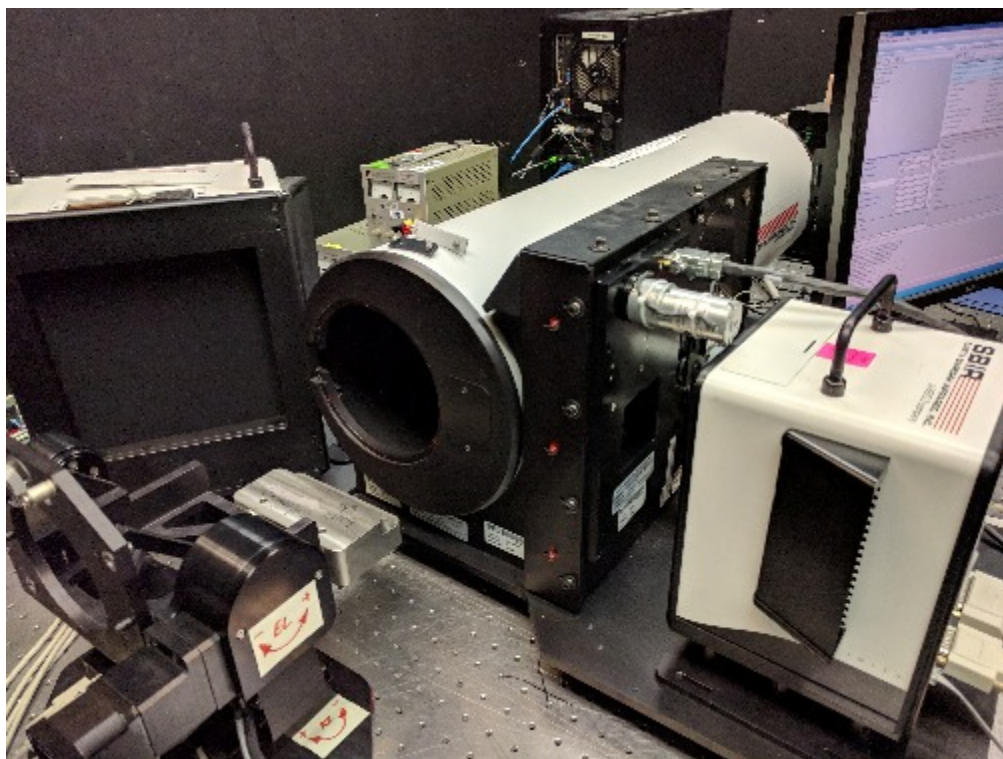


Figure 5. Laboratory test configuration. Two-axis rotating camera mount assembly at lower left of photo permits UUT to directly view 8-inch extended-area blackbody source (upper left) or collimating target projector (center), as appropriate for each system test measurement.

Three tests, with accompanying test configurations, were defined for measuring the system performance of the camera: SiTF, 3D Noise, and MTF (using the ISO12233-2014 methodology). The test parameters selected were based on the extensive use of this camera in-house (see Figure 6).

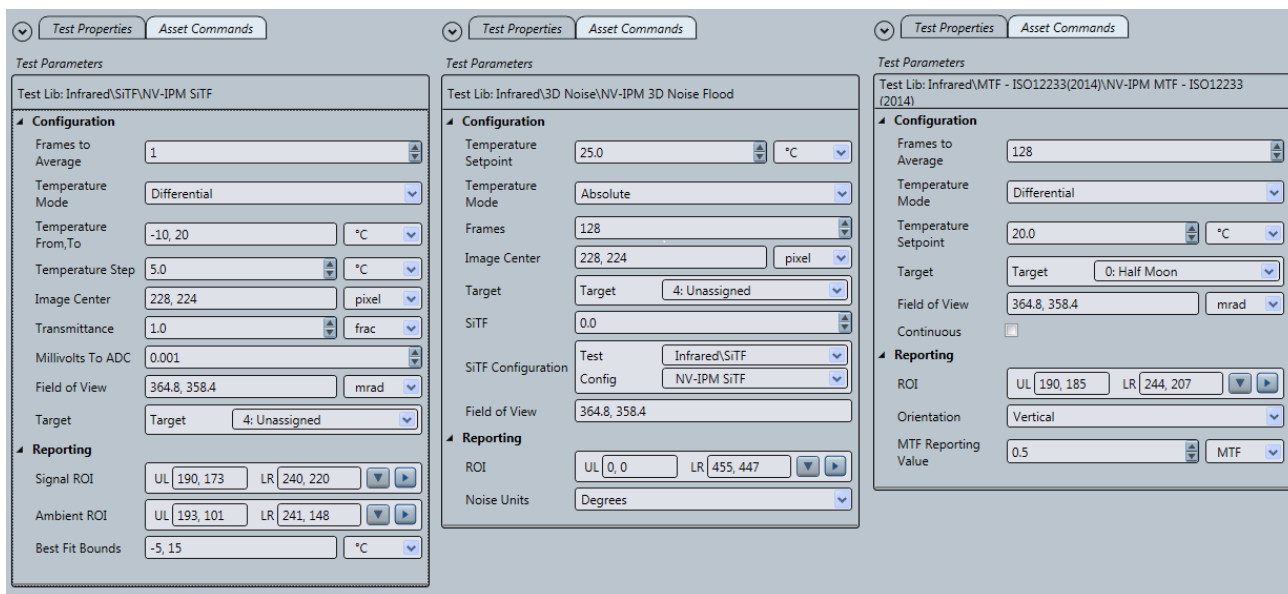
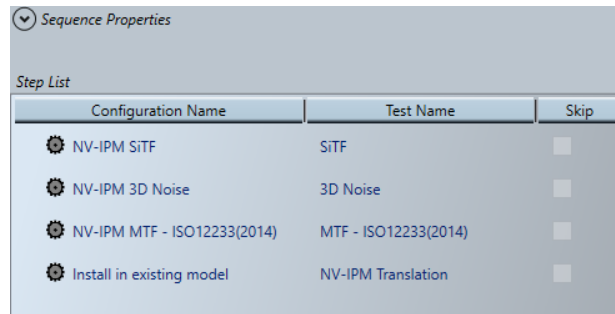


Figure 6. Test configurations for the SiTF, 3D Noise, and MTF measurements

Using the translation module developed in IRWindows, a test script was developed to collect camera parameters defined in the UUT Model and the measurements stored in the test history database from the completed tests. Unit translations were performed as needed, and then the collected values were stored into a designated NV-IPM model file.

Finally, all three test configurations, along with a configuration for the NV-IPM translation module, were packaged into a single test sequence and run from IRWindows (Figure 7).



Configuration Name	Test Name	Skip
NV-IPM SITF	SITF	<input type="checkbox"/>
NV-IPM 3D Noise	3D Noise	<input type="checkbox"/>
NV-IPM MTF - ISO12233(2014)	MTF - ISO12233(2014)	<input type="checkbox"/>
Install in existing model	NV-IPM Translation	<input type="checkbox"/>

Figure 7. Test Sequence to perform system measurements and then collect the results into an existing NV-IPM model

3.2 NV-IPM

Initially, a simple NV-IPM model was created to determine the performance of the camera (as a parametric model) with a sample observation task in a specified environment. The default V50 values for detection, recognition, and identification tasks were used in the TTP Metric across a set of ranges to generate the primary performance results (Figure 8 and Figure 9).

The model was then expanded to loop between the parametric camera model and an MSC (Figure 10). This permits a direct comparison of the predicted performance of the measured system to its modeled (idealized) behavior, as discussed in the next section.

Iteration:		Range to Target
<input checked="" type="checkbox"/>	Multi Input:	
<input type="checkbox"/>	Unit Input:	meter
<input checked="" type="checkbox"/>	Entry 1	1.0
<input checked="" type="checkbox"/>	Entry 2	100.0
<input checked="" type="checkbox"/>	Entry 3	200.0
<input checked="" type="checkbox"/>	Entry 4	300.0
<input checked="" type="checkbox"/>	Entry 5	400.0
<input checked="" type="checkbox"/>	Entry 6	500.0
<input checked="" type="checkbox"/>	Entry 7	600.0
<input checked="" type="checkbox"/>	Entry 8	700.0
<input checked="" type="checkbox"/>	Entry 9	800.0
<input checked="" type="checkbox"/>	Entry 10	900.0
<input checked="" type="checkbox"/>	Entry 11	1000.0
<input checked="" type="checkbox"/>	Entry 12	1100.0
<input checked="" type="checkbox"/>	Entry 13	1200.0
<input checked="" type="checkbox"/>	Entry 14	1300.0
<input checked="" type="checkbox"/>	Entry 15	1400.0
<input checked="" type="checkbox"/>	Entry 16	1500.0
<input checked="" type="checkbox"/>	Entry 17	1600.0
<input checked="" type="checkbox"/>	Entry 18	1700.0
<input checked="" type="checkbox"/>	Entry 19	1800.0
<input checked="" type="checkbox"/>	Entry 20	1900.0
<input checked="" type="checkbox"/>	Entry 21	2000.0

Figure 8. NV-IPM model ranges

Iteration:		V50
<input checked="" type="checkbox"/>	Multi Input:	
<input type="checkbox"/>	Unit Input:	
<input checked="" type="checkbox"/>	Detection	2.0
<input checked="" type="checkbox"/>	Recognition	7.5
<input checked="" type="checkbox"/>	Identification	13.0

Figure 9. NV-IPM model TTP metric specifications

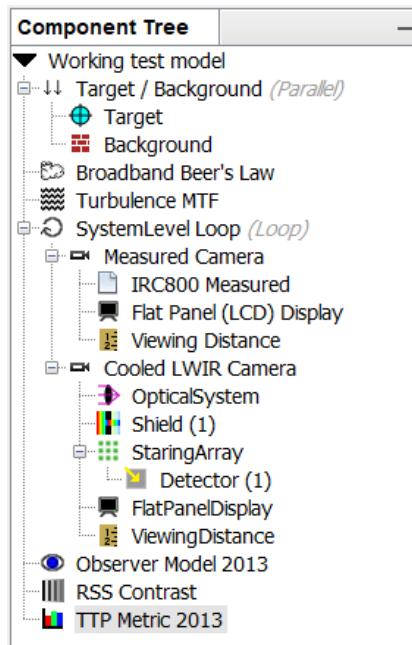


Figure 10. NV-IPM working test model

4. RESULTS

The specific camera that was tested has been quite extensively (ab)used in several labs at SBIR and has suffered damage over the years. Some optimization was done (windowing down the FPA, 2-pt NUC, focusing with a real-time MTF measurement, etc.) to peak the performance of the camera in order to get results as close to the expected (modeled) performance as possible.

We expected to perform an FOV test for an additional set of measured values inserted into the measured component, but a problem with the UUT mount prevented this from completing. Instead, we used the FOV calculated from the UUT parameters.

The test sequence was run using IRWindows and the previously described test equipment on the sample camera. The automated platform configured the various assets to stimulate the camera, collect the measurements from the camera, calculate the system performance values (see Figure 11, Figure 12, and Figure 13), and insert them into the NV-IPM model.

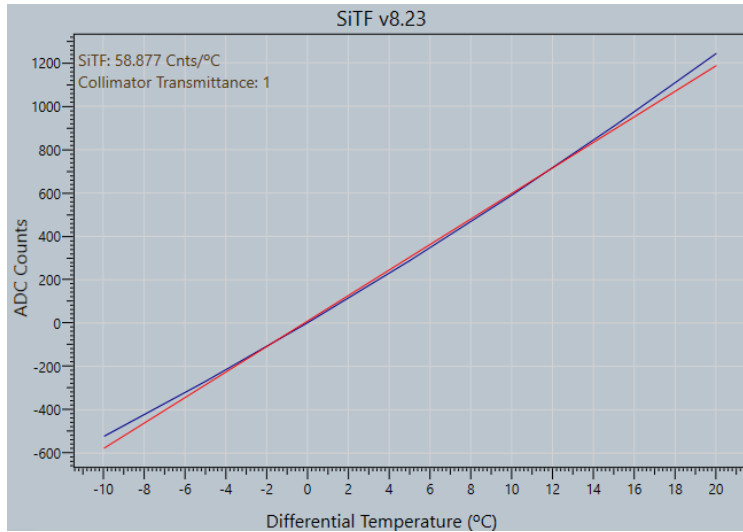


Figure 11. SiTF test results

Components	Counts	Noise (°C)	SiTF
TVH	2.785	0.047	58.877
VH	1.215	0.021	
TV	0.574	0.010	
TH	0.439	0.007	
T	0.521	0.009	
V	0.384	0.007	
H	0.349	0.006	
System Total	3.208	0.054	
Ratio VH/TVH	0.436		
Ratio V/TVH	0.138		
Ratio H/TVH	0.125		

Figure 12. 3D noise test results

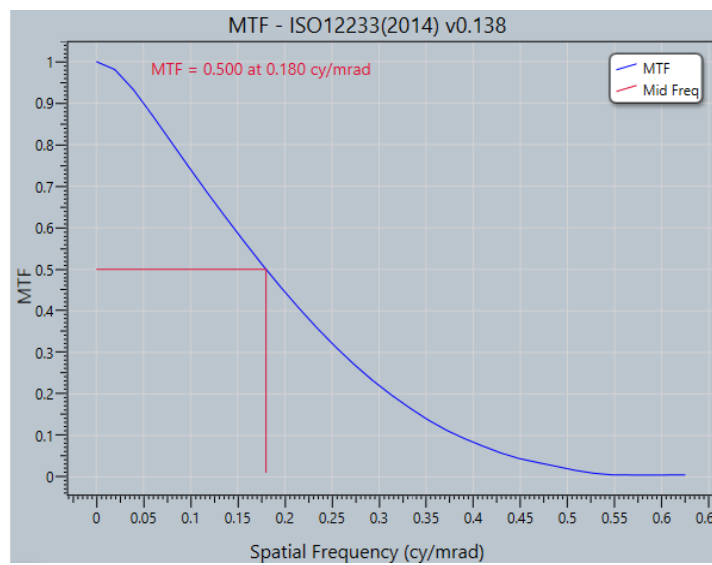


Figure 13. MTF test results

NV-IPM was started. The updated model was selected and then run. The results, produced by NV-IPM, are shown in Figure 14 (predicted performance from the measured values) and Figure 15 (predicted performance from the modeled values).

The divergent graphs of the two systems is somewhat expected due to the degraded performance of the measured camera. The noise is significantly high and the effective size of the FPA has been reduced due damage.

The important outcome is that a complete prediction was produced from both systems at once, and they can now be compared to identify discrepancies and determine the source of those discrepancies.

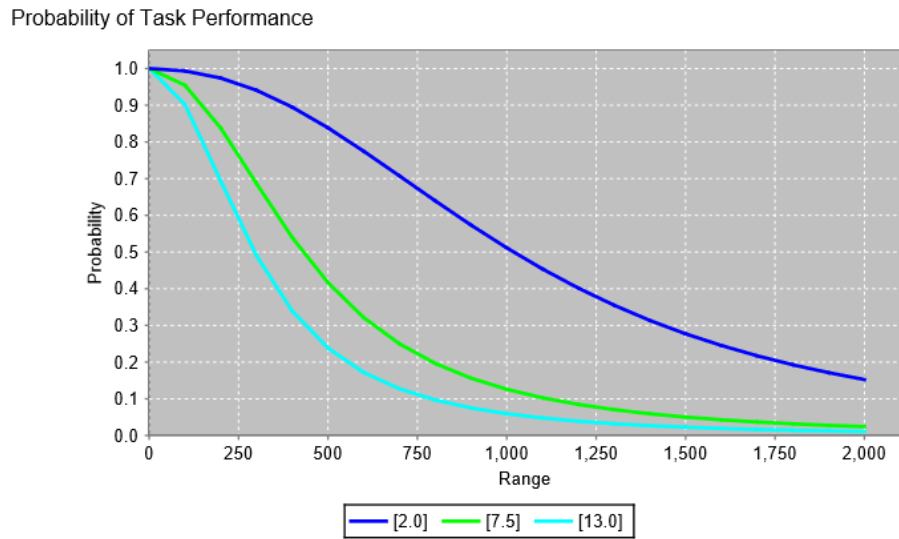


Figure 14. TTP predicted performance based on measured values of the IRC806 camera

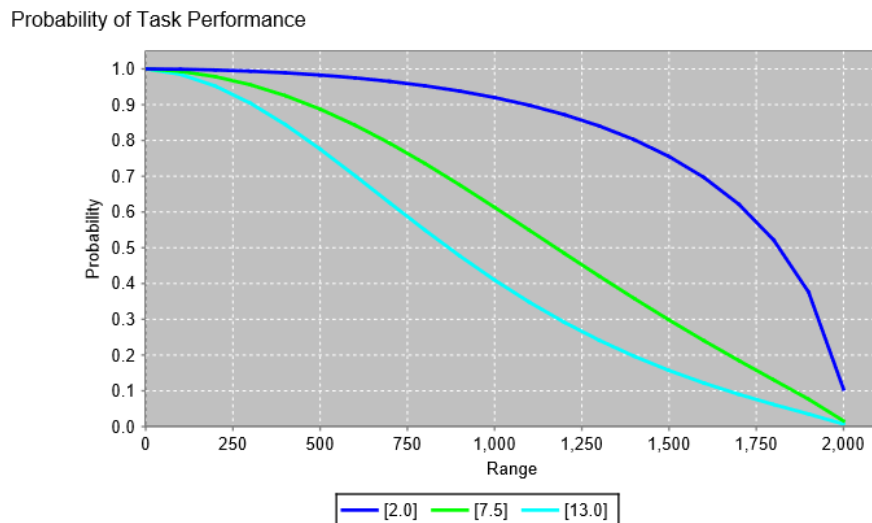


Figure 15. TTP predicted performance based on modeled values for the IRC806 camera

5. CONCLUSIONS

As indicated previously, this effort was largely an exercise in programming. Its primary intent was to show the feasibility of incorporating measured test results directly into a performance model in order to predict the performance of a measured system. Although we were able to meet these programming objectives, time limitations prevented us from extending our testing over a variety of sensor systems and in multiple laboratories.

The next steps are to enhance the robustness and flexibility of the interface so that we can demonstrate the capabilities over a variety of sensors systems. We will also need to demonstrate agreement between the predicted results from a model, and those results based on measured systems. We will then want to verify the agreement of our results across different laboratories.

We also anticipate that the wider use of the interface into NV-IPM by platforms such as NVLabCap⁷ will help validate the basic model as well as expand the API into NV-IPM⁸.

As this development goes forward, we hope to finally address the long-standing goal of determining the probability (or even feasibility) of whether a particular EO device would be able to accomplish a mission (as modeled) based on the measurements made just before being deployed⁶.

ACKNOWLEDGMENTS

I would like to thank David Haefner and Brian Teaney, both from NVESD, for their support of this project. They provided documentation, sample files, and the guidance necessary for any of the successes of this project. All problems and failures are due to the author, alone.

I would also like to thank my colleagues at SBIR who provided insights, support, and instruction throughout this development.

REFERENCES

- [1] Preece, Bradley L., Olson, Jeffrey T., Reynolds, Joseph P., Fanning, Jonathan D., "Improved noise model for the US Army sensors performance metric," *Infrared Imaging Systems: Design, Analysis, Modeling, and Testing XXII* (2011)
- [2] Fanning, Jonathan and Teaney, Brian, "Imaging system sensitivity analysis with NV-IPM," *Proceedings of SPIE - The International Society for Optical Engineering 9071:90710J* (2014)
- [3] Teaney, Brian and Haefner, David P., U S Army RDECOM CERDEC NVESD, direct communication (2018)
- [4] Irwin, Alan and Nicklin, Robert L., "Standard software for automated testing of infrared imagers, IRWindows, in practical applications," *Infrared Imaging Systems: Design, Analysis, Modeling, and Testing IX* (1998)
- [5] Errea, Steve, Grigor, J., King, D. F., Matis, Gregory P., McHugh, Steve W., McKechnie, James, Nehring, Brian, "Advanced E-O test capability for Army Next-Generation Automated Test System (NGATS)," *Infrared Imaging Systems: Design, Analysis, Modeling, and Testing XXVI* (2015)
- [6] McHugh, Steve W., Irwin, Alan, "Mission Assist Planning System for Field Prediction of EO Test Performance," *Tri-Services NxTest Working Group, presentation* (2011)
- [7] Burks, Stephen D., Doe, Joshua M., Haefner, David P., Teaney, Brian P., "NVLabCAP: an NVESD-developed software tool to determine EO system performance," *Infrared Imaging Systems: Design, Analysis, Modeling, and Testing XXV* (2014)
- [8] Teaney, Brian, Haefner, David P., "Evaluating the performance of an IR imaging system: A tutorial," *Infrared Imaging Systems: Design, Analysis, Modeling, and Testing XXIX* (2018)